Packet Delay Minimization in Multi-hop Wireless Sensor Networks with Periodic Traffic

Bartłomiej Ostrowski
Institute of Telecommunications
Warsaw University of Technology
Warsaw, Poland
b.ostrowski@tele.pw.edu.pl

Michał Pióro
Institute of Telecommunications
Warsaw University of Technology
Warsaw, Poland
m.pioro@tele.pw.edu.pl

Artur Tomaszewski
Institute of Telecommunications
Warsaw University of Technology
Warsaw, Poland
a.tomaszewski@tele.pw.edu.pl

***Abstract*—The paper is devoted to minimization of end-to-end packet delay in multi-hop TDMA-based wireless sensor networks composed of gateways and wireless routers that form a mesh topology. The network is supposed to deliver packets, generated at the sensors, to the gateways. It is assumed that each sensor (connected to one of the routers) produces a new packet at the beginning of each consecutive TDMA frame, and each such packet is then transmitted to a given subset of gateways along a multicast routing tree rooted at the sensor's router. Assuming that the transmission pattern of the TDMA frame is given (i.e., knowing what nodes are selected to broadcast in each time slot of the frame) and sufficient to carry the so described packet streams with finite delay, we aim at finding a packet broadcast scheduling pattern that minimizes the maximum packet delay over all packet streams generated by the sensors. To achieve that, we introduce an exact mixed-integer programming formulation for the related optimization problem. Since the problem formulation can be effectively solved only for networks of limited size, we propose a relevant heuristic method, and demonstrate, through an extensive numerical study, its capability of finding near optimal solutions in reasonable time.**

***Index Terms*—Wireless sensor networks, multicast, transmission scheduling, mixed-integer programming, simulated annealing, IoT**

I. Introduction

The considerations of the presented paper are motivated by the issue of minimizing the end-to-end delay experienced by the packets that are to be sent from the sensors to the selected gateways in meshed wireless sensor networks (WSN) [1]–[3], that are based on time division multiple access (TDMA) to radio channel. It is known that in terms of throughput maximization, TDMA potentially outperforms other medium access control schemes like CSMA. Since TDMA is supported by medium access control (MAC) schemes such as WiMax, WirelessHART [4], ISA100 [5], and the whole family of low-rate personal area network (LR-WPAN) technologies based on the IEEE 802.15.4 specifications, it is worth considering for WSN supporting IoT.

In the considered network the sensors are connected to routers. At the beginning of each consecutive (repeated) TDMA frame, each sensor produces a new packet (containing measurement data); each such packet is sent over a fixed multicast tree from the sensor's host router to a set of gateways selected for the sensor in question. Each node of the tree broadcasts the received packets to its neighbors, according to a packet broadcast schedule specified for the (time) slots of the frame.

The basic research question addressed in this paper is as follows: how to rigorously model and optimize the packet delay in the above described multi-hop wireless sensor networks of the kind described above for a given frame composition? This question includes the delay objective to be minimize as well as description of the frame composition that is necessary in the modeling.

The main body of this paper is composed of Sections II and III. In the first of them, we describe the considered network and frame configuration, introduce a mixed-integer programming formulation of the basic packet delay optimization problem, and discuss the issues that concern its input data preprocessing, as well as extensions of the problem. Since the problem turns out to be difficult and hence finding its exact solutions is inefficient, in Section III we present a solution algorithm based on the simulated annealing meta-heuristic that is capable of finding near optimal solutions of the studied problem in reasonable time for networks of practical size. The considerations of these two sections are illustrated and justified by a numerical study presented in Section IV, followed by Section V that briefly concludes the paper.

The considerations of the paper are continuation of the research presented in paper [6], in particular of the packet delay minimization introduced in Appendix B2 of that paper.

II. Packet Delay Minimization Problem (PDMP)

In this section we will first introduce the notation and basics facts about the considered network configuration and then described in detail the basic problem dealt with in this paper, that is PDMP – *packet delay minimization problem*.

A. Network description

The considered network configuration is described below, using notation summarized in Table I. The network graph is composed of the set of nodes $\mathcal{V}$ and the set of arcs $\mathcal{A}$. The arcs represent directed radio links between the nodes so that each

This work was supported by the National Science Centre, Poland, under the grant no. 2017/25/B/ST7/02313: "Packet routing and transmission scheduling optimization in multi-hop wireless networks with multicast traffic".

ISBN 978-3-903176-28-7

TABLE I
ADDITIONAL NOTATIONS.

