Chapter 14
Scheduling Algorithms for Tree-Based Data Collection in Wireless Sensor Networks

Ozlem Durmaz Incel, Amitabha Ghosh, and Bhaskar Krishnamachari

Abstract Data collection is a fundamental operation in wireless sensor networks (WSN) where sensor nodes measure attributes about a phenomenon of interest and transmit their readings to a common base station. In this chapter, we survey contention-free *time division multiple access* (TDMA)-based scheduling protocols for such data collection applications over tree-based routing topologies. We classify the algorithms according to their common design objectives, identifying the following four as the most fundamental and most studied with respect to data collection in WSNs: (i) minimizing schedule length, (ii) minimizing latency, (iii) minimizing energy consumption, and (iv) maximizing fairness. We also describe the pros and cons of the underlying design constraints and assumptions and provide a taxonomy according to these metrics. Finally, we discuss some open problems together with future research directions.

14.1 Introduction

Data collection from a set of sensors to a common sink over a tree-based routing topology is a fundamental traffic pattern in wireless sensor networks (WSNs). This *many-to-one* communication pattern in which data flows from many nodes to a single node is known as *convergecast*. One may view convergecast as opposite to broadcast or multicast in which data flows from a single node to a set of nodes in the network. Figure 14.1 shows a simple example that illustrates the characteristics of a typical broadcast and convergecast. In broadcast, as shown in Fig. 14.1a, node s is the message source and nodes a, b, and c are expected recipients. Node a hears the message directly from s and forwards a copy to nodes b and c. In case of a convergecast, as shown in Fig. 14.1b, nodes a, b, and c each has a message destined to the sink node s and a serves as a relay for b and c.

O. Durmaz Incel (✉)
NETLAB, Department of Computer Engineering, Bogazici University, 34342 Bebek, Istanbul, Turkey
e-mail: ozlem.durmaz@tam.boun.edu.tr

S. Nikoletseas, J.D.P. Rolim (eds.), *Theoretical Aspects of Distributed Computing in Sensor Networks,* Monographs in Theoretical Computer Science. An EATCS Series,
DOI 10.1007/978-3-642-14849-1_14,

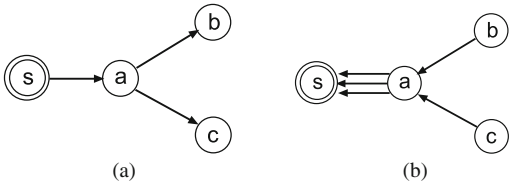


Fig. 14.1 (**a**) Broadcast — data flows from a single node s to a set of nodes a, b, c. (**b**) Convergecast — data flows from nodes a, b, and c to a single node s

Once data is collected at the sink, it either can be recorded and stored for future analysis or can be processed immediately to take certain actions depending on application requirements. In a WSN, data collection either can be triggered by external sources, such as *queries* to get a snapshot view of the network or *events* as and when they appear, or can be for continuous periodic monitoring without any external triggering. In all cases, however, the many-to-one communication pattern is common.

Depending on application requirements, different objectives can be associated with data collection. For instance, in disaster early warning applications, such as detection of forest fire [73] and gas/oil leaks [14], or structural damage identification [9], bursty traffic generated by events needs to be delivered to the sink as quickly and as reliably as possible to prevent catastrophes. On the other hand, in applications where sensor nodes only report periodic data, such as animal habitat monitoring [50], energy efficiency may become a more important concern as opposed to quick data collection.

Particularly under regular, heavy traffic conditions, contention-free medium access control (MAC) protocols, such as *time division multiple access* (TDMA), where nodes communicate on different time slots to prevent conflicts, offer several advantages for data collection as compared to contention-based protocols [27]. They eliminate collisions, overhearing, and idle listening, which are the main sources of energy consumption in wireless communications [15]. In addition, they also permit nodes to enter into sleep modes during inactive periods, thus achieving low duty cycles and conserving energy. Furthermore, TDMA-based communications can provide provable guarantee on the completion time of data collection, for instance, in timely detection of events. Another key aspect of time-slotted communication is robustness during peak loads. When the number of source nodes are many or the data rates are high, carrier-sense multiple access protocols, such as CSMA, may fail to successfully allocate the medium, causing retransmissions and collisions. A number of TDMA-based MAC protocols for WSNs have been proposed to exploit these advantages [2, 20, 36, 58, 61].

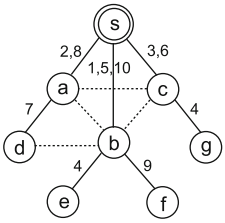
In this chapter, we survey TDMA-based scheduling algorithms for data collection in sensor networks. We first classify the algorithms based on their common objectives and then identify different design constraints and assumptions and provide a taxonomy according to these metrics. We identify the following four objectives as the most studied in literature and the most fundamental with respect to data collection in WSN: (i) minimizing schedule length, (ii) minimizing latency,

(iii) minimizing energy consumption, and (iv) maximizing fairness. We also find that some algorithms target joint optimization of multiple objectives, such as minimizing delay as well as energy. We note that there exist many other studies, not necessarily about convergecast, that focus on these objectives; however, in this chapter, we consider only studies that use TDMA scheduling for tree-based data collection.

In terms of design constraints and assumptions, the algorithms differ mainly in the following dimensions: (a) use of communication and interference models, (b) implementation methods, such as centralized or distributed, (c) topology assumptions, (d) types of data collection, such as use of in-network processing versus raw data, and (e) capability of transceivers available on the sensor nodes. We briefly explain the fundamentals of the algorithms, give the details of some of the highlighted ones, discuss the advantages and disadvantages of the algorithms, and comment on the pros and cons of the design constraints and assumptions. For instance, most of the scheduling algorithms use the *protocol model* [34] for interference, which is a graph theoretic approach that assumes correct reception of a message if and only if there is no other simultaneous transmission within proximity of the receiver. Although this enables the use of simple graph coloring-based scheduling schemes, the model is idealistic and may fail in practice, because interference is not a binary phenomenon, and two nearby concurrent transmissions can actually be successful if the interference level is tolerable. To this end, the use of the *physical model*, which is based on the *signal-to-interference-plus-noise-ratio* (SINR), provides a better solution in terms of realistic capturing of interference from multiple transmissions, thus resulting in correct and realizable schedules [33, 38].

Due to its ability to provide time bounds, TDMA-based scheduling algorithms are widely exploited for *fast* and *timely* delivery of data with the objective of minimizing the time to complete convergecast, i.e., minimizing the latency. In a TDMA schedule, time is slotted and each slot is long enough for transmission or reception of a single packet. Consecutive time slots are grouped into non-overlapping frames, and the schedule for each frame is repeated when data collection is periodic. It is assumed that some form of time synchronization exists among the nodes, which can be achieved using one of the protocols such as [19]. Under this setting, minimizing the data collection time for (aggregated/raw-data) convergecast is equivalent to minimizing the number of time slots required per frame, called the *schedule length*, such that all (aggregated/raw) packets from the source nodes reach the sink.

Since multi-hop TDMA allows spatial reuse of time slots, more than one node can transmit simultaneously if their receivers are in non-conflicting parts of the network. There are two types of conflicts that arise: (i) *primary conflict* and (ii) *secondary conflict*. A primary conflict occurs when a node transmits and receives at the same time or receives more than one transmissions destined to it at the same time. This is due to the single, half-duplex transceiver on each node, which is typical of current WSN hardware [4, 62]. A secondary conflict occurs when a node, an intended receiver of a particular transmission, is also within the range of another transmission intended for other nodes. For instance, in Fig. 14.2, nodes b and d cannot be scheduled simultaneously under the protocol interference model because b's transmission, which is intended for sink s, will interfere with d's transmission at



	Frame									
slots	1	2	3	4	5	6	7	8	9	10
nodes	b	a	c	e,g	b	c	d	a	f	b

Fig. 14.2 Raw-data convergecast that takes 10 time slots to complete. Table shows the schedule

a due to the presence of the interfering link, as indicated by the dotted line between *b* and *a*. For the same reason, nodes *d* and *e* and nodes *a* and *b* cannot be scheduled simultaneously. The figure also illustrates raw-data convergecast where the routing tree rooted at sink *s* is indicated by the solid lines. The table on the right shows the schedule for one frame with the minimum possible schedule length of 10 slots.

The rest of the chapter is organized as follows: Sect. 14.2 explains the details of the followed classification methodology. Section 14.3 presents the existing work on scheduling algorithms for data collection with different objectives and comparisons, along with a taxonomy. Section 14.4 discusses open problems and future research directions. Section 14.5 draws some relevant conclusions.

14.2 Classification Approach and Methodology

Our surveyed algorithms have the common purpose of "*data collection using TDMA-based communication schedules in WSNs*." However, they differ according to their objectives, which are usually set by varying application requirements and the underlying design constraints and assumptions. Our classification methodology is primarily based on the design objectives. In this section, we summarize the most studied objectives, describe their importance in the context of data collection applications, explain the underlying design constraints and assumptions, and comment on the pros and cons of alternative models or approaches that can be used.

14.2.1 Design Objectives

1. ***Minimizing Schedule Length***: Minimizing the schedule length or, equivalently, minimizing the time to complete convergecast, is the most studied design objective for data collection in sensor networks. It translates to quicker data collection at a fast rate. In many WSN applications, it is of interest to maximize the rate at which the sink can receive data from the network [52]. For instance, it is noted that in networked structural health monitoring, more than 500 samples per second are required to efficiently detect and localize damages [9]. Second, a minimal length TDMA schedule allows for a longer sleep period in each data collection cycle, especially for periodic traffic, which contributes to lesser energy

consumption on the sensor nodes. Minimal schedule length can be achieved by maximizing the reuse of the time slots. Therefore, most of the existing algorithms aim to maximize the number of concurrent transmissions and enable spatial reuse by devising strategies to eliminate interference. In certain cases, a minimal schedule length may also contribute to minimizing the latency of data collection. However, the transmission sequence and the number of hops to reach the sink should also be jointly considered as factors impacting the latency.

2. ***Minimizing Latency***: Minimizing the data collection latency is important for applications that are required to take certain (precautionary) actions based on deadlines, such as mission-critical and event-based applications. Although minimizing the schedule length may as well minimize the latency under certain conditions, most algorithms do not consider the *average* latency experienced by individual packets at each hop. Moreover, the ones that aim to minimize the schedule length achieve this by maximizing the reuse of time slots, which, however, for some topologies does not necessarily result in schedules with minimal delay. For instance, a line topology may allow higher spatial reuse, but due to larger number of hops from the sources to the sink, it may cause high latency. Therefore, minimizing the data collection latency may require considering additional constraints in addition to those required for minimizing the schedule length.
3. ***Minimizing Energy Consumption***: Minimizing energy consumption and maximizing network lifetime are fundamental to successfully operating resource-constrained WSNs for long durations. As a major source for energy consumption, radio activity should be managed efficiently. The most common method is to operate the radio in duty cycles with periodic switching between sleep and wake-up modes, which can be easily incorporated with TDMA schedules by maximizing the sleep time. Transmission power control is another well-known technique to reduce energy consumption, contention, and packet losses. Instead of transmitting at maximum power, sending packets at an optimal power level can save energy and extend network lifetime. It is well known that wireless links exhibit variable link qualities over time (due to multi-path effects, fading, etc.), and transmission power control can transform bad quality links into good ones with high packet delivery rates. In addition, packet losses due to contention during peak traffic periods can also be mitigated by power control.
4. ***Maximizing Capacity***: Although maximizing the throughput capacity is not considered to be one of the primary objectives in low-rate data collection applications over small networks, it is important for large, dense sensor networks and for complex applications that require efficient delivery of large amounts of data. The performance of a data-gathering WSN can be characterized by the rate at which information can be delivered to the sink [52]. However, maximizing capacity and minimizing energy consumption are conflicting to each other, and so studying their trade-off is an interesting topic for complex data-gathering applications.
5. ***Maximizing Fairness***: Fairness is one of the key objectives in WSN applications in order to maintain a balanced view of the sensor environment. In applications where each of the sensor readings is important, fairness becomes an important

issue, especially under high data rates. For instance, fair data gathering may become necessary for reducing the estimation error in an application involving field reconstruction.

6. ***Other Objectives***: Minimizing communication costs, maximizing parallel transmissions, meeting deadlines, minimizing interference, and self-stabilization are some of the other objectives studied for data collection in WSNs.
7. ***Joint Objectives***: In most applications, there is not always a single objective, but often multiple, and sometimes conflicting, objectives involved. Some examples include minimizing communication cost and delay, minimizing energy consumption and completion time of data collection, maximizing capacity and minimizing energy. Scheduling algorithms need to address the optimal trade-offs in satisfying these conflicting objectives.

14.2.2 Design Constraints and Assumptions

In this section, we discuss the underlying design constraints and assumptions that the algorithms are based on, ranging from communication models to hardware issues.