Notation	Description
$\mathcal{B}(s)$	multicast routing tree for sensor $s \in \mathcal{S}$ (rooted at node $r(s)$)
$\mathcal{V}(s), \mathcal{A}(s)$	set of nodes and arcs, respectively, composing tree $\mathcal{B}(s)$
$\mathcal{G}(s)$	set of leaves of tree $\mathcal{B}(s)$ (destination gateways for sensor s)
$\mathcal{V}'(s)$	set of nodes broadcasting in tree $\mathcal{B}(s)$; $\mathcal{V}'(s) := \{r(s)\} \cup \mathcal{V}''(s)$, where $\mathcal{V}''(s) \subseteq (\mathcal{R} \setminus \{r(s)\})$
$\mathcal{V}'(s,w)$	set of nodes appearing after node $w \in \mathcal{V}'(s)$ in tree $\mathcal{B}(s)$ ($\mathcal{V}(s,w) := \delta^+(w) \cap \mathcal{A}(s)$)
$\mathcal{C}(s,w)$	set of c-sets applicable for s at node $w \in \mathcal{V}'(s)$; $\mathcal{C}(s,w) := \{c \in \mathcal{C} : w \in \mathcal{W}(c), \mathcal{U}(c,w) \cap \mathcal{V}'(s,w) \neq \emptyset\}$
$\mathcal{C}(s,a)$	subset of c-sets in $\mathcal{C}(s,o(a))$ used for s on arc $a \in \mathcal{A}(s)$; for $a = (o(a), w)$, $\mathcal{C}(s,a) := \{c \in \mathcal{C}(s,o(a)) : w \in \mathcal{U}(c,o(a))\}$
$\mathcal{S}(w)$	set of sensors s with $w \in \mathcal{V}'(s)$
$p(s,a)$	(unique) arc preceding arc a in $\mathcal{A}(s)$, $a \in \mathcal{A}(s) \setminus \{\delta^+(r(s))\}$
$q(s,g)$	(unique) arc in $\mathcal{A}(s)$ incoming to node $g \in \mathcal{G}(s)$

arc a ($a \in \mathcal{A}$) corresponds to a directed pair nodes; hence, $\mathcal{A} \subseteq \mathcal{V}^2 \setminus \{(v,v) : v \in \mathcal{V}\}$. The set of all arcs originating at node $v \in \mathcal{V}$ will be denoted by $\delta^+(v)$, and the set of all arcs terminating at v by $\delta^-(v)$; hence, $\delta^+(v) := \{(v,w) : w \in \mathcal{V}\} \cap \mathcal{A}$ and $\delta^-(v) := \{(w,v) : w \in \mathcal{V}\} \cap \mathcal{A}$. The network utilizes *time division multiple access* (TDMA) to radio channel.

The set of nodes is divided into two non-empty disjoint sets $\mathcal{R}$ and $\mathcal{G}$, called *routers* and *gateways*, respectively. The routers are traffic originating as well as traffic transiting nodes, while gateways are traffic terminating nodes only, i.e., gateways only receive packets and do not transit them (which implies $\delta^+(g) = \emptyset$, $g \in \mathcal{G}$). Each router r is associated with a set of *sensors* $\mathcal{S}(r)$, where the sets $\mathcal{S}(r)$, $r \in \mathcal{R}$, are pairwise disjoint. Note that some of $\mathcal{S}(r)$ can be empty and the set of all sensors $\mathcal{S}$ is equal to $\bigcup_{r \in \mathcal{R}} \mathcal{S}(r)$. The particular router $r \in \mathcal{R}$ that serves sensor $s \in \mathcal{S}$ will be denoted by $r(s)$, i.e., $r = r(s)$ if, and only if, $s \in \mathcal{S}(r)$.

Each sensor $s \in \mathcal{S}$ generates a stream of packets destined to a given set of gateways $\mathcal{G}(s)$, where $\mathcal{G}(s) \subseteq \mathcal{G}$. Such a stream is composed of a sequence of packets (called s-packets) generated at the beginning of each consecutive TDMA frame. Each s-packet is sent to its destinations $g \in \mathcal{G}(s)$ along a given *multicast routing tree* (tree in short) denoted by $\mathcal{B}(s)$. Such a tree is rooted at router $r(s)$ and its leaves are specified by $\mathcal{G}(s)$. Note that in general the tree may contain other nodes from $\mathcal{R} \setminus \{r(s)\}$. These extra nodes will be denoted by $\mathcal{V}''(s)$, and hence $V'(s) := \{r(s)\} \cup \mathcal{V}''(s)$ is the set of nodes that broadcast s-packets. Thus, $\mathcal{V}(s) = \{r(s)\} \cup \mathcal{V}''(s) \cup \mathcal{G}(s)$. Additionally, the set of arcs composing tree $\mathcal{B}(s)$ will be denoted by $\mathcal{A}(s)$, and the set of nodes appearing after node $w \in \mathcal{V}'(s)$ in tree $\mathcal{B}(s)$ by $\mathcal{V}'(s,w)$. In the following we assume that all packets generated by the sensors are of the same size and that transmitting of one packet takes exactly one time slot of the frame.

On its way to destinations in $\mathcal{G}(s)$, each particular s-packet is broadcast (in general more than once) from each node $w \in \mathcal{V}'(s)$ to its neighbors in $\mathcal{B}(s)$, i.e., to nodes in $\mathcal{V}'(s,w)$. These broadcasts are executed in the consecutive (time) slots of the TDMA frame. Let $\mathcal{T} = \{1, 2, \ldots, T\}$ denote the set of the slots composing the frame and let $c(t)$ denote the so called *compatible set* (c-set in short) applied in slot $t \in \mathcal{T}$. Each such compatible set is characterized by the set $\mathcal{W}(c(t))$ of the nodes (simultaneously) broadcasting in slot t, and by the sets $\mathcal{U}(c(t), w)$, $w \in \mathcal{W}(c(t))$, where $\mathcal{U}(c(t), w)$ is the set of nodes that can successfully decode the signal from node w when all nodes in $\mathcal{W}(c(t))$ are broadcasting. It is assumed that all these sets are non-empty. Note that the same sequence $\widehat{\mathcal{C}} := (c(1), c(2), \ldots, c(T))$ is applied in all consecutive TDMA frames and thus defines the *(frame) transmission pattern*.

Now let us consider the family $\mathcal{C}$ of the c-sets appearing in sequence $\widehat{\mathcal{C}}$, and let $T(c)$ denote the number of times c-set $c \in \mathcal{C}$ appears in $\widehat{\mathcal{C}}$ (note that by definition $T(c) > 0$, $c \in \mathcal{C}$). In the following, the family $\mathcal{C}$ together with the sequence $(T(c),\, c \in \mathcal{C})$ will be called the *frame composition*.

It turns out that the traffic carrying capability of the network depends only on the composition $[\mathcal{C}, (T(c),\, c \in \mathcal{C})]$ of its frame, and not on a particular transmission pattern $\widehat{\mathcal{C}}$ corresponding to this composition. To explain this let us consider a sequence of binary coefficients $h = (h(s,w,c),\, s \in \mathcal{S}, w \in \mathcal{V}'(s), c \in \mathcal{C}(s,w))$, where $\mathcal{C}(s,w)$ is the subfamily of $\mathcal{C}$ containing all c-sets applicable for s at node $w \in \mathcal{V}'(s)$. Using these coefficients, it can be shown (see [6]) that a TDMA frame with composition $[\mathcal{C}, (T(c),\, c \in \mathcal{C})]$ is capable of delivering all s-packets (for each sensor s in $\mathcal{S}$) to their destinations in $\mathcal{G}(s)$ with finite delay if, and only if, there exists a sequence h such that the following conditions hold:

$$\sum_{c \in \mathcal{C}(a)} h(s, o(a), c) \geq 1, \quad s \in \mathcal{S}, a \in \mathcal{A}(s) \qquad (1a)$$

$$\sum_{s \in \mathcal{S}} h(s,w,c) \leq T(c), \quad c \in \mathcal{C}, w \in \mathcal{W}(c) \qquad (1b)$$

(where $o(a)$ denotes the originating node of arc $a \in \mathcal{A}$, and $\mathcal{C}(a)$ is the subfamily of $\mathcal{C}$ containing all the c-sets realizing transmission over arc a, i.e., $c \in \mathcal{C}(a)$ if, and only if, $o(a) \in \mathcal{W}(c)$ and $\mathcal{U}(c, o(a))$ contains the terminating node of arc a). Above, the coefficients in h are interpreted as follows: $h(s,w,c) = 1$ if, and only if, node $w \in \mathcal{W}(c)$ broadcasts an s-packet in one of the slots that applies the c-set c. Hence, condition (1a) ensures that within one frame, for each stream $s \in \mathcal{S}$, one s-packet transmission is realized over each arc in $\mathcal{A}(s)$. Condition (1b), in turn, ensures that there is a sufficient number of occurrences of c to realizes the broadcasts implied by the values in h.

Frame compositions fulfilling the above conditions will be called *feasible*. Note that to check whether a given frame composition $[\mathcal{C}, (T(c),\, c \in \mathcal{C})]$ is feasible we can treat the

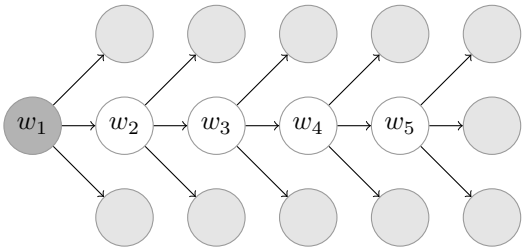

Fig. 1. A network with a multicast packet stream.

coefficients in h as optimization variables, and check, using a mixed-integer programming (MIP) solver, whether the resulting MIP formulation (1) is feasible. Observe that in general for a given feasible frame composition there are multiple solutions h that fulfil conditions (1). In the following any such vector h will be called *packet assignment requirement*.

B. PDMP formulation

Consider a network where (i) fixed multicast trees $\mathcal{B}(s), s \in \mathcal{S}$, (ii) a feasible composition $[\mathcal{C}, (T(c), c \in \mathcal{C}]$, and (iii) a corresponding packet assignment requirement h are given. Now, in order to fully describe the network operation we need to make the following selections:

- ☐ a transmission pattern: choose one of $\frac{T!}{\prod_{c\in\mathcal{C}} T(c)!}$ possible sequences $\widehat{\mathcal{C}} = (c(1), c(2), \ldots, c(T))$ of c-sets that correspond to the assumed frame composition (each such pattern is a permutation with repetitions of the elements c of $\mathcal{C}$, where element c is repeated $T(c)$ times)
- ☐ a packet broadcast schedule: for each $s \in \mathcal{S}, w \in \mathcal{V}'(s), c \in \mathcal{C}(s,w)$ such that $h(s,w,c) = 1$, choose a time slot $t \in \mathcal{T}$ with $c(t) = c$ and reserve the broadcasts from node $w \in \mathcal{W}(c(t))$ for the consecutive s-packets.