1. ***Communication and Interference Models***: In the literature, there are two common approaches to model interference: (i) *protocol model* and (ii) *physical model* [34], also known as the SINR model. The protocol model states that a message is correctly received if there is no other sender transmitting at the same time within a close proximity of the intended receiver. The advantage of this approach is that it enables the use of simple graph coloring-based scheduling algorithms. In [33], Gronkvist et al. analyze the performance of the protocol interference model and indicate that it does not always provide a comprehensive view of reality due to the cumulative effects of interference in wireless networks. The model can also be pessimistic at times, such as when two nearby communications, which are concurrently not admissible by protocol model constraints, can actually take place if the interference level is tolerable. Other models, such as those based on RTS/CTS or hop counts, are extensions or special cases of the protocol model. The physical model, on the other hand, is richer in the sense that it can capture cumulative interference from multiple concurrent transmissions and considers a message to be successfully received if the SINR at the receiver is greater than a certain threshold. Moscibroda [52] studies the impact of the physical model on the achievable capacity in wireless multi-hop networks and shows that protocols designed with the SINR model can surpass the theoretically achievable performance of graph-based scheduling protocols.
2. ***Types of Data Collection***: We identify two fundamental types of data collection in WSN: (i) *raw-data convergecast*, where every packet is relayed individually and (ii) *aggregated convergecast*, where packets are aggregated at each hop before being relayed. Aggregated convergecast is applicable when a strong correlation exists in the sensor readings or the goal is to collect summarized information, such as the maximum sensor reading. Raw-data convergecast, on the other hand, is applicable when every measurement is equally important or

the correlation is minimal. These two types correspond to two extreme cases of data collection in sensor networks.

3. ***Network Topology***: Tree-based routing topologies are most common in many-to-one data collection; however, line, star, or dynamic routing topologies have also been considered. Many of the works assume a fixed topology while some others consider dynamic routing, where the next-hop forwarding node is selected based on some criteria, such as battery level or link reliability.
4. ***Sensor Deployment***: The sensor deployment method usually varies with application requirements. Besides the commonly used random and grid deployments, some applications may support redeployment of nodes, for instance, to eliminate sensing holes in the network [29].
5. ***Buffer Size***: As nodes generate and forward packets toward the sink, they may need to buffer the packets before transmissions. Although some algorithms assume unlimited buffer sizes, minimizing the buffer size can offer advantages considering the limited memory resources available at the nodes.
6. ***Transceiver Assumptions***: Each sensor node is typically equipped with a single, omnidirectional radio that can be tuned to a single channel at any given time. However, radios with multiple transceivers that can support multiple channels and directional antennas are also studied in the literature.
7. ***Implementation Method***: While some algorithms rely on the sink node to compute schedules in a centralized way, others take a distributed approach where nodes compute their schedules based on local information exchanged in their neighborhood.
8. ***Use of Joint Solutions***: Use of TDMA scheduling together with transmission power control, multi-channel scheduling, specific routing solutions are common approaches in the surveyed algorithms.
9. ***Data Collection Pattern***: Data collection can be periodic or one shot. One-shot data collection is typically query based that provides a snapshot of the monitored conditions or event based where nodes report data if an event of interest occurs.
10. ***Granularity of Assignments***: There are two general methods in TDMA scheduling: (i) *node scheduling*, also referred to as broadcast scheduling, and (ii) *link scheduling*, also known as link activation or point-to-point scheduling. In broadcast scheduling, time slots are assigned to the nodes, and a node's transmission is intended for all its neighbors [59]. In link scheduling, individual links are scheduled such that a transmission is intended for and must be received collision free by a particular neighbor. Most of the algorithms use link scheduling in convergecast.

14.3 Scheduling Algorithms for Data Collection

14.3.1 Algorithms on Minimizing Schedule Length

In this section, we survey the TDMA-based scheduling algorithms that identify minimizing the schedule length as their primary objective. We first describe works

that pertain to raw-data convergecast and then focus on aggregated convergecast. Since packets are aggregated at each hop in aggregated convergecast, the number of packets transmitted and delivered to the sink is substantially lower than that of raw-data convergecast.

14.3.1.1 Raw-Data Convergecast

For raw-data convergecast, finding a minimum-length, conflict-free assignment of time slots, such that every packet generated by a node reaches the sink, is fundamental to efficient network operations. Several variants of the problem exist depending on network topology, interference model, packet generation scheme, number of frequency channels, buffer constraints, antenna models, etc.

One of the early works in this category is by Florens et al. [22–24], who address the problem of scheduling for packet *distribution* in sensor networks and argue that it can be considered as an inverse of the convergecast problem. Assuming the protocol interference model, they propose optimal centralized algorithms for special network topologies, such as line, multi-line, and tree networks, for both omnidirectional and directional antennas.

For packet distribution in line networks, where the sink sends $p(i) \geq 0$ packets to node i which is i hops away, the basic idea is to first transmit packets destined to the furthest node, then packets for the second furthest node, and so on, as quickly as possible respecting channel reuse constraints. Nodes between the sink and a packet's destination are required to forward that packet as soon as it arrives (i.e., in the next time slot following its arrival). This basic idea can be extended to a multi-line network and a tree network as well. The upper part of Fig. 14.3 shows an example of scheduling for packet distribution on a 10-node line network for directional antennas with $p(i) = 2$, $p(2) = 1$, $p(8) = 1$, and $p(9) = 1$. Once an optimal schedule is found, the schedule for the inverse problem of convergecast where node i sends $p(i)$ packets to the sink is constructed by symmetry as shown in the bottom part of Fig. 14.3. In particular, a transmission from node i to $i + 1$ occurring at time slot j for the distribution problem corresponds to a transmission from node $i + 1$ to i in time slot $T - j + 1$ for the convergecast problem. Here, N is the total number of nodes and T is the minimal schedule length which for omnidirectional antennas is given by

$$T = \max_{1 \leq i \leq N} \left(i - 1 + p(i) + 2p(i+1) + 3 \sum_{j \geq i+2}^{N} p(j) \right) \tag{14.1}$$

and for directional antennas is given by

$$T = \max_{1 \leq i \leq N-1} \left(i - 1 + p(i) + 2 \sum_{j \geq i+2}^{N} p(j) \right) \tag{14.2}$$

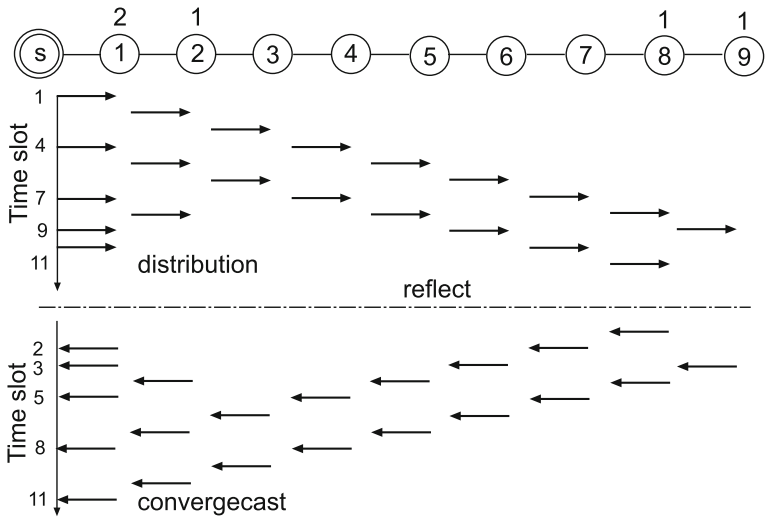


Fig. 14.3 Optimal time scheduling for a 10-node line network with minimum schedule length 11. *Upper part* shows the schedule for packet distribution, and *bottom part* shows the schedule for convergecast, which is obtained by symmetry, i.e., by reflecting the upper schedule with respect to the fictitious *horizontal line*. Note that nodes that are closer than or at two hops do not transmit concurrently to respect interference issue

Ergen et al. in [20] prove that the problem of minimizing the schedule length is NP-complete by reducing it from the *graph coloring* problem. The scheduling difficulty arises since many subsets of non-conflicting nodes are candidates for transmission in each time slot, and the subset chosen in one slot affects the number of transmissions in the next slot. This is due to the fact that some eligible nodes may not have any packet to transmit because of the subset selected in the previous slot. When a graph-based interference model is used, a conflict-free schedule can be found by coloring a *conflict graph*. A conflict graph is one in which every node represents an edge in the original graph and two nodes are connected if their corresponding edges interfere in the original graph, i.e., give rise to primary or secondary conflicts. Using such a graph coloring strategy, they propose a *node-based* and a *level-based* scheduling heuristic and show that one outperforms the other depending on the distribution of packet generation in the network. In particular, node-based scheduling is better for topologies that have equal density of packets across the network or higher density of packets at low levels of the tree, whereas level-based scheduling is better for topologies when the packet density is higher at the upper levels of the tree.

A *virtual node expansion*-based approach that also uses graph coloring to find a minimum-length, conflict-free schedule where every node generates a single packet in each frame is proposed by Lai et al. [41]. They first construct a conflict graph from the original graph and then expands it by creating, for each parent node, a number of virtual nodes equal to the size of the subtree rooted at that node in the original

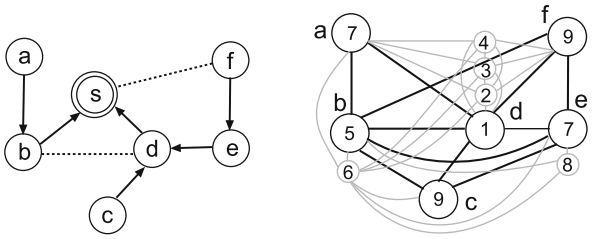


Fig. 14.4 (**a**) The original graph and the routing tree: *arrows* indicate tree edges and *dotted lines* represent interfering links. (**b**) *Dark circles* and *lines* represent nodes and edges in the conflict graph, which is expanded by adding virtual nodes and virtual edges, marked in *gray*

tree. As illustrated in Fig. 14.4b, four virtual nodes, marked as 1, 2, 3, and 4, are created for node d in the *expanded conflict graph* as the size of the subtree rooted at d is four in the original graph, as shown in Fig. 14.4a. This graph expansion is done to accommodate multiple transmissions by intermediate parent nodes which relay packets from nodes in its subtree. Since the virtual nodes also conflict with any node that has an edge to its original node, edges are added between the virtual nodes and the conflicting node. Similarly, an edge is added between each virtual node and its original node.

Once the expanded conflict graph is constructed, an approximate coloring algorithm, originally due to Li et al. [45], is used to find a time slot assignment. The coloring algorithm works by finding a vertex with the least degree and removing it from all its adjacent edges. This is repeated until all the vertices are removed, after which it greedily assigns colors in the reverse order of removing the vertices. This results in a conflict-free schedule for each of the edges in the original graph. As illustrated in Fig. 14.4b, the following schedule is generated: d transmits in slots 1, 2, 3, and 4[1]; b transmits in slots 5 and 6; a transmits in slot 7; e transmits in slots 7 and 8; and c and f both transmit in slot 9. It is shown that the schedule length achieved by this coloring algorithm is no more than $2\delta/k + 1$, where δ is the largest degree in any subgraph of the conflict graph in which every vertex has a degree at least δ and k is the maximum size of an independent set in the neighborhood of any node in the conflict graph.

The authors of this paper also discuss the effect of different routing structures on the schedule length and propose a *disjoint-strip*-based approach to construct an efficient routing topology. This results in uniform flow of data along different paths in the network and prevents certain nodes from being overloaded. The basic idea is

[1] There is no causality constraint, such that node d does not need to wait for data from its children before being scheduled, and the data collection is periodic.

to construct several disjoint, equally spaced *node strips*, all with the same number of nodes. Two distanced strips are likely to relay data simultaneously without interfering with each other. It is shown that this disjoint-strip routing, although increases the total number of transmissions, yields a shorter schedule length for unbalanced node deployments as compared to shortest path routing, which is suitable for balanced deployments minimizing the total number of transmissions but not necessarily the schedule length.

Choi et al. in [12] formulate the scheduling problem as a *minimum information gathering time problem*, where every node has a single packet to send and the goal is to find routing paths from the nodes to the sink as well as an optimal time slot assignment. By reducing it from the *partition problem*, they prove that the problem is NP-complete on general graphs and propose algorithms for line and tree topologies that take at most $3N - 3$ time slots to deliver all the packets to the sink. For general networks a heuristic is proposed, which starts with a minimum spanning tree and trims the edges such that transmissions on different branches of the tree do not interfere with each other and can be scheduled in parallel. This results in a backbone forest whose segments are then scheduled independently respecting adjacency and two-hop interfering constraints.

Following a strategy similar to [24], Gandham et al. in [26, 27] propose distributed scheduling algorithms for raw-data convergecast where every node generates a single packet in one data collection cycle. They give an *integer linear programming* (ILP) formulation of the problem and propose a distributed time slot assignment scheme that takes (i) at most $3N - 3$ time slots for linear networks, which is optimal, (ii) at most $\max(3n_k - 1, N)$ time slots for multi-line and tree networks, where the lower bound for multi-line networks is $\max(3n_k - 3, N)$, and (iii) at most $3N$ time slots for general networks. Here n_k represents the maximum number of nodes in any subtree of the routing structure. Similar results are also obtained by Tsai et al. [69]. We note that (14.1) reduces to $3N - 3$ when $p_i = 1$ for $i = 1, \ldots, N$, which matches with Gandham's result for line networks.