Now observe that the so described *packet broadcast schedule* (specified on top of the transmission pattern) determines packet delays. In fact (see [6]), for each $s \in \mathcal{S}$ all consecutive s-packets (let us denote them with $s(1), s(2), \ldots$) will appear at each gateway $g \in \mathcal{G}(s)$ with the same delay $D(s,g)$ measured in the number of time slots elapsed between the epoch in which such a packet is generated at s (note that packet $s(k)$ is generated at the beginning of frame number k) and the time epoch it is delivered to g. Clearly, these delays will in general be different for different choices of the packet broadcast schedule. Thus, defining the maximum delay for sensor s (called *sensor measurement delay*) as $D(s) := \max_{g\in\mathcal{G}(s)} D(s,g)$, $s \in \mathcal{S}$, we can finally state the packet delay minimization problem (PDMP): *given a frame composition and a packet assignment requirement, find a packet broadcast schedule that minimizes the measurement delay over all sensors*.

The so described problem can be illustrated by the following example studied already in [6]. Consider the network shown in Figure 1 that is supposed to carry only one multicast packet stream with the source node (dark grey) and the set of destination gateways (light gray). Assume that the frame

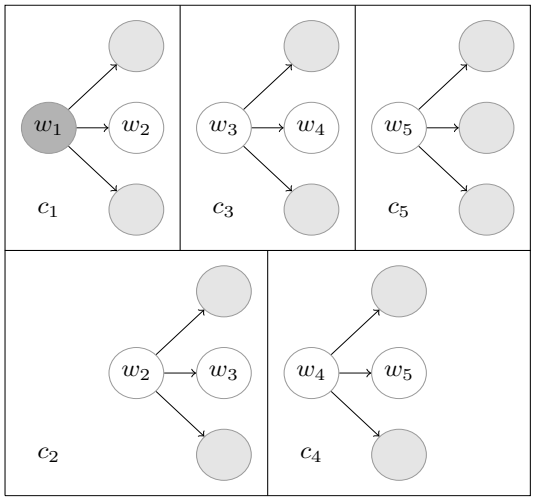

Fig. 2. A family of compatible sets.

is composed of five time slots and each time slot uses one c-set from the family of c-sets depicted in Figure 2. Clearly, each of the considered c-sets has to be used in the frame, i.e., $T(c_1) = T(c_2) = T(c_3) = T(c_4) = T(c_5) = 1$. It is easy to notice that the order in which the considered c-sets are situated in the frame greatly influences the packet delay. Indeed, the sequence $(c_1, c_2, c_3, c_4, c_5)$ results in the packet delay equal to 5 time slots, while the reversed sequence increases the delay to as many as 21 time slots, which shows the importance of appropriate scheduling in terms of the packet delay.

A MIP problem corresponding to the above verbal statement is given in formulation (2). The parameters in that formulation are the frame composition $[\mathcal{C}, (T(c), c \in \mathcal{C}]$, the packet assignment requirement $h = (h(s,w,c), s \in \mathcal{S}, w \in \mathcal{V}'(s), c \in \mathcal{C}(s,w))$, and the upper bound M on the maximum number of frames necessary to deliver a packet to its destination; this means that the maximum delay expressed in the number of slots is not greater than $\overline{T} := M \cdot T$, and hence $\overline{\mathcal{T}} := \{1, 2, \ldots, \overline{T}\}$ is the set all considered time slots. Additional objects used in (2) are the arcs $p(s,a)$ and $q(s,g)$ defined in Table I.

The formulation makes use of the following optimization variables:

- ☐ d delay – continuous variable
- ☐ $x_c^t = 1$ if c-set $c \in \mathcal{C}$ is used in time slot $t \in \mathcal{T}$ of the frame; otherwise $x_c^t = 0$ (binary variables)
- ☐ $Y_{sw}^t = 1$ if node $w \in \mathcal{V}'(s)$ is reserved for broadcasting s-packets, $s \in \mathcal{S}$, in time slot $t \in \mathcal{T}$ of the frame; otherwise $Y_{sw}^t = 0$ (binary variables)
- ☐ $z_{sa}^t = 1$ if packet $s(1)$ from sensor $s \in \mathcal{S}$ is sent over arc $a \in \mathcal{A}(s)$ in time slot $t \in \overline{\mathcal{T}}$; otherwise $z_{sa}^t = 0$ (binary variables)
- ☐ $Z_{sa}^t = 1$ if packet $s(1)$ from sensor $s \in \mathcal{S}$ has been sent over $a \in \mathcal{A}(s)$ in one of the slots between 1 and $t \in \overline{\mathcal{T}}$; otherwise $Z_{sa}^t = 0$ (binary variables).