In addition to minimizing the schedule length, the proposed algorithm in [27] also considers memory constraints on the sensor nodes and requires storage for at most two packets in each node buffer. Links are assumed to be symmetric and the interference model is assumed to be graph based, with the interference range of a node equal to its transmission range. Their results also extend to the case where nodes generate multiple packets and when channel propagation characteristics are not ideal. In the following, we first give their ILP formulation and then briefly explain the details of the algorithm.

Let $G = (V, E)$ represent the sensor network, where V is the set of nodes including the sink s and E is the set of wireless links. Let $p_0(v)$ be the number of packets originated at node v and $p_t(v)$ be the number of packets at node v at the end of slot t. Let $f_t(u, v) \in \{0, 1\}$ represent the state of the link (u, v) at slot t; $f_t(u, v) = 1$ if node u transmits a packet to node v in slot t and 0 otherwise. Let $N(v)$ be the number of one-hop neighbors of node v and T be the number of time slots to complete convergecast. Then the ILP is given by the following:

$$\textbf{Minimize} \quad T$$

$$\text{subject to} :$$

$$p_T(s) = \sum_{u \in V} p_0(u) \tag{14.3}$$

$$\forall u \in V,\ \forall t \in \{1, \ldots, T\} : \quad \sum_{w \in N(v)} \sum_{v \in N(u)} f_t(v, w) \leq 1 \tag{14.4}$$

$$\forall u \in V,\ \forall t \in \{1, \ldots, T\} : \quad \sum_{v \in N(u)} f_t(u, v) \leq 1 \tag{14.5}$$

$$\forall u \in V,\ \forall t \in \{1, \ldots, T\} : \quad \sum_{v \in N(u)} (f_t(v, u) + f_t(u, v)) \leq 1 \tag{14.6}$$

$$\forall u \in V,\ \forall t \in \{1, \ldots, T\} : \quad p_t(u) + \sum_{v \in N(u)} f_t(u, v) = p_{t-1}(u) \tag{14.7}$$

$$\forall u \in V,\ \forall t \in \{1, \ldots, T\} : \quad p_t(u) - \sum_{v \in N(u)} f_t(v, u) = p_{t-1}(u) \tag{14.8}$$

$$\forall u, v \in V : \quad f_t(u, v) \in \{0, 1\} \tag{14.9}$$

The objective is to minimize T, the total number of time slots required to complete convergecast. Constraint (14.3) ensures that all packets are delivered to the sink at the end of convergecast. Constraint (14.4) states that at most one neighbor of a node can transmit in a slot, thus addressing the hidden terminal problem. Similarly, constraint (14.5) states that a node transmits to at most one neighbor, and constraint (14.6) restricts a node from both transmitting and receiving in the same time slot. Constraints (14.7) and (14.8) are for conservation of messages, and constraint (14.9) guarantees an integral solution. The above ILP can be solved using tools like CPLEX; however, typically solutions to ILP's have exponential running time. Moreover, such a solution will be centralized and not scalable in nature. Hence combinatorial solutions are preferred.

Starting from linear networks, the algorithm proposed by Gandham et al. is generalized for multi-line networks, which are multiple linear networks intersecting with the sink, and also to tree networks. The basic idea for linear networks is that each node is assigned an initial state depending on its hop count from the sink. As illustrated in Fig. 14.5a, a node at hop distance h is assigned a state of transmitting (T) if h mod 3 is 1, idle (I) if h mod 3 is 2, and receiving (R) if h mod 3 is 0. A node comes back to this initial state after every three time slots and follows the state transition diagram as shown in Fig. 14.5b. Effectively, this implies that for every node in state R, there is only one node in the neighborhood which is in state T, and so packet transmission is always successful resulting in exactly one packet reception by the sink in every three time slots.

The above basic idea extends to more complex topologies, such as multi-line networks, as shown in Fig. 14.5c, tree networks where transmissions are scheduled at parallel along multiple branches, and general networks where packets are routed over a breadth first search (BFS) tree. However, for the distributed algorithm to

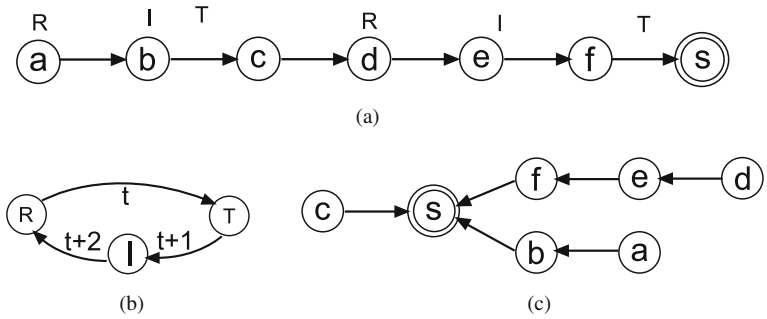


Fig. 14.5 (**a**) A linear network with initial state assignment (R, I, T) depending on the hop distance from the sink s. (**b**) State transition of the nodes. (**c**) A multi-line network as a composition of multiple linear networks

work, each node must know the branch ID and the number of nodes in all the other branches, but need not be aware of the entire network topology. The scheduling rule for multi-line networks is that the branch with the largest number of remaining packets and whose root has at least one packet gets priority to transmit to the sink (ties are broken based on the lowest branch ID). This results in a schedule length of $\max(3n_k - 1, N)$. For general networks, since there are interfering edges that are not part of the spanning tree, the goal is to first eliminate interference by constructing a BFS tree and then scheduling as before. However, in addition to knowing the number of nodes in all the branches and the branch ID, for general networks, a node also has to know a conflict map at the initialization phase. This gives a schedule length of $3N$, although the simulations presented in the paper require only $1.5N$ time slots.

The use of orthogonal codes, such as direct sequence spread spectrum (DSSS) and frequency hopping spread spectrum (FHSS), to eliminate interference is studied by Annamalai et al. [1]. They propose a greedy, top–down tree construction scheme that chooses the children of a node based on the nearest neighbor criterion starting from the sink and traversing the graph in BFS order. To reduce interference, nodes that fall within the transmission range of a parent other than their own are assigned different codes if available; otherwise, the code that is least used by the interfering neighbors is used. Once the channel allocation is done, time slots are assigned in a greedy fashion such that a parent does not transmit before its children. Simulation results indicate that the schedule length on such a tree constructed specifically for convergecast is shorter than on a tree constructed for broadcast [11]. However, one limitation of this approach is that the miniature hardware design of sensor nodes may not permit employing complex radio transceivers required for spread spectrum codes or frequency bands systems.

In [37], Incel et al. explore and evaluate a number of different techniques using realistic simulation models to study the data collection rate for raw-data convergecast. First, a simple spatial-reuse TDMA scheme is employed to minimize the schedule length, which is then combined with multiple frequency channels and transmission power control to achieve further improvement. A receiver-based channel

assignment (RBCA) scheme is proposed where the receivers (i.e., parents) of the tree are statically assigned a channel and the children of a common receiver transmit on that channel. This avoids pairwise, per-packet channel negotiation overheads. Once multiple frequencies are used to completely eliminate interference (i.e., secondary conflicts), it is shown that the lower bound on the schedule length is $\max(2n_k - 1, N)$, and a time slot assignment scheme is proposed that achieves this bound with no nodes requiring to buffer more than one packet at any time. Here, as in [27], n_k is the maximum number of nodes in any branch of the tree.

Next, the authors of [37] show that once interference is eliminated, the data collection rate often becomes limited by the routing topology. To overcome this, trees with specific properties are constructed, which help in further enhancing the data collection rate. In particular, *capacitated minimum spanning trees* [57], which aim to have an equal number of nodes on each branch, are shown to achieve a factor of two improvement as compared to single-channel TDMA scheduling on minimum-hop shortest path trees.

Algorithm 1 LOCAL-TIMESLOTASSIGNMENT

```
1.  node.buffer = full
2.  if {node is sink} then
3.      Among the eligible top-subtrees, choose the one with the largest number of total
        (remaining) packets, say top-subtree i
4.      Schedule link (root(i), s) respecting interfering constraint
5.  else
6.    if {node.buffer == empty} then
7.        Choose a random child c of node whose buffer is full
8.        Schedule link (c, node) respecting interfering constraint
9.        c.buffer = empty
10.       node.buffer = full
11.   end if
12. end if
```

The key idea behind our algorithm, which is formally presented in Algorithm 1 as LOCAL-TIMESLOTASSIGNMENT, is to (i) schedule transmissions in parallel along multiple branches of the tree and (ii) keep the sink busy in receiving packets for as many time slots as possible. Each node maintains a buffer and its associated state, which can be either *full* or *empty* depending on whether it contains a packet or not. Initially, all the buffers are full because every node has a packet to send.

The first block of the algorithm in lines 2–4 gives the scheduling rules between the sink and the roots of the top-subtrees. A *top-subtree* TS(r) is defined as one whose root r is a child of the sink, and it is said to be *eligible* if r has at least one packet to send. For instance, in Fig. 14.6a, the top-subtrees are $\{1, 4\}$, $\{2, 5, 6\}$, and $\{3, 7\}$. For a given time slot, the root of an eligible top-subtree which has the *largest* number of total remaining packets is scheduled. If none of the top-subtrees are eligible, the sink does not receive any packet during that time slot. Inside each top-subtree, nodes are scheduled according to the rules in lines 5–12. A subtree

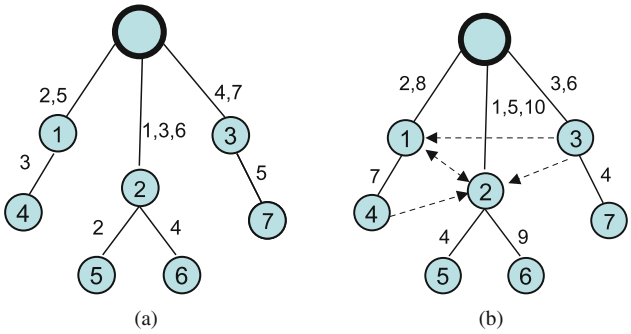


Fig. 14.6 Raw-data convergecast using algorithm LOCAL-TIMESLOTASSIGNMENT: (**a**) Schedule length 7 when secondary conflicts are eliminated. (**b**) Schedule length 10 when secondary conflicts are present

is defined to be *active* if there are still packets left in it (excluding its root) to be relayed. If a node's buffer is empty and the subtree rooted at this node is active, one of its children is scheduled at random whose buffer is not empty. The algorithm guarantees that in an active subtree there will always be at least one child whose buffer is not empty, and so whenever a node empties its buffer, it will receive a packet in the next time slot, thus emptying buffers from the bottom of the subtree to the top.

Figure. 14.6a shows an illustration of the working of the algorithm. In slot 1, since the eligible top-subtree containing the largest number of remaining packets is $\{2, 5, 6\}$, link $(2, s)$ is scheduled and the sink receives a packet from node 2. In slot 2, the eligible top-subtrees are $\{1, 4\}$ and $\{3, 7\}$, both of which have two remaining packets. We choose one of them at random, say $\{1, 4\}$, and schedule the link $(1, s)$. Also, in the same time slot since node 2's buffer is empty, it chooses one of its children at random, say node 5, and schedule the link $(5, 2)$. In slot 3, the eligible top-subtrees are $\{2, 5, 6\}$ and $\{3, 7\}$, both of which have two remaining packets. We choose the first one at random and schedule the link $(2, s)$, and so the sink receives a packet from node 5 (relayed by node 2). We also schedule the link $(4, 1)$ in slot 3 because node 1's buffer is empty at this point. This process continues until all the packets are delivered to the sink, yielding an assignment that requires seven time slots. Note that, in this example, $2n_k - 1 = 5$, and so $\max(2n_k - 1, N) = 7$. In Fig. 14.6b, an assignment is shown when all the interfering links are present, yielding a schedule length of 10.

A similar result of $\max(2n_k - 1, N)$ is obtained by Song et al. [66] where they also extended it to the case when the nodes have different number of packets to send. Assuming node i generates d_i packets, their proposed algorithm takes $\max\left(2\sum_{i\in \mathrm{TS}(r_k)} d_i - d_{r_k} + \sum_{i=2}^{k}(d_{r_i} - 1), N'\right)$ time slots, where $r_k, \ldots, r_1$ are the roots of the top-subtrees sorted in descending order of the total number of packets generated in it, k is the total number of top-subtrees, and N' is the total number of packets in the whole network.