Packet Delay Minimization Problem:

$$\text{minimize } d \tag{2a}$$

subject to:

$$d \geq \sum_{t\in\overline{\mathcal{T}}} t \cdot z^t_{sq(s,g)}, \quad s \in \mathcal{S}, g \in \mathcal{G}(s) \tag{2b}$$

$$\sum_{c\in\mathcal{C}} x^t_c = 1, \quad t \in \mathcal{T} \tag{2c}$$

$$\sum_{t\in\mathcal{T}} x^t_c = T_c, \quad c \in \mathcal{C} \tag{2d}$$

$$x^{t+(k-1)T}_c = x^t_c, \quad c \in \mathcal{C}, t \in \mathcal{T}, 2 \leq k \leq K \tag{2e}$$

$$\sum_{s\in\mathcal{S}(w)} Y^t_{sw} \leq 1, \quad w \in \mathcal{V}, t \in \mathcal{T} \tag{2f}$$

$$Y^t_{sw} \leq \sum_{c\in\mathcal{C}(s,w)} h(s,w,c)x^t_c, \quad s \in \mathcal{S}, w \in \mathcal{V}'(s), t \in \mathcal{T} \tag{2g}$$

$$\sum_{t\in\mathcal{T}} Y^t_{sw} = \sum_{c\in\mathcal{C}(s,w)} h(s,w,c), \quad s \in \mathcal{S}, w \in \mathcal{V}'(s) \tag{2h}$$

$$Y^{t+(k-1)T}_{sw} = Y^t_{sw}, \quad s \in \mathcal{S}, w \in \mathcal{V}'(s), t \in \mathcal{T}, 2 \leq k \leq K \tag{2i}$$

$$\sum_{t\in\overline{\mathcal{T}}} z^t_{sa} = 1, \quad s \in \mathcal{S}, a \in \mathcal{A}(s) \tag{2j}$$

$$Z^t_{sa} = \sum_{\tau=1}^{t} z^\tau_{sa}, \quad s \in \mathcal{S}, a \in \mathcal{A}(s), t \in \overline{\mathcal{T}} \tag{2k}$$

$$z^t_{sa} \leq Z^t_{sp(s,a)}, \quad s \in \mathcal{S}, a \in \mathcal{A}(s) \setminus \{\delta^+(r(s))\}, t \in \overline{\mathcal{T}} \tag{2l}$$

$$z^t_{sa} \leq Y^t_{sb(a)}, \quad s \in \mathcal{S}, a \in \mathcal{A}(s), t \in \overline{\mathcal{T}} \tag{2m}$$

$$z^t_{sa} \leq \sum_{c\in\mathcal{C}(s,a)} x^t_c, \quad s \in \mathcal{S}, a \in \mathcal{A}(s), t \in \overline{\mathcal{T}} \tag{2n}$$

$$d \in \mathbb{R}; \quad x^t_c \in \mathbb{B}, \ c \in \mathcal{C}, t \in \overline{\mathcal{T}} \tag{2o}$$

$$Y^t_{sw} \in \mathbb{B}, \ s \in \mathcal{S}, w \in \mathcal{V}'(s), t \in \overline{\mathcal{T}} \tag{2p}$$

$$z^t_{sa}, Z^t_{sa} \in \mathbb{B}, \ s \in \mathcal{S}, a \in \mathcal{A}(s), t \in \overline{\mathcal{T}}. \tag{2q}$$

Due to constraint (2b), objective (2a) minimizes, over all sensors s, the maximum of number of slots required to deliver packet $s(1)$ (and hence all subsequent packets $s(2), s(3), \dots$) to all destinations, i.e., to all gateways g in $\mathcal{G}(s)$. This particular objective will be referred to as *min-max delay*.

The three subsequent constraints specify a frame transmission pattern $\widehat{\mathcal{C}}$ in which $c(t) = c$, if, and only if, $x^t_c = 1$. Constraint (2c) assigns exactly one c-set from $\mathcal{C}$ to each slot of the frame, while constraint (2d) ensures that pattern $\widehat{\mathcal{C}}$ is consistent with the assumed frame composition. Constraint (2e), in turn, ensures that $\widehat{\mathcal{C}}$ is repeated in the consecutive frames.

Next, constraints (2f)-(2i) define the packet broadcast schedule consistent with the specified frame transmission pattern and the assumed packet assignment requirement h.

Then, by means of variables z^t_{sa}, and Z^t_{sa}, constraints (2j)-(2n) specify, for each arc a in the tree of sensor s, which time slot in $1, 2, \dots, \overline{T}$ is used to transmit packet $s(1)$ over this arc. Note that constraint (2l) prevents transmitting packet $s(1)$ over a if it has not been transmitted over a yet.

The final three constraints specify the range of the variables, where $\mathbb{R}$ denotes the set of real numbers and $\mathbb{B} := \{0, 1\}$.

PDMP, represented by MIP formulation (2), is an $\mathcal{NP}$-hard problem, as demonstrated in [7] for its special case assuming unicast packet traffic rather than multicast packet traffic assumed in this paper (the unicast case is substantially simpler to treat that the multicast one). Certainly, PDMP can be directly solved by MIP solvers. However, as illustrated in Section IV, such exact solving of (2) becomes inefficient for large networks, and therefore in Section III we will introduce a heuristic algorithm for PDMP.