A similar result utilizing multiple channels that minimize the schedule length for raw-data convergecast in WirelessHART networks [65] is obtained by Zhang et al. [61]. One significant difference between WirelessHART networks and sensor networks is that the former performs channel hopping on a per-packet basis, while most existing TDMA convergecast schemes do not support this feature. Thus, parallel transmissions scheduled in the same time slot must use different channels, whereas most of the existing TDMA-based multi-channel protocols first statically assign channels to eliminate potential interference and then perform time slot scheduling. Like in [37] and [66], they also consider buffer requirements at each node and show that when nodes can store at most one packet, the minimum schedule length for line topologies is $2N - 1$ using at most $\lceil N/2 \rceil$ channels (see Fig. 14.7a and b). However, when the nodes can buffer multiple packets, the optimal convergecast time remains the same while the number of channels required can be reduced to $\left\lceil N - \sqrt{N(N-1)/2} \right\rceil$ (see Fig. 14.7c). The basic idea of their proposed approach, which is similar to [37] and [66], is to schedule as many transmissions as possible in each time slot in order to maximize the use of available channels and to make sure that a node which does not have a packet at the beginning of a time slot receives one packet at the beginning of the next time slot.

Zhang et al. also study the effects of packet copying between the microcontroller and the radio transceiver [75] and propose a novel method that separates packet copying from packet transmission in order to improve the schedule length for WirelessHART networks. They show that packet copying could create a bottleneck in the critical path of packet forwarding and presented in [54] a scheme called *conditional immediate transmission* (CIT) which nearly tenfolds multi-hop 802.15.4 throughput.

The basic idea of CIT stems from the following observation. After receiving a packet, the microcontroller triggers an interrupt and notifies a process to fetch the incoming data over the serial peripherals interface (SPI) bus. Before transmitting a packet, the microcontroller first copies the packet into the radio's transmit buffer over the SPI bus and then sends a separate command to the radio to start transmission (see Fig. 14.8a). CIT removes this packet copying off the critical path by copying packet $n - 1$ into the transmit buffer before packet n arrives. When packet

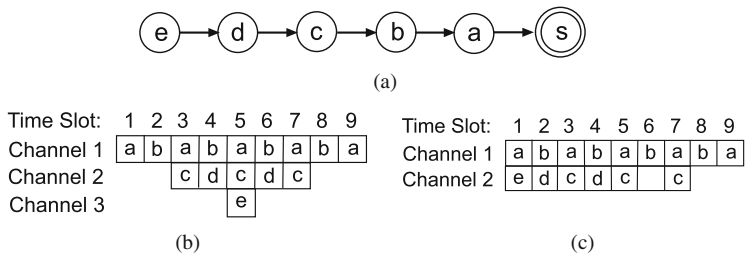


Fig. 14.7 (**a**) A line network with five source nodes. (**b**) Schedule length is 9 using three channels when nodes can buffer at most one packet. (**c**) Schedule length is 9 using two channels when nodes can buffer multiple packets

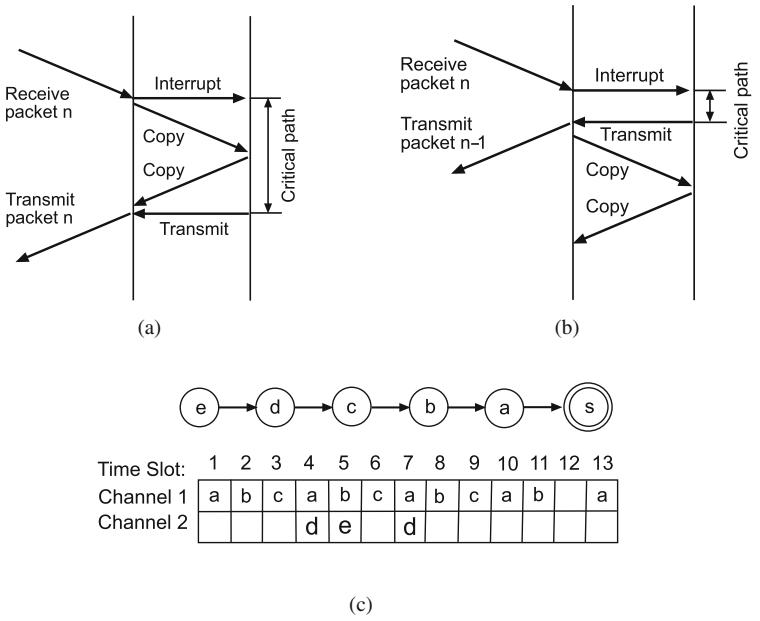


Fig. 14.8 (**a**) Packet copying is on the critical path. (**b**) Packet copying is removed from the critical path. (**c**) Optimal CIT-based convergecast schedule for a line network of five source nodes

n arrives, packet $n-1$ can be immediately forwarded without any copying (see Fig. 14.8b), and so the channel can be released immediately after transmitting/receiving and can be allocated to another node in the next time slot, thereby improving channel utilization. The authors show that although the number of time slots needed increases, the length of each slot is significantly reduced, enabling convergecast with higher throughput.

For line networks with N source nodes each with one packet to send, they show that the lower bound to complete a CIT-based convergecast in the presence of at most $\lceil N/3 \rceil$ channels is $3N-2$. They also propose an algorithm that achieves this lower bound, requiring no node to buffer more than one packet at any time slot. The basic idea, as illustrated with a five-node network in Fig. 14.8c, is to schedule transmission for the immediate child of the sink at every $t = 3m+1$ time slots, for $m = 0, \ldots, N-1$ and for every other node, which still has packets to forward from its subtree, at slot t if its parent was scheduled at slot $t-1$. This idea is extended to tree networks where the root of a top-subtree with the maximum number of remaining packets is scheduled every three time slots. Within each top-subtree a node is scheduled at slot t if its parent was scheduled at slot $t-1$, and the subtree rooted at this node has the maximum number of remaining packets among all the subtrees of the top-subtree. The algorithm requires $\max\{3n_k + \beta, N\}$ time slots, which is optimal to complete CIT-based convergecast on a tree, requiring number of channels at most equal to the depth of the tree. Here $\beta = -1$ if $n_k = n_{k-1}$, and

$\beta = -2$ otherwise, and n_k, as in [37] and [66], is the maximum number of nodes in any subtree sorted in decreasing order of sizes $n_k \geq n_{k-1} \geq \cdots \geq n_1$.

14.3.1.2 Aggregated Data Convergecast

Unlike raw-data convergecast where the application requires every single packet generated by the nodes to be delivered to the sink, periodic data collection often requires delivery of only summarized information in the form of aggregated packets. In general, such aggregated convergecast requires less number of time slots than raw-data convergecast because of the reduced volume of traffic en route to the sink. Under this setting, it is assumed that every node generates a single packet at the beginning of every frame and perfect data aggregation is possible, i.e., each node is capable of aggregating all the packets received from its children as well as that generated by itself into a single packet before transmitting to its parent. This means that the size of aggregated data is constant and does not depend on the actual raw sensor readings. Typical examples of such aggregation functions are MIN, MAX, MEDIAN, COUNT, AVERAGE, etc., which are known as *algebraic* and *distributive* functions [49].

Since the goal is to minimize the schedule length, each parent node ideally should wait to receive all data from its children and then aggregate those with its own data before transmitting. Thus, in aggregated convergecast, a node transmits only once per frame and it maintains an intrinsic order of transmission with respect to its children. When the routing tree is not specified as part of the application requirements, the algorithms in this category also construct the routing tree suitable for aggregation and then perform scheduling. Two of the studies that we discuss below also consider multiple frequencies to eliminate interference. In the following, we discuss several of these algorithms that address the aggregated convergecast problem and its variants.

One of the early works is by Chen et al. [8], where the problem is slightly generalized by considering only a subset $S \subseteq V$ of nodes generating data instead of all the nodes. Assuming uniform transmission range and a unit disk graph (UDG) model, they formulate it as a *minimum data aggregation time* (MDAT) problem where the goal is to find a collision-free schedule that routes data from the subset of nodes to the sink in the minimum possible time. They prove that MDAT is NP-complete, even when restricted to UDGs, by reducing it from the restricted planar 3-SAT problem and design a centralized $(\Delta - 1)$-approximation algorithm, where $\Delta + 1$ is the maximum number of nodes within the transmission range of any node. Their proposed approach does not assume that the routing tree is known a priori; instead, the algorithm finds the data aggregation tree after the schedule is made. If the height of the routing tree is h, then a trivial lower bound on the schedule length is $\max\{h, \log_2 |S|\}$.

The basic idea of the algorithm, called the *shortest data aggregation* (SDA), is to incrementally construct smaller and smaller shortest path trees (SPT) rooted at the sink that span nodes possessing all the data, i.e., in the current iteration, the SPT rooted at the sink spans a set of nodes that possess all data aggregated from S till

the previous iteration. The current iteration produces a collision-free schedule that comprises a set of simultaneously transmitting senders, which are selected from the leaves of the SPT based on the number of non-leaf neighbors in the graph, and a set of corresponding receivers.

Malhotra et al. [3] consider the joint routing and scheduling problem for aggregated convergecast with the goal to construct an optimal routing tree that will help minimizing the schedule length. The basic idea of the tree construction algorithm is to create a shortest path tree and balance the number of children per node so that more parallel transmissions can take place without any single node causing bottleneck. The authors show that for a given routing tree, a lower bound on the schedule length is $\max_{i \in V}\{\xi_i + h_i\}$, where ξ_i and h_i are the number of children and hop distance from the sink, respectively, for node i. To balance the number of children per node, an optimal *semi-matching* formulation on bipartite graphs, originally due to Harvey et al. [35], is used where the goal is to assign nodes from level $h+1$ to the parents at level h such that every parent has an equal number of children. Once the balanced tree is constructed, a ranking-based heuristic is used for scheduling, where the idea is to rank all eligible nodes in decreasing order of their weights which are taken as the number of *non-leaf neighbors*. A higher weight gives a higher relative priority to a node to be scheduled in the current slot over other eligible nodes.

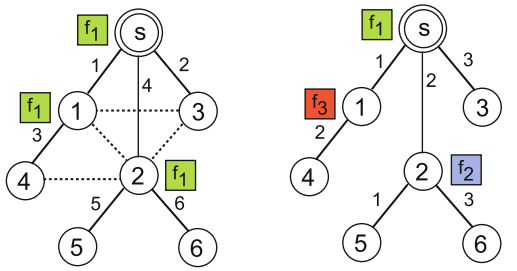
A variation of the aggregated convergecast problem where nodes can adjust their transmission ranges is studied by Shang et al. [63]. They propose an approximation algorithm that has a constant factor guarantee on the optimal schedule length for unit disk graphs. It first constructs a BFS tree rooted at the sink and then constructs a *maximal independent set* using the greedy First-Fit algorithm by choosing nodes in order of their increasing hop distances from the sink in the BFS tree. Note that this results in a *dominating set* which contains the sink but is not a connected set. Then, a minimal number of connector nodes are added to construct a *connected dominating set*, V_{CDS}, and the transmission ranges of all the nodes in this set are set to 1. Next, the scheduling phase runs in two stages. In the first stage, nodes in $V \setminus V_{\text{CDS}}$ are scheduled first so that all their data reach the nodes in V_{CDS}. In the second stage, data is sent from the nodes in V_{CDS} to the sink. It is shown that the first stage takes $15 \log_2 |V \setminus V_{\text{CDS}}|$ time slots, whereas the second stage takes $16d(T_{\text{BFS}}) - 12$ time slots, where $d(T_{\text{BFS}})$ is the depth of the BFS tree. Combining these two, it is shown that the schedule length is at most 31 times the optimal.

Zhang et al. extend their work which was focused on minimal time convergecast scheduling for raw-data convergecast [26] to aggregated convergecast scheduling in [77]. When the size of data is much smaller than the size of the data frame, nodes aggregate the received packets instead of sending single packets. They use the same scheduling algorithm proposed in [26, 27]. They show that using their scheduling algorithm with packet aggregation requires at most $N+2$ time slots in a linear network with N nodes. According to the algorithm, the sink receives the first packet in the first time slot. Then in every three time slots it receives an aggregated packet which contains data from three original unaggregated packets, due to two-hop scheduling to prevent interference. Hence, the total number of required time

slots is $1 + 3\left\lceil \frac{N-1}{3} \right\rceil \leq N + 2$. For multi-line networks, the algorithm achieves a schedule length of $\max\left(n_k + 3\left\lceil \frac{N+2k}{3} \right\rceil\right)$, where k is the number of branches and n_k is the maximum number of nodes in a branch. Finally, for tree networks aggregation-enabled convergecast requires $\max\left(\hat{n}, \left\lceil \frac{N+2L+2(L-k)}{3} \right\rceil\right)$, where L is the number of leaf nodes, k is the number of one-hop subtrees, $\hat{n} = \max_i(n_i + 4l_i - 2)$, n_i is the number of nodes, and l_i is the number of leaf nodes in the ith one-hop subtree.

A variant of the aggregated convergecast problem where a parent node need not wait to receive all data from its children within a single frame before transmitting is investigated by Incel et al. [38] and Ghosh et al. [30]. This is particularly applicable for continuous and periodic monitoring applications that sustain over long durations of time. As explained in the following, the transmission ordering constraint between a parent node and its children within a single frame disappears once a *pipeline* is established, after which the sink starts receiving aggregated data from all the nodes in the network once every frame. In [38], the problem is studied through extensive simulations and experiments, whereas its theoretical aspects are discussed in [30]. In both works, multiple frequency channels are considered as means to eliminate interference. In the following, we first explain the notion of schedule length and pipelining in this variant of aggregated convergecast.

In Fig. 14.9, we show a network of six source nodes where the solid lines represent tree edges and the dotted lines represent interfering links. The numbers beside the links represent the time slots at which the links are scheduled to transmit. The



(a) (b)

	Frame 1						Frame 2					
	S1	S2	S3	S4	S5	S6	S1	S2	S3	S4	S5	S6
s	1	2	3	-	-	-	{ 1,4 }	{ 2,5,6 }	3	-	-	-
1	-	-	-	4	-	-	-	-	-	4	-	-
2	-	-	-	-	5	6	-	-	-	-	5	6

(c)

Fig. 14.9 Aggregated convergecast: (**a**) Schedule length of 6 for single frequency. (**b**) Schedule length of 3 when multiple frequencies are used to eliminate interference. (**c**) Node IDs from which aggregated data is received by their corresponding parents in each time slot over different frames

frequencies assigned to the receivers of the tree are shown in boxes. The entries in the table list the nodes from which packets are received by their corresponding receivers in each time slot for Fig. 14.9a. We note that at the end of frame 1, the sink does not have packets from nodes 5 and 6; however, as the same schedule is repeated, it receives aggregated packets from nodes 2, 5, and 6 in slot 2 of the next frame. Similarly, the sink also receives aggregated packets from nodes 1 and 4 starting from slot 1 of frame 2. The entries {1, 4} and {2, 5, 6} in the table represent single packets comprising aggregated data from nodes 1 and 4 and from nodes 2, 5, and 6, respectively. Thus, a pipeline is established from frame 2, and the sink continues to receive aggregated packets from *all* the nodes once every six time slots. Thus, the minimum schedule length is 6. However, if all nodes are assigned different frequencies, as shown in Fig. 14.9b, then the minimum schedule length turns out to be 3.

The authors in [38] explore a number of different techniques that provide a hierarchy of successive improvements, the simplest among which is an interference-aware, minimum-length TDMA scheduling that enables spatial reuse. To achieve further improvement, they combine transmission power control with scheduling and use multiple frequency channels to enable more concurrent transmissions. The multiple frequencies are assumed to be orthogonal, and a *receiver-based channel assignment* (RBCA) scheme is proposed where the receivers (i.e., parents) in the tree are statically assigned different frequencies to eliminate interference. It is shown through extensive simulations that once multiple frequencies are used along with spatial-reuse TDMA, the data collection rate often no longer remains limited by interference, but by the topology of the network. Thus, in the final step, *degree-constrained trees* are constructed that further enhances the data collection rate.

Ghosh et al. in [30] prove that minimizing the schedule length under multiple frequencies is NP-hard on general graphs and propose approximation algorithms with worst-case provable performance guarantees for geometric networks. In particular, they design a constant factor approximation algorithm for *unit disk graphs* where every node has a uniform transmission range and a $O(\Delta(T)\log n)$ approximation for general disk graphs where nodes have different transmission ranges, where $\Delta(T)$ is the maximum node degree in the routing tree. They also show that a constant factor approximation is still achievable when the routing topology is not known a priori so long as the maximum node degree in the tree is bounded by a constant. Among other theoretical results, Yu et al.[72] proposed a greedy distributed aggregation scheme that takes at most $24D + 6\Delta + 16$ slots, where D is the network diameter and Δ is the maximum node degree.

14.3.2 Algorithms on Minimizing Latency

Minimizing the schedule length to complete convergecast certainly contributes to minimizing latency in data collection; however, in certain cases, it does not guarantee minimizing the average latency for individual packets. For instance, in aggregated data collection, where each sensor node is scheduled once per frame, and

instead of relaying individual packets, they aggregate packets before forwarding toward the sink node, the minimal schedule length is equal to the maximum degree of the routing tree, as shown in [38] and discussed in Sect. 14.3.1. If causality is important, such that a node needs to wait for data from its children before being scheduled (i.e., when a pipeline, as discussed in Sect. 14.3.1, cannot be established), data collection cannot be completed in a period equal to this minimal schedule length. In this section, we survey algorithms that identify minimizing latency as their primary objective.

In [13], Cui et al. focus on minimizing the latency and analyzing the energy latency trade-off for data collection in WSNs where each node generates the same number of packets within each frame of length T. Data is transmitted to a relay node which forwards it toward the sink node; relay nodes are assumed to be not generating data. The authors first present sufficient conditions on link scheduling in order to achieve the minimum worst-case latency T and then present a link scheduling algorithm satisfying these conditions. They propose and prove that it is sufficient for every node to schedule its outgoing links *after* its incoming links in order to achieve the minimum possible latency T. The proposed algorithm classifies the links into levels according to their distance in number of hops from the sink, and the schedule is constructed in reverse order of hop distance, as illustrated in Fig. 14.10. A similar study is presented in [16], where it is shown that minimum-length scheduling does not automatically guarantee minimum latency, and a heuristic is proposed to minimize latency by scheduling the incoming links before the outgoing links.

In [56], Pan et al. propose algorithms for quick convergecast in ZigBee tree-based WSNs. The objective is to enable quick convergecast operations with minimum latency and complying with the ZigBee standard. Different from the studies presented in [12, 27], which minimize latency by minimizing the schedule length and assigning slots to the senders, this study considers receiver-based scheduling. This is due to the fixed wake-up/sleep scheduling specified in the ZigBee stack: in each cycle, nodes wake up twice, first to receive packets from their children and

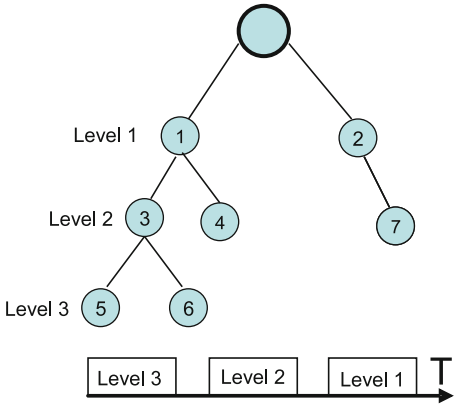


Fig. 14.10 Scheduling to minimize latency

second to transmit to their parents in a ZigBee beacon-enabled tree network. The authors first define a minimum latency beacon scheduling problem for quick convergecast in ZigBee networks and prove it to be NP-complete. Then they propose an algorithm which gives optimal performance for line topologies and within 1.5 times the optimal for ring topologies. The algorithm is also extended for tree-based schemes as a heuristic. A centralized tree-based algorithm traverses the nodes on a tree in a bottom-up manner, starting with the leaf nodes, similar to the previously discussed algorithm [13]. Nodes at the same depth of the tree are sorted according to the interference values (i.e., the number of links that cause interference on the link between the node and its parent) and starting with the most interfered node, and scheduling continues sequentially by assigning the first minimum available slot. Finally, a distributed version of the time slot assignment algorithm is proposed. The Performance of the proposed scheme is evaluated via extensive simulations, and the results are compared with random and greedy scheduling algorithms. Compared to the random and greedy schemes, the proposed heuristics can effectively achieve quicker convergecast. The performance of the heuristics decrease when the number of interference neighbors is high.

Revah et al. in [60] extends the work of Florens [22] by considering minimizing both average delivery time and completion time for convergecast. The authors argue that scheduling strategies that aim to minimize the completion time of a convergecast do not take into account the idle time of messages. For instance, it is unreasonable not to transmit a message toward the destination if it can be transmitted without any delay. Polynomial time solutions are presented for different network topologies: linear, two-branch, and star (or multi-branch) network. The sink node is assumed to have full information about the network topology and computes the schedules on demand. They show that in order to avoid possible delays, an algorithm should start immediate transmission of packets belonging to different groups, such that packets belonging to some group can be transmitted without being delayed by packets from other groups. Although the presented algorithms are centralized, they provide lower bounds for the problem. The nodes are assumed to be equipped with directional radios with two separate control channels for upstream and downstream communication. The protocol interference model is used in the analysis. Evaluations of the algorithms over tree networks are missing in the paper.

Another variant of scheduling is considered by Lu et al. in [46] where joint scheduling and routing with minimum latency requirement is studied. They define the *minimum latency joint scheduling and routing* (MLSR) problem as follows: Given a graph $G = (V, E)$, number of slots K, and flows M, the goal is to find paths P and a slot assignment f such that it maximizes the number of flows with minimum average latency. The problem is solved using a graph coloring approach: First a delay graph is constructed and it is shown that the minimum-weight M node-disjoint paths in the delay graph can be mapped to a solution of MLSR. In the next step, their proposal iteratively finds M node-disjoint paths.

In [40, 43, 47], the performance of different sleep scheduling techniques to minimize latency is evaluated. Although the focus is on sleep scheduling where nodes stay in low-power or sleep modes for most of the time, periodically waking up to

check for activity, the proposed algorithms define the transmission rights for nodes similar to TDMA scheduling. Fully synchronized pattern, shifted even–odd pattern, ladder pattern, two-ladder pattern, multi-parent pattern, and multi-clustering pattern are explored, and their latency behaviors are analyzed in terms of minimum, maximum, and average latency.

14.3.3 Algorithms with Other Objectives

Besides the well-studied objectives of minimizing the schedule length and latency, which are the main focus of this chapter, there also exist studies that focus on other criteria. In this section, we briefly survey some of these studies with different objectives, such as minimizing energy and transmission power, maximizing capacity, maximizing fairness, and meeting deadlines.

14.3.3.1 Algorithms on Minimizing Energy

Energy efficiency is the biggest challenge in designing long-living sensor networks. Since radio communication consumes a lot of energy, a common method is to operate the radio with duty cycling that periodically switches the radio between sleep and wake-up modes. TDMA-based protocols offer the advantage of permitting nodes to enter into sleep mode during inactive periods, thus achieving low duty cycles and conserving energy. Additionally, TDMA-based medium access efficiently eliminates collisions and prevents overhearing, which are the main sources of energy consumption in wireless communication. Therefore, all the TDMA-based protocols proposed for WSNs have the inherent objective of minimizing energy consumption. Transmission power control is one of the well-studied methods in minimizing energy consumption and alleviating interference in wireless networks. Excessive levels of interference can be eliminated if the signals are transmitted with just enough power instead of maximum power.

In [39], Kalpakis et al. consider the maximum lifetime data-gathering problem, with and without aggregation, and propose polynomial time algorithms for maximizing the network lifetime, which is defined as the time until the first sensor runs out of energy. They formulate the problem for the aggregation case as a network flow problem using an ILP and propose an iterative algorithm called, *maximum lifetime data aggregation* (MLDA), to find a maximum lifetime schedule. The case without aggregation, called the *maximum lifetime data routing* (MLDR) problem, is formulated as a maximum flow problem with energy budgets on the nodes and again solved using an ILP.

In [51], a TDMA scheduling scheme for many-to-one communication is studied. TDMA-based communication provides a common energy-saving advantage by allowing nodes to turn their radio off when not engaged in communication; however, too much state transitions between the active and the sleep modes can waste energy. Accordingly, the desired objectives in this paper are to minimize the total time for data collection as well as to minimize the energy consumed on switching between

the active and sleep states. To solve this optimization problem, two population-based stochastic optimization techniques, *particle swarm optimization* and *genetic algorithm*, are hybridized. The former guarantees that there is no empty slot during scheduling, and the latter ensures a strong searching ability to find the optimal slot allocation. It is shown by simulations that the hybrid algorithm outperforms the particle swarm optimization algorithm and the coloring methods in terms of the energy efficiency and finding minimal schedule lengths.

In [18], ElBatt et al. study the problem of joint scheduling and power control. Although the ideas presented in this paper are not directly targeted for WSNs, the problem of joint power control and TDMA scheduling also arises in WSNs, and the solution presented in the paper has been used for minimizing the data collection time in [38]. The algorithm proposed in [18] is a cross-layer method for joint scheduling and power control to improve the throughput capacity. The goal is to find a TDMA schedule that can support as many transmissions as possible in every time slot. It has two phases: (i) scheduling and (ii) power control. The scheduling phase searches for a *valid transmission schedule* where no node is to transmit and receive simultaneously or to receive from multiple nodes simultaneously. The power control phase then iteratively searches for an *admissible schedule* with power levels chosen to satisfy all the interfering constraints in the given valid schedule. In each iteration, the scheduler adjusts the power levels depending on the current RSSI at the receiver and the SINR threshold according to the iterative rule: $P_{\text{new}} = \frac{\beta}{\text{SINR}} \cdot P_{\text{current}}$, which is the well-known power control algorithm by Foschini and Miljanic [25]. If the maximum number of iterations is reached and there are nodes which cannot meet the interfering constraints, the scheduling phase excludes the link with minimum SINR. The power control phase is then repeated until an admissible transmission scenario is found.

The problem of joint scheduling and transmission power control is studied by Moscibroda [52], which will be surveyed in the next section for constant and uniform traffic demands in WSNs. In this paper, it is shown that unbounded improvements in the asymptotic capacity of data collection can be achieved by employing nonlinear power assignment at nodes.