C. Comments

The first simple comment is that the min-max delay objective used in formulation (2) is not the only relevant for PDMP. For example, the total packet delay $d = \sum_{s\in\mathcal{S}} \sum_{g\in\mathcal{G}(s)} D(s,g)$ (where $D(s,g) = \sum_{t\in\overline{\mathcal{T}}} t \cdot z^t_{sq(s,g)}$) could be used, possibly with additional upper-bound constraints on $d(s)$. Another possibility is min-max fair optimization [8].

The min-max delay optimization problem stated in Section II-B assumes that frame composition $[\mathcal{C}, (T(c), c \in \mathcal{C})]$ and packet assignment requirement h are fixed, and hence have to be preprocessed in a reasonable way. This can be done by finding a frame composition and packet assignment requirement that result in minimum frame length T. To solve the corresponding optimization problem (*frame size minimization problem*) the following mixed-integer programming problem formulation, based on constraints (1) (where quantities $T(c)$ and $h(s,w,c)$ become optimization variables and that is why are denoted as T_c and h_{swc} below) can be used:

$$\min \ T = \sum_{c\in\mathcal{C}} T_c \tag{3a}$$

$$\sum_{c\in\mathcal{C}(a)} h_{so(a)c} \geq 1, \quad s \in \mathcal{S}, a \in \mathcal{A}(s) \tag{3b}$$

$$\sum_{s\in\mathcal{S}} h_{swc} \leq T_c, \quad c \in \mathcal{C}, w \in \mathcal{W}(c) \tag{3c}$$

$$T_c \in \mathbb{R}, \ c \in \mathcal{C} \tag{3d}$$

$$h_{swc} \in \mathbb{B}, \ s \in \mathcal{S}, w \in \mathcal{V}'(s), c \in \mathcal{C}(s,w). \tag{3e}$$

Above, $\mathcal{C}$ represents the family of all possible of c-sets and because of that formulation (3) is non-compact since the number of the elements of $\mathcal{C}$ grows exponentially with the size of the network. This issue was treated in detail in [6] and solved using a price-and-branch approach involving c-set generation. It is important to note that formulation (3) assumes that the multicast trees $\mathcal{B}(s)$, $s \in \mathcal{S}$, are fixed (for example, for each s the shortest-path tree rooted at $r(s)$ can be used). In general, however, combining frame size minimization with tree optimization will lead to significant decrease of the minimum frame size. This is shown in papers [6], [9], where such combined optimization is considered. In any case, the trees used/optimized during frame size minimization are used in formulation (2).

Clearly, the c-set family used in the frame composition assumed for PDMP will contain those c-sets c for which $T_c > 0$ in an optimal solution obtained for (3). Then, the packet assignment requirement assumed for PDMP will involve only values of those h_{swc} for which c belongs to the obtained composition. The so obtained frame compositions will be used in the numerical study presented in Section IV.

Another issue related to PDMP is that the minimum delay d resulting from (2) depends in general on the particular packet assignment requirement h (out of many possible for a given frame composition) used as an element of the input data. To overcome this, we can assume that only the frame composition

is fixed, make $h(s, w, c)$ optimization variables h_{swc}, and add constraints (3b) and (3c) to the formulation.

In fact, when packet assignment requirement optimization is added to (2), we can add frame composition optimization as well. This is achieved by converting the c-set occurrence parameters $T(c)$ to non-negative continuous variables T_c, and adding a new constraint

$$\sum_{c \in \mathcal{C}} T_c \leq T \tag{4}$$

to formulation (2).

In fact, although this extension will in general allow only for a marginal improvement of the minimum packet delay when the family $\mathcal{C}$ resulting from the frame size minimization (3) is used as input to the so modified PDMP, it may decrease the minimum packet delay to a more substantial degree when a larger family $\mathcal{C}$ is assumed for (2). This is because there is a tradeoff between minimization of packet delays and minimization of the frame size. (We are aware of network settings where this is the case.) Thus, from the delay minimization viewpoint it could be beneficial to increase the value of T above its minimum value resulting from (3) and possibly extend the family $\mathcal{C}$ by adding other c-sets generated during the frame size minimization solution process but not used in the final frame composition. Note that in general this procedure will decrease the delay at the expense of decreasing the network throughput since the increasing the frame size implies smaller packet intensity (one per frame) of the streams produced by the sensors.

Observe also that in order to consider all possible compatible sets we could still apply price-and-branch to the just described version of the delay minimization problem, using its linear relaxation for the master problem. However, the effectiveness of such an approach would be unacceptable. Also, we could explicitly embed the problem of finding the c-sets for all the slots in $\mathcal{T}$ into (2). Such a full version of the problem would require adding, for each slot $t \in \mathcal{T}$, a set of constraints (and associated binary variables) forcing that the broadcasts used in the slots represent c-sets (these constraints are given by conditions (2) in [6]). However, this would make the problem virtually intractable for realistic networks.