14.3.3.2 Algorithms on Maximizing Capacity

Although maximizing the throughput capacity is not considered to be one of the prioritized objectives in traditional low-rate data collection applications, it may become an important concern in dense deployment or during certain periods when large bursts of packets are generated, for instance, due to a change in the monitored conditions. Moreover, with the adoption of WSNs in different application areas, such as industrial monitoring, health care, and surveillance where large amounts of data need to be collected and sometimes streamed, maximizing the capacity with limited resources has become a popular topic [17]. In this section, we describe two data collection studies that focus on this objective.

In [17], Duarte et al. present ideas which, although not based on TDMA scheduling, are very closely related to the problem of many-to-one data gathering. The

trivial upper bound per node is presented as W/N (W is the transmission capacity and N is the number of nodes), which can be achieved when the sink is 100% busy in receiving. They show circumstances under which this bound is achievable, such as when all the sources can directly transmit to the sink node. On the other hand, if each source cannot directly reach the sink node and the communication takes place in multiple hops, it may or may not be possible to achieve the upper bound depending on the transmission and interference ranges. The bounds presented in the capacity domain can be translated into bounds on schedule length in TDMA. The trivial upper bound for the minimal schedule length is N, which similarly can be achieved when all the sources can directly transmit to the sink node.

The worst-case capacity of wireless sensor networks is studied by Moscibroda in [52], where it is theoretically shown that nonlinear power control mechanisms (without discrete power levels) can significantly help in minimizing the scheduling complexity and also in improving the capacity of WSNs. In this work, the aggregated data capacity and the notion of worst-case capacity, which concerns the question of how much information can each node transmit to the sink, regardless of the network's topology, are investigated for typical *worst-case* structures, such as chains. They prove that the achievable rate in the protocol model is $\theta(1/N)$, whereas it is $\Omega(1/\log_2 N)$ in the physical SINR model using transmission power control techniques (N is the number of nodes). For instance, in Fig. 14.11, two time slots are required to schedule both transmissions in the protocol model, but in the physical model both transmissions can be scheduled simultaneously by transmitting at appropriate power levels. Hence, the most important conclusion of this study is that there is an exponential gap between these two interference models in terms of achievable data rates. However, the assumption that the transceivers can set their power level to any value without considering discrete power levels and limits on the transmission power is somewhat limiting.

Motivated by Moscibroda's findings on using SINR-based interference model for computing capacity, Chafekar et al. in [6] develop polynomial time algorithms to provably approximate the total throughput. They define a throughput maximization problem with SINR constraints: Given a set of nodes V, a set of source–destination pairs, and a transmission power level at each edge, the problem is to (a) choose routes for the pairs, (b) choose flow rates on the routes, and (c) schedule packets at each time slot respecting SINR constraints for all parallel transmissions, such that the total throughput capacity is maximized. They develop a linear programming formulation to approximate the maximum throughput rate vector for SINR constraints. For the case of uniform power levels, they develop a polynomial time

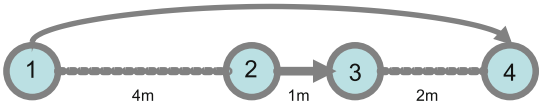


Fig. 14.11 Transmissions from node 1 to node 2 and from node 3 to node 4 can simultaneously take place in the physical interference model, whereas two time slots are needed in the protocol model

approximation algorithm that finds a feasible rate vector with a total throughput of at least $\Omega(r_{\text{opt}}/\log \Delta)$, where r_{opt} is the maximum possible throughput for an instance and Δ is defined as $\max_{u,v\in V} d(u,v)/\min_{u',v'\in V,u'\neq v'} d(u',v')$, where u, u', v, v' represent edges. When non-uniform power levels are used, they show that $O(\log \Delta \log \Gamma)$ approximation can be achieved, where Γ is the ratio between maximum and minimum power levels used. Similarly, in [21], approximation algorithms for link scheduling with SINR models are proposed.

14.3.3.3 Algorithms on Maximizing Fairness

In WSNs, fairness issue may arise as a problem under high data rates, for instance, when the data rates are comparable to available channel bandwidths. In this case, traditional randomized access schemes face the problem of unfair data delivery.

In [67], Sridharan et al. study max–min fair collision-free scheduling in WSNs by developing a linear programming formulation. They propose a distributed max–min fair scheduling mechanism suited for continuous traffic on a data-gathering tree and incorporate it with a time slot-based bandwidth allocation scheme to guarantee collision-free traffic. The algorithm for max-min fair resource allocation works in rounds, and at each round source nodes that are not constrained increase their generated data rate by a small incremental value ε. Nodes become constrained when the total bandwidth usage (the combination of incoming data, generated data, and interfering data traffic) at those nodes is within ε of the total bandwidth. Also, all nodes that are in the subtree below a constrained node, and all nodes whose output traffic interferes at a constrained node also become constrained. The algorithm terminates when all nodes on the routing tree become constrained and the rate available to all sources is the allocated rate. The scheme is incorporated with a TDMA-based scheduling algorithm with the goal to provide enough number of time slots for the data originating at each source at each frame. First, the root node allocates the required number of time slots to each of its children. The time slot allocation algorithm then runs in an iterative manner in a BFS order. At each iteration, only one node allocates time slots to its children. Simulations show that it outperforms an overhearing avoidance MAC (similar to S-MAC) and pruned 802.11 (without ACK's) in terms of energy consumption, fairness, and delay.

Another TDMA-based scheduling algorithm, called AI-LMAC, focusing on fairness was introduced by Chatterjea et al. in [7], which is an extension to the schedule-based MAC protocol, LMAC [36]. In the original LMAC protocol, each node on a data collection tree is allocated one time slot at each frame, whereas in AI-LMAC, nodes are assigned multiple time slots according to the traffic flowing over them. This ensures fairness such that the bandwidth allocated to a node corresponds to the traffic it is expected to encounter. Similar to the scheduling scheme proposed in [67], time slot allocation is based on a parent–child relationship over a data-gathering tree. A parent advises a child, i.e., sends a message to every one of its children indicating the ideal number of slots that a particular node should take up under the current traffic conditions. The process starts at the root node and percolates down the branches of the tree toward the leaf nodes.

14.3.3.4 Algorithms on Meeting Deadlines

Collecting data from a WSN within a specific deadline is closely related with the objectives of minimizing the schedule length and latency. However, it puts an additional constraint associated with the maximum latency allowed per message. For instance, in safety and mission-critical applications where sensor nodes are deployed to detect events, such as oil/gas leak or structural damages, the actuators or controllers need to receive data from all the sensors within specific deadlines [10]. Failure to receive data within the deadlines even from a single sensor may lead to unpredictable and catastrophic failures.

Scheduling messages with deadlines in multi-hop, real-time WSNs is first studied by Li et al. in [44]. They focus on the problem of providing timeliness guarantees for multi-hop transmissions in a real-time robotic sensor application where each message has a specific deadline. They show that the problem of meeting message deadlines is NP-hard and propose heuristics with deadline constraints. A central scheduler schedules messages based on their per-hop timeliness constraints and associated routes, transmission ranges, and the location of the nodes. First, the scheduler divides the transmission requests into disjoint sets such that transmissions within each set do not interfere with one another. While constructing these sets, the scheduler considers an order according to the latest transmission time (LST). The transmission with the minimum LST is chosen at each iteration and is assigned to a set where it does not interfere with other parallel transmissions without missing its deadline and without causing other transmissions in the same set to miss their deadlines. Performance of the proposed algorithm is compared with a method based on CSMA-CA where nodes make local scheduling decisions independent of others. Given a set of messages that are queued up at a node, the node schedules the message with the smallest LST. The proposed algorithm outperforms CSMA-CA-based scheduling in terms of the deadline miss ratio, especially when the utilization is high and/or the probability of collisions is high.

14.3.4 Algorithms with Joint Objectives

Besides the TDMA scheduling algorithms focusing on a particular objective for data collection in WSNs, another approach is to consider multiple objectives and address their trade-offs. One of the most studied joint objectives is minimizing latency together with minimizing energy consumption [74].

In [51], Mao et al. propose a TDMA scheduling scheme with the objective of minimizing the total time for completing a convergecast and minimizing the energy consumed on switching the transceiver between the active and the sleep states. The details of this algorithm is presented in Sect. 14.3.3.

Similar to [51], Wang et al. in [71] also focus on packet delay and the energy consumed on node state transitions. They propose a hierarchical solution to solve the multi-objective TDMA scheduling problem. Particle swarm optimization is exploited due to its strong search ability in combinatorial optimization and to reach

multi-objective optimality. However, since there exists a conflict between delay and energy consumption in TDMA scheduling, the objectives cannot be optimized in parallel. For instance, after collecting data from its child nodes, the sensor node should wait to transmit packets to its own parent instead of switching off. As a result, delay performance is sacrificed. To solve this mutually conflicting multi-objective optimization problem, the concept of Pareto optimality was used in the evaluation system. Minimizing delay and energy consumption is presented as a case study in this paper for the evaluation of the proposed multi-objective optimization algorithm.

In [48], Macedo et al. propose a TDMA-based MAC protocol for latency–energy minimization and interference avoidance for alarm-driven, event-based WSN applications, such as surveillance of sensitive areas. The authors emphasize the trade-off between end-to-end delay and energy efficiency. As in the other protocols discussed in Sect. 14.3.2, a cascading time slot assignment is used to minimize the end-to-end transmission delay. With this slot assignment, the algorithm achieves very low duty cycles, since each node should only listen during the slots assigned to its children at the beginning of each frame and switch to sleep mode afterward. Allocation of slots is performed by parent nodes in a localized manner without requiring a central scheduler. The algorithm to assign the slots is based on Request To Assign and Clear to Assign messages exchanged by the parents and their children that are similar to RTS/CTS messages in CSMA-CA protocols. Instead of using a simple hop-based interference model, the algorithm checks the link quality directly experienced by the nodes.

In [68], Trigoni et al. propose wave scheduling and routing in sensor networks for energy efficiency in WSNs by considering data dissemination strategies that avoid collisions and message retransmissions at the cost of higher message latency. They define the energy minimization problem with the goal to determine a data dissemination scheme that minimizes the energy consumed in delivering all messages within a bounded delay. First, they prove that the problem is NP-hard. For the analysis, unit disk graphs are used such that two nodes are connected by an edge if and only if the Euclidean distance between them is at most 1. Moreover, to simplify the problem, they use a partitioning scheme that allows to schedule communication tasks at the cell level, rather than at the node level. They partition the network into square cells, where the length of each cell is set so that a node anywhere in a cell can typically communicate directly with nodes in adjacent cells. In their proposal, the scheduler schedules concurrently activate cells that are sufficiently spaced apart so that the message transmissions within these cells do not interfere with one another. In the wave scheduling algorithm, directed edges of the rectilinear grid that connect two adjacent cells are activated periodically in a sequential manner by avoiding interference at the MAC layer and allowing nodes to turn off their radios whenever they do not need to communicate. The algorithm jointly works with a routing protocol to reduce interference, and extensive simulations show that the proposed scheduling method results in significant energy savings at the expense of increased message latency.

Similar to [68], Oswald et al. present tight bounds for delay-sensitive aggregation in [55] by studying the trade-off between latency and energy cost. In order to

reduce the communication cost, nodes aggregate messages while forwarding toward the sink node. In other words, nodes wait to receive/generate more packets before relaying so as to decrease the communication cost at the expense of late delivery at the sink node. The authors present competitive ratios (a metric, in which the performance of an on-line algorithm is compared to the performance of an optimal off-line algorithm) for a simple algorithm that aims to balance the total latency and energy cost in which nodes aggregate information about multiple events in one message until a forwarding condition is satisfied. The forwarding condition implies that a message should be forwarded to the parent node as soon as the current latency exceeds the transmission cost. They prove that any oblivious algorithm (i.e., given the packet arrival times all nodes react in the same way, independent of its distance to the sink node) achieves a competitive ratio of $O(\min(h, c))$, where h is the the height of the tree and c is the transmission cost per edge, for tree networks and $\Theta\left(\min\left(\sqrt{h}, c\right)\right)$ for chain networks.

In [42], Lee et al. study the scheduling problem with the objective of energy efficiency and reliability. In this work, besides collision-free scheduling, network flow optimization techniques and optimized tree topologies are used to achieve energy efficiency. Moreover, reliability is guaranteed by including many retransmission opportunities in the schedules. The protocol proposed gets the connectivity graph as the input and first assigns layers to the nodes to create a hierarchy in the network. The sink node is assigned to layer 0 and the nodes connected to the sink are assigned to layer 1. By using a BFS order, all nodes in the graph are assigned layers. After the layer assignment, a parent graph is created by removing the links among the nodes at the same layer. By using the parent graph, all the potential parent sets are created for the nodes. Based on the parent graph, optimal flow rates are calculated that minimize the energy consumption and accordingly usage percentages of links are calculated. Based on these percentages, the algorithm constructs a set of trees to be used in each data collection cycle. Finally, for each tree a collision-free schedule list is created by the sink node and the schedule is broadcast back to the nodes in the network.