Finally, let us notice that tree optimization could be added to formulation (2) as well, in one of the ways used in [6] for frame size minimization. Then, the trees $\mathcal{B}(s)$ would be characterized by additional binary variables, constrained by additional conditions added to the formulation. This, however, would lead to an inefficient problem formulation, tractable only for "toy" network instances.

III. Solution Algorithm

Since for realistic networks the number of binary variables in formulation (2) becomes tremendous and thus likelihood of solving it in reasonable time by a MIP solver is low, a heuristic algorithm is in place. For that we make use of the simulated annealing (SA) meta-heuristic [10] given below in the form of Algorithm 1.

Algorithm 1 Simulated annealing (SA)

```
1:  procedure SA
2:      P := initial_solution;
3:      P^best := P; F^best := F(P^best);
4:      Θ := Θ^0;
5:      while (stopping_criterion = false) do
6:          l := 0
7:          while l < L do
8:              Q := neighbor(P)
9:              ΔF := F(Q) − F(P)
10:             if ΔF ≤ 0 then
11:                 P := Q
12:                 if F(P) < F^best then
13:                     F^best := F(P); P^best := P;
14:                 end if
15:             else if random(0, 1) < e^(−ΔF/Θ) then
16:                 P := Q
17:             end if
18:             l := l + 1
19:         end while
20:         reduce_temperature(Θ)
21:     end while
22: end procedure
```

Roughly speaking, the SA algorithm represents a random walk through the points (called *solutions*) of a feasible set $\mathcal{P}$ (called the *solution space*) aiming at minimizing the objective function $F : \mathcal{P} \to \mathbb{R}$. Each solution $P \in \mathcal{P}$ has a specified *neighborhood* $\mathcal{N}(P)$ (where $\mathcal{N}(P) \subseteq \mathcal{P}$), and the moves of the random walk are allowed only to the neighboring nodes. Starting from an initial solution, in each step (the steps are executed in the **while** loop composed of lines 7-19 in Algorithm 1) the algorithm moves from the current solution P to its randomly selected neighbor $Q \in \mathcal{N}(P)$ if such a move does not increase the objective function ($\Delta F \leq 0$) or it passes the so called *Metropolis test* applied in line 15. Note that the test allows for the "uphill" moves, that is for the moves that increase the objective function. The idea behind the test is that the chance of passing the test depends on the value of ΔF (the higher the value the lower the chance) and on the current value of the *temperature* parameter Θ (the lower the value the lower the chance) and when $\Delta F > 0$ or the test is not passed, the constructed random walk trajectory stays at the current point P.

Note that for any fixed temperature value Θ, L steps of the random walk are executed, and since in the main **while** loop (comprising lines 5-21) the temperature is reduced, in the consecutive sequences of L steps the chance of accepting an uphill move is decreasing. When the stopping criterion is fulfilled, the best solution visited by the generated random walk trajectory is stored in P^{best} and F^{best}.

In order to apply the SA procedure to PDMP (with a given frame composition $[\mathcal{C}, (T(c),\, c \in \mathcal{C})]$ and the corresponding packet assignment requirement h) we need to characterize the solution space and specify the functions required in Algo-

rithm 1. This is done as follows.

- ☐ Function $initial\_solution$. To construct the initial solution we first randomly select a transmission pattern $\widehat{\mathcal{C}} = (c(1), c(2), \ldots, c(T))$ realizing the assumed frame composition. In effect, for each $c \in \mathcal{C}$ we define the set $\mathcal{T}(c)$ of time slots that apply the c-set c: $\mathcal{T}(c) := \{t \in \mathcal{T} : c(t) = c\}$ (note that $|\mathcal{T}(c)| = T(c)$). Then, again randomly, we specify the corresponding packet broadcast schedule as follows. We start with assigning a status "unreserved" to all nodes $w \in \mathcal{W}(c(t)), t \in \mathcal{T}$. Then, for each $s \in \mathcal{S}$ and $w \in \mathcal{V}'(s)$, and each c-set $c \in \mathcal{C}(s, w)$ such that $h(s, w, c) = 1$, we choose at random a slot $t \in \mathcal{T}(c)$ among the slots where node w is unreserved and reserve it (and change its status to "reserved") for broadcasting s-packets. Clearly, the so obtained packet broadcast schedule will be, due to fulfilment of inequality (3c), consistent with the assumed packet assignment requirement h.
- ☐ Objective function F. In order to calculate the value of $F(P)$ notice, that solution P constructed by function $initial\_solution$ (and every solution Q generated during execution of Algorithm 1) defines a solution of PDMP, i.e., a feasible solution of formulation (2). More precisely, solution P specifies feasible values $x(t, c)$ corresponding to variables x^t_c, and $Y(t, s, w)$ corresponding to variables Y^t_{sw} in (2). Using these values, the values of $z(t, s, a)$ and $Z(t, s, a)$ (corresponding to variables z^t_{sa} and Z^t_{sa}, respectively) can be computed from conditions (2j)-(2n), leading to the value $F(P)$ of the SA objective function. This value is simply the smallest d fulfilling all inequalities in (2b). (In fact, for given $x(t, c)$ and $Y(t, s, w)$, the value of $F(P)$ can be efficiently calculated through an appropriate algorithm that does not involve calculation of $z(t, s, a)$ and $Z(t, s, a)$.)
- ☐ Function $stopping\_criterion$. Returns **true** when Θ becomes less than or equal to an assumed final value $\overline{\Theta}$.
- ☐ Function $neighbor(P)$. A solution obtained by swapping the c-sets (preserving the reservations assigned to broadcasting nodes in each of the swapped c-sets) in P between two randomly selected time slots in $\mathcal{T}$.
- ☐ Function $random(0, 1)$. Returns a random number from interval $[0, 1]$.
- ☐ Function $reduce\_temperature(\Theta)$: $\Theta := \alpha \times \Theta$ for a given parameter $alpha$ $(0 < \alpha < 1)$.