14.3.5 Taxonomy

In this section, we provide a classification and a summary of the surveyed TDMA-based scheduling algorithms for data collection in WSNs. The classification is based on the design objectives, model constraints, and assumptions that are presented in Sect. 14.2. Table 14.1 summarizes the general aspects of the algorithms that are discussed in Sect. 14.3, ordered by the design objectives.

Most of the algorithms focus on the objective of minimizing the schedule length using an interference model that is a variant of the graph-based protocol model. Both aggregated and raw-data collection operations are investigated in the surveyed solutions. Most of the algorithms consider data collection operation over tree topologies, whereas some of them focus on specific topologies, such as chain, ring, or

Table 14.1 Taxonomy of TDMA-based data collection algorithms in WSNs

Reference	objective	Interference model	Data collection method	Topology	Hardware assumption	Implementation	Assignment scheme
[22–24]	Minimize schedule length	Protocol	Raw	Line, multi-line, tree	–	Centralized	Links
[20]	Minimize schedule length	Graph	Raw	Tree	–	Centralized	Links, nodes
[41]	Minimize schedule length	Interference range	Raw	Tree, disjoint strips	–	Centralized	Links
[12]	Minimize schedule length	2-hop	Raw	Line, tree	–	Centralized	Links
[27]	Minimize schedule length	2-hop	Raw	Line, multi-line, tree	–	Distributed	Links
[69]	Minimize schedule length	2-hop	Raw	Tree, general topology	–	Centralized	Links
[1]	Minimize schedule length	Graph	Raw	Tree	CDMA	Centralized	Links
[37]	Minimize schedule length	Protocol and Physical	Raw and Aggregated	Tree	Power control and multi-channels	Centralized	Links
[66]	Minimize schedule length	2-hop	Raw	Tree	Multi-channels	Centralized, distributed	Links
[61, 75]	Minimize schedule length	Measurements	Raw	Tree	Channel hopping	Distributed	Links
[8]	Minimize schedule length	Unit disk graph	Aggregated	Tree	–	Centralized	Links
[3]	Minimize schedule length	Graph	Aggregated	Tree	–	Centralized	Links
[63]	Minimize schedule length	–	Aggregated	Tree	Power control	Centralized	Links
[77]	Minimize schedule length	Interference range	Aggregated	Line, multi-line, tree	–	Distributed	Links

Table 14.1 (continued)

[30, 38]	Minimize schedule length	Protocol and physical	Aggregated	Tree	Power control and multi-channels	Centralized	Links
[13]	Minimize delay	–	Aggregated	Tree, general topology	–	Centralized	Links
[16]	Minimize delay	2-hop	Aggregated	Chain, general	–	Centralized	Links
[56]	Minimize delay	2-hop	Aggregated	Tree	–	Centralized, distributed	Links
[60]	Minimize delay	Protocol	–	Line, star, multi-line	Directional antennas	Centralized	Links
[46]	Minimize delay	Interference range	–	Tree	FDMA	Centralized, distributed	Links
[51]	Minimize total time and energy	2-hop	–	Tree	–	Centralized	Links
[71]	Minimize delay and energy	2-hop	–	General graph	–	Centralized	Links
[48]	Minimize latency and energy	Link quality — SINR	–	General	–	Distributed	Links
[68]	Trades off latency for energy	Cell based	–	General	–	Centralized	Links
[39]	Maximizing Lifetime	–	Raw and Aggregated	General graph	–	Centralized	Links
[55]	Minimize delay and energy	–	Aggregated	Chain, tree	–	Distributed	Links
[42]	Minimize delay and energy	Interference range	Aggregated	Dynamic trees	–	Centralized	Links
[17]	Maximize capacity	Protocol	Aggregated	Flat, clustered	–	Centralized, distributed	–
[52]	Maximize capacity, minimize power	Physical model	Aggregated	Line (worst case)	Power control	Centralized	Links
[6]	Maximize capacity, minimize power	Physical model	Aggregated	General	Power control	Centralized	Links
[67]	Maximize fairness	Interference range	Raw	Tree, general	–	Distributed	Links
[44]	Meeting message deadlines	Interference range	–	Star, tree	–	Centralized	Links

star. All of the algorithms require time synchronization to operate under a TDMA schedule, but none of them mention the granularity of the synchronization or the kind of synchronization schemes used. It is assumed that nodes are synchronized by a technique available in the literature [64]. Most of the algorithms work with simple transceivers available on the sensor nodes, while some only work with transceivers with a special capability, such as directional antennas. Additionally, some of the solutions explore the benefits of using transceivers that can adjust its transmission power level and/or operating frequency (these types of radios are commonly available on widely used sensor mote platforms [4, 62]), besides the use of simple radios. Most of the algorithms aim to find lower bounds or compute approximation algorithms for the studied NP-complete problems. Therefore, the algorithms provide centralized solutions which are usually computed by the sink node. Some of the algorithms present scheduling solutions that are coupled with additional methods, such as multiple frequencies, transmission power control, and CDMA to eliminate interference. The surveyed algorithms usually support all patterns of convergecast, such as one-shot or continuous data collection, while some are optimized for a specific one. Lastly, almost all the algorithms consider scheduling at link level rather than node level.

14.4 Future Research Directions/Open Problems

As we have seen, extensive research has been done with many different objectives in the field of TDMA scheduling for data gathering in WSNs. However, there still exist some open questions to be addressed, especially related to real implementation and evaluation of the proposals on testbeds or on real deployments. There also exist several theoretical questions that need to be addressed. In this section, we briefly summarize these open problems.

Most of the solutions are theoretical or simulation based (except [75]) offering only centralized solutions. Real implementation on sensor nodes where schedules are computed locally and are adaptive to network dynamics is necessary to enhance the operation of WSNs and to meet application requirements. For instance, we observe a trend in using WSNs to support more complex operations ranging from industrial control to health care, which require complex operations like detection of events in real time or responsive querying of the network by collecting streams of data in a timely manner. Thus, supporting QoS (quality of service) metrics such as delay and reliability become more important [31]. Therefore, distributed implementation and performance testing of the proposed algorithms on testbed or real deployments become essential. Additionally, real implementation and deployment will help in addressing the problems of intermittent connectivity and channel errors with unreliable links and handling asymmetric links. We should note that many of the existing algorithms provide lower bounds for the studied problems, and so they can be used as benchmarks for comparing the performance of the distributed solutions.

Although there exist some works that address multiple joint objectives, more detailed investigations to address the trade-offs between conflicting objectives will be beneficial. Most of the studies consider the trade-offs between energy efficiency and latency objectives. Different trade-offs can be identified between other objectives, such as minimizing latency and maximizing reliability or maximizing capacity and minimizing energy consumption. For instance, with the extension of WSNs in the visual domain where embedded cameras act as sensors, criteria such as reliability, QoS, and timeliness of the streamed data are becoming important. Solutions to address different objectives and trade-offs, for instance, consumed energy versus reconstructed image quality, should be explored within the perspective of data collection in WSNs.

Most of the surveyed algorithms consider fixed traffic patterns, i.e., every node generates a fixed number of packets in each data collection cycle. In a real scenario, some nodes may have a lot of packets that require more than one time slot per frame, while some others may not have any data to send in a time slot, thus wasting bandwidth. It will be interesting to explore the performance in such scenarios with random packet arrivals and combining the solutions of TDMA scheduling with rate allocation algorithms, especially in applications where high data rates are necessary.

In the surveyed papers, either raw-data collection or aggregated data collection are considered. Another possibility is to investigate different levels of aggregation, i.e., how much of the data received from the children is forwarded to the parent node. Investigating different levels of aggregation was proposed in [70] where the efficiency of different tree construction mechanisms was analyzed in terms of latency and energy metrics. This study can be extended for TDMA-based data collection algorithms in WSNs.

Some of the surveyed algorithms provide cross-layer solutions, where the schedules are computed together with methods such as transmission power control, optimal routing trees, and multi-frequency scheduling. It is indeed essential to address the problems from a cross-layer perspective to achieve the target functions and offer better performances. Along this line of research, Chafekar et al. in [5] extend the work by Moscibroda [53] in designing cross-layer protocols using the SINR model and proposed polynomial time algorithms with provable worst-case performance guarantee for the latency minimization problem. Their cross-layer approach chooses power level for all transceivers, routes for all connections, and constructs an end-to-end schedule such that SINR constraints are satisfied. A prominent research direction is to consider such cross-layer approaches from a theoretical point of view. More research can be done in this direction to combine the existing work with the solutions at different layers. In most studies, static topologies are assumed. Problems related to dynamic topologies, such as topological changes and addition of new nodes, are open. In addition, the time complexity of data gathering under various hypothesis [28], such as when some nodes have no packet to transmit, or when no buffering is allowed, remains open.

As was pointed by Moscibroda in [52], the type of interference model may heavily impact the achievable results. Use of realistic models for communication and interference is another direction that can be further investigated. Along this

direction, Goussevskaia et al. [32] present the first NP-completeness proofs (by reducing from the partition problem) on two scheduling problems using the SINR interference model. The first problem consists in finding a minimum-length schedule for a given set of links. The second problem receives a weighted set of links as input and consists in finding a maximum-weight subset of links to be scheduled simultaneously in one shot. In [53], Moscibroda et al. study a generalized version of the SINR interference model and obtain theoretical upper bounds on the scheduling complexity of arbitrary topologies. They prove that if signals are transmitted with correctly assigned transmission power levels, the number of time slots required to successfully schedule all links in an arbitrary topology is proportional to the squared logarithm of the number of nodes times a previously defined static interference measure. More of such works that bridge the gap between static graph-based interference models and the physical models are needed.

14.5 Conclusions

In this chapter, we have surveyed TDMA-based scheduling algorithms for data collection in wireless sensor networks. We classified the algorithms according to their design objectives and constraints and provided a survey of existing algorithms with comparisons. In terms of the design objectives, most of the surveyed algorithms aim at (i) minimizing schedule length, (ii) minimizing latency, (iii) minimizing energy, and (iv) maximizing fairness, whereas some algorithms aim for joint objectives, such as maximizing capacity and minimizing energy or minimizing delay and energy. The surveyed algorithms also vary according to the design constraints and assumptions. For instance, some of the algorithms use simple models for communication and interference while some of them assume complex transceivers available on the nodes. As it was shown in some of the surveyed papers, the unrealistic models or assumptions may heavily impact the achievable results. Use of realistic models for communication, modification of those solutions that base their assumptions on unrealistic models, implementations on real sensor nodes and performance evaluations over real testbeds and deployments, and addressing the trade-offs between conflicting design objectives are identified as some of the important directions for future research.

References

1. V. Annamalai, S. Gupta, L. Schwiebert. On tree-based convergecasting in wireless sensor networks. In: *WCNC '03*, volume 3, pages 1942–1947, New Orleans, LA, USA, 2003.
2. K. Arisha, M. Youssef, M. Younis. Energy-aware tdma-based mac for sensor networks. System-level power optimization for wireless multimedia communication pages 21–40, 2002.
3. I.N. Baljeet Malhotra, M.A. Nascimento. Aggregation convergecast scheduling in wireless sensor networks. Technical report, University of Alberta, 2009.
4. CC2420: Single-chip 2.4 ghz ieee 802.15.4 compliant and zigbee(tm) ready rf transceiver. http://www.ti.com/lit/gpn/cc2420.

5. D. Chafekar, V.A. Kumar, M. Marathe, S. Parthasarathy, A. Srinivasan. Cross-layer latency minimization in wireless networks with SINR constraints. In: *MobiHoc '07*, ACM, New York, NY, pages 110–119, Montreal, Quebec, Canada, 2007.
6. D. Chafekar, V.S.A. Kumar, M.V. Marathe, 0002. S.Parthasarathy, A. Srinivasan. Approximation algorithms for computing capacity of wireless networks with SINR constraints. In: *INFOCOM*, pages 1166–1174, Phoenix, AZ, USA, 2008.
7. S. Chatterjea, L. van Hoesel, P. Havinga. Ai-lmac: An adaptive, information-centric and lightweight mac protocol for wireless sensor networks. In: *Issnip '04*, Melbourne, Australia, 2004.
8. X. Chen, X. Hu, J. Zhu. Minimum data aggregation time problem in wireless sensor networks. In: *MSN*, pages 133–142, Wuhan, China, 2005.
9. K. Chintalapudi, T. Fu, J. Paek, N. Kothari, S. Rangwala, J. Caffrey, R. Govindan, E. Johnson, S. Masri. Monitoring civil structures with a wireless sensor network. *IEEE Internet Computing*, 10(2): 26–34, 2006.
10. K.K. Chintalapudi, L. Venkatraman. On the design of mac protocols for low-latency hard real-time discrete control applications over 802.15.4 hardware. In: *IPSN '08*, pages 356–367, St. Louis, MO, USA, 2008.
11. I. Chlamtac, S. Kutten. Tree-based broadcasting in multihop radio networks. *IEEE Transactions on Computers* 36(10): 1209–1233 (1987)
12. H. Choi, J. Wang, E. Hughes. Scheduling for information gathering on sensor network. Wireless Networks (Online) (2007)
13. S. Cui, R. Madan, A. Goldsmith, S. Lall. Energy-delay tradeoffs for data collection in tdma-based sensor networks. In: *ICC '05*, volume 5, pages 3278–3284, Seoul, Korea, 2005.
14. M. Dalbro, E. Eikeland, A.J.i. Veld, S. Gjessing, T.S. Lande, H.K. Riis, O. Sør(*à*)sen. Wireless sensor networks for off-shore oil and gas installations. In: *SENSORCOMM '08*, pages 258–263, Cap Esterel, France, 2008.
15. I. Demirkol, C. Ersoy, F. Alagoz. Mac protocols for wireless sensor networks: A survey. *IEEE Communications Magazine* 44(4): 115–121, 2006.
16. P. Djukic, S. Valaee. Link scheduling for minimum delay in spatial re-use tdma. In: *Infocom '07*, pages 28–36, IEEE, Anchorage, Alaska, USA, 2007.
17. E. Duarte-Melo, M. Liu. Data-gathering wireless sensor networks: Organization and capacity. *Computer Networks* 43(4): 519–537, 2003.
18. T. ElBatt, A. Ephremides. Joint scheduling and power control for wireless ad-hoc networks. In: *Infocom '02*, volume 2, pages 976–984, 2002.
19. J. Elson, L. Girod, D. Estrin. Fine-grained network time synchronization using reference broadcasts. *SIGOPS Operator Systems Review*, 36(SI): 147–163, 2002.
20. S. Ergen, P. Varaja. Tdma scheduling algorithms for sensor networks. Technical report, University of California, Berkeley, 2005.
21. S. Fan, L. Zhang, Y. Ren. Approximation algorithms for link scheduling with physical interference model in wireless multi-hop networks. CoRR abs/0910.5215, 2009.
22. C. Florens, M. Franceschetti, R. McEliece. Lower bounds on data collection time in sensory networks. *IEEE Journal on Selected Areas in Communications* 22(6): 1110–1120, 2004.
23. C. Florens, R. McEliece. Scheduling algorithms for wireless ad-hoc sensor networks. In: *Globecom '02*, pages 6–10, IEEE, Taipei, Taiwan, 2002.
24. C. Florens, R. McEliece. Packets distribution algorithms for sensor networks. In: *Infocom '03*, volume 2, pages 1063–1072, IEEE, San Francisco, CA, USA, 2003.
25. G. Foschini, Z. Miljanic. A simple distributed autonomous power control algorithm and its convergence. *IEEE Transactions on Vehicular Technology* 42(4): 641–646, 1993.
26. S. Gandham, Y. Zhang, Q. Huang. Distributed minimal time convergecast scheduling in wireless sensor networks. In: *ICDCS '06*, IEEE Computer Society, Washington, DC, page 50, 2006. DOI http://dx.doi.org/10.1109/ICDCS.2006.30
27. S. Gandham, Y. Zhang, Q. Huang. Distributed time-optimal scheduling for convergecast in wireless sensor networks. *Computer Networks* 52(3): 610–629, 2008.

28. L. Gargano. Time optimal gathering in sensor networks. In: *SIROCCO '07*, pages 7–10, 2007.
29. A. Ghosh. Estimating coverage holes and enhancing coverage in mixed sensor networks. In: *LCN '04*. IEEE Computer Society, Washington, DC, pages 68–76, 2004. DOI http://dx.doi.org/10.1109/LCN.2004.53
30. A. Ghosh, O.D. Incel, V.A. Kumar, B. Krishnamachari. Multi-channel scheduling algorithms for fast aggregated convergecast in sensor networks. In: *MASS '09*, pages 362–372, IEEE, Macau, China.
31. GINSENG: Performance control in wireless sensor networks. www.ict-ginseng.eu
32. O. Goussevskaia, Y.A. Oswald, R. Wattenhofer. Complexity in geometric SINR. In: *MobiHoc '07*, ACM, New York, NY, USA, pages 100–109, 2007. http://doi.acm.org/10.1145/1288107.1288122
33. J. Grönkvist, A. Hansson. Comparison between graph-based and interference-based stdma scheduling. In: *MobiHoc '01*, pages 255–258, ACM, Long Beach, CA, USA, 2001.
34. P. Gupta, P. Kumar. The capacity of wireless networks. *IEEE Transactions on Information Theory* IT-46(2): 388–404, 2000.
35. N.J. Harvey, R.E. Ladner, L. Lovász, T. Tamir. Semi-matchings for bipartite graphs and load balancing. *Journal of Algorithms* 59(1): 53–78, 2006. http://dx.doi.org/10.1016/j.jalgor.2005.01.003
36. L. van Hoesel, P. Havinga. A lightweight medium access protocol (LMAC) for wireless sensor networks. In: *INSS' 04*. SICE (Society of Instrument and Control Engineers), Tokyo, Japan, 2004.
37. D.O. Incel, A. Ghosh, B. Krishnamachari, K. Chintalapudi. Fast data collection in tree-based wireless sensor networks. *IEEE Transactions on Mobile Computing* (*submitted*), 2009.
38. O.D. Incel, B. Krishnamachari. Enhancing the data collection rate of tree-based aggregation in wireless sensor networks. In: *SECON '08*, pages 569–577, IEEE, San Francisco, CA, USA, 2008.
39. K. Kalpakis, K. Dasgupta, P. Namjoshi. Efficient algorithms for maximum lifetime data gathering and aggregation in wireless sensor networks. *Computing Network* 42(6): 697–716, 2003.
40. A. Keshavarzian, H. Lee, L. Venkatraman. Wakeup scheduling in wireless sensor networks. In: *MobiHoc '06*, ACM, New York, NY, pages 322–333, 2006. http://doi.acm.org/10.1145/1132905.1132941
41. N. Lai, C.King, C. Lin. On maximizing the throughput of convergecast in wireless sensor networks. In: *GPC '08*, pages 396–408, Kunming, China, 2008.
42. H. Lee, A. Keshavarzian. Towards energy-optimal and reliable data collection via collision-free scheduling in wireless sensor networks. In: *INFOCOM*, pages 2029–2037, Phoenix, AZ, USA, 2008.
43. H. Lee, A. Keshavarzian, H.K. Aghajan. Multi-cluster multi-parent wake-up scheduling in delay-sensitive wireless sensor networks. In: *GLOBECOM*, pages 430–435, New Orleans, LA, USA, 2008.
44. H. Li, P. Shenoy, K. Ramamritham. Scheduling messages with deadlines in multi-hop real-time sensor networks. In: *RTAS 2005*, pages 415–425, San Francisco, CA, USA, 2005.
45. X.Y. Li, Y. Wang. Simple heuristics and ptass for intersection graphs in wireless ad hoc networks. In: *DIALM '02*, ACM, New York, NY, pages 62–71, 2002. http://doi.acm.org/10.1145/570810.570819
46. G. Lu, B. Krishnamachari. Minimum latency joint scheduling and routing in wireless sensor networks. *Ad Hoc Netw*. 5(6): 832–843, 2007. http://dx.doi.org/10.1016/j.adhoc.2007.03.002
47. G. Lu, N. Sadagopan, B. Krishnamachari, A. Goel. Delay efficient sleep scheduling in wireless sensor networks. In: *INFOCOM '05*, pages 2470–2481, Miami, FL, USA, 2005.
48. M. Macedo, A. Grilo, M. Nunes. Distributed latency-energy minimization and interference avoidance in tdma wireless sensor networks. *Computing Network* 53(5): 569–582, 2009. http://dx.doi.org/10.1016/j.comnet.2008.10.015
49. S. Madden, M. Franklin, J. Hellerstein, W. Hong. Tinydb: An acquisitional query processing system for sensor networks. *ACM Transactions on Database Systems* 30(1): 122–173, 2005.

50. A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, J. Anderson. Wireless sensor networks for habitat monitoring. In: *WSNA '02*, pages 88–97, Atlanta, GA, USA, 2002.
51. J. Mao, Z. Wu, X. Wu. A tdma scheduling scheme for many-to-one communications in wireless sensor networks. *Computer Communications* 30(4): 863–872, 2007.
52. T. Moscibroda. The worst-case capacity of wireless sensor networks. In: *IPSN '07*, pages 1–10, Cambridge, MA, USA, 2007.
53. T. Moscibroda, R. Wattenhofer, A. Zollinger. Topology control meets SINR: The scheduling complexity of arbitrary topologies. In: *MobiHoc '06*, pages 310–321, 2006.
54. F. Osterlind, A. Dunkels. Approaching the maximum 802.15.4 multi-hop throughput. In: *HotEmNets 2008*, Charlottesville, VA, page 6, 2008. `http://eprints.sics.se/3426/01/osterlind08approaching.pdf`
55. Y.A. Oswald, S. Schmid, R. Wattenhofer. Tight bounds for delay-sensitive aggregation. In: *PODC '08*, ACM, New York, NY, pages 195–202, 2008. http://doi.acm.org/10.1145/1400751.1400778
56. M. Pan, Y. Tseng. Quick convergecast in zigbee beacon-enabled tree-based wireless sensor networks. *Computer Communications* 31(5): 999–1011, 2008.
57. C. Papadimitriou. The complexity of the capacitated tree problem. *Networks* 8(3): 217–230, 1978.
58. V. Rajendran, K. Obraczka, J. Garcia-Luna-Aceves. Energy-efficient, collision-free medium access control for wireless sensor networks. In: *SenSys '03*, pages 181–192, 2003.
59. S. Ramanathan, E. Lloyd. Scheduling algorithms for multihop radio networks. *IEEE/ACM Transactions on Networking* 1(2): 166–177, 1993.
60. Y. Revah, M. Segal. Improved lower bounds for data-gathering time in sensor networks. In: ICNS '07, IEEE Computer Society, Washington, DC, page 76, 2007. http://dx.doi.org/10.1109/ICNS.2007.71
61. I. Rhee, A. Warrier, M. Aia, J. Min. Z-mac: A hybrid mac for wireless sensor networks. In: *SenSys '05*, pages 90–101, 2005.
62. Nordic Semi Conductors, nrf905 multiband transceiver. http://www.nordicsemi.com
63. W. Shang, P. Wan, X. Hu. Approximation algorithm for minimal convergecast time problem in wireless sensor networks. Wireless Networks, 2009. 10.1007/s11276-009-0207-9
64. F. Sivrikaya, B. Yener. Time synchronization in sensor networks: A survey. *IEEE Network* 18(4): 45–50, 2004. 10.1109/MNET.2004.1316761
65. J. Song, S. Han, A. Mok, D. Chen, M. Lucas, M. Nixon. Wirelesshart: Applying wireless technology in real-time industrial process control. In: *RTAS '08*, pages 377–386, St. Louis, MO, USA, 2008.
66. W.Z. Song, F. Yuan, R. LaHusen, B. Shirazi. Time-optimum packet scheduling for many-to-one routing in wireless sensor networks. International Journal Parallel Emergent Distributed Systems 22(5): 355–370, 2007. http://dx.doi.org/10.1080/17445760601111459
67. A. Sridharan, B. Krishnamachari. Max-min fair collision-free scheduling for wireless sensor networks. In: *IPCCC '04*, pages 585–590, Austin, TX, USA, 2004.
68. N. Trigoni, Y. Yao, A. Demers, J. Gehrke, R. Rajaraman. Wave scheduling and routing in sensor networks. *ACM Transactions on Sensor Networks* 3(1): 2, 2007.
69. H.W. Tsai, T.S. Chen. Minimal time and conflict-free schedule for convergecast in wireless sensor networks. In: *ICC '08*, pages 2808–2812, 2008.
70. S. Upadhyayula, S. Gupta. Spanning tree based algorithms for low latency and energy efficient data aggregation enhanced convergecast (dac) in wireless sensor networks. *Ad Hoc Networks* 5(5): 626–648, 2007.
71. T. Wang, Z. Wu, J. Mao. A new method for multi-objective tdma scheduling in wireless sensor networks using pareto-based pso and fuzzy comprehensive judgement. In: *HPCC '07*, Springer, Berlin, pages 144–155, 2007.
72. B. Yu, J. Li, Y. Li. Distributed data aggregation scheduling in wireless sensor networks. In: *Infocom '09*, Rio de Janeiro, Brazil, 2009.

73. L. Yu, N. Wang, X. Meng. Real-time forest fire detection with wireless sensor networks. In: *WiCom*, volume 2, pages 1214–1217, 2005.
74. Y. Yu, B. Krishnamachari, V.K. Prasanna. Energy-latency tradeoffs for data gathering in wireless sensor networks. In: *INFOCOM*, Hong Kong, China, 2004.
75. H. Zhang, F. Österlind, P. Soldati, T. Voigt, M. Johansson. Time-optimal convergecast with separated packet copying. Technical report, Royal Institute of Technology (KTH) (2009)
76. H. Zhang, P. Soldati, M. Johansson. Optimal link scheduling and channel assignment for convergecast in linear wirelessHART networks. In: *WiOPT '09*, Seoul, Korea, 2009.
77. Y. Zhang, S. Gandham, Q. Huang. Distributed minimal time convergecast scheduling for small or sparse data sources. In: *RTSS '07, IEEE Computer Society*, Washington, DC, pages 301–310, 2007. http://dx.doi.org/10.1109/RTSS.2007.19