Note that frame composition $[\mathcal{C}, (T(c),\ c \in \mathcal{C})]$, packet assignment requirement $h = (h(s, w, c),\ s \in \mathcal{S}, w \in \mathcal{V}'(s), c \in \mathcal{C}(s, w))$, initial temperature Θ^0, final temperature $\overline{\Theta}$, temperature reduction factor α, and number of steps in a single iteration L are input parameters for the SA algorithm.

It should be noted that the time slot permutations of the initial solution considered in the SA algorithm do not exhaust the whole solution space. As an example consider two time slots (t' and t''), two packet streams (s' and s''), and a c-set c with two transmitting nodes (w' and w''). Assume that as a result of the frame size minimization we obtain the following values: $T(c) = 2$, $h(s', w', c) = 1$, $h(s', w'', c) = 1$, $h(s'', w', c) = 1$, $h(s'', w'', c) = 1$. Now consider the initial solution with both nodes of c applied in slot t' reserved for s'-packets, and both nodes of c applied in slot t'' reserved for s''-packets. For this solution the only neighbor is the solution with both nodes of c applied in slot t' reserved for s''-packets, and both nodes of c applied in slot t'' reserved for s'-packets. Hence, there is no possibility to obtain the solution (and its swapped version) where nodes w' and w'' in time slot t' are reserved for s'-packets and s''-packets, respectively, and nodes w' and w'' in time slot t'' are reserved for s''-packets and s'-packets, respectively. This issue is important since the latter solutions can have different min-max delays. In our numerical study presented in Section IV this issue is taken into account by starting multiple SA algorithm instances concurrently, each from a different (randomly generated) initial solution.

IV. Numerical Study

Below we present a numerical study illustrating effectiveness of solving PDMP by means of the simulated annealing algorithm described in Section III, as well as in exact way by solving formulation (2) using a MIP solver. The SA algorithm and all optimization models related to formulations (2) and (3) were implemented in C# and executed on a dedicated Windows Server 2016 Datacenter x64 virtual machine, configured for 20 logical processors and up to 80 GB of RAM. CPLEX 12.9.0 was used for solving the MIP formulations.

A. Network setting

We consider irregular network topologies of five different sizes: extra small networks with $|\mathcal{V}| = 20$ nodes, small networks with $|\mathcal{V}| = 30$ nodes, medium networks with $|\mathcal{V}| = 40$ nodes, large networks with $|\mathcal{V}| = 50$ nodes, and extra large networks with $|\mathcal{V}| = 60$ nodes. The networks were generated using the tool available at [11] that places nodes randomly in an assumed square network area according to the uniform distribution. To achieve constant network density we increase the network area proportionally to $|\mathcal{V}|$ and assume the network area equal to 163 m $\times$ 163 m for extra small networks, 199.5 m $\times$ 199.5 m for small networks, 230 m $\times$ 230 m for medium networks, 257.5 m $\times$ 257.5 m for large networks, and 282 m $\times$ 282 m for extra large networks. In each network, 40% of nodes are the *traffic originating routers*, i.e., the routers r in $\mathcal{R}$ with non-empty sets of associated sensors $\mathcal{S}(r)$ (note that these routers are also capable of transiting packets), 15% of nodes are the gateways (for small and large networks the number of the gateways was rounded up), and the remaining nodes are the *(purely) transit routers* with $\mathcal{S}(r) = \emptyset$. Examples of such networks (one for each of the considered sizes) are depicted in Figures 3-7, where Network 1 has 20 nodes and 97 links, Network 2 has 30 nodes and 174 links, Network 3 has 40 nodes and 263 links, Network 4 has 50 nodes and 406 links, and Network 5 has 60 nodes and 451 links. In the figures, the traffic originating routers, the gateways, and the transit routers are shown in red, blue, and white, respectively.

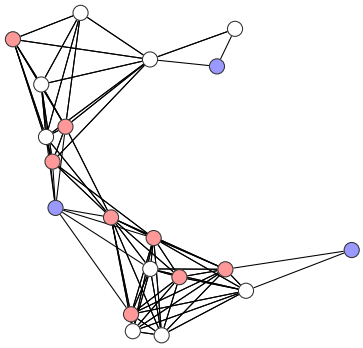

Fig. 3. Network 1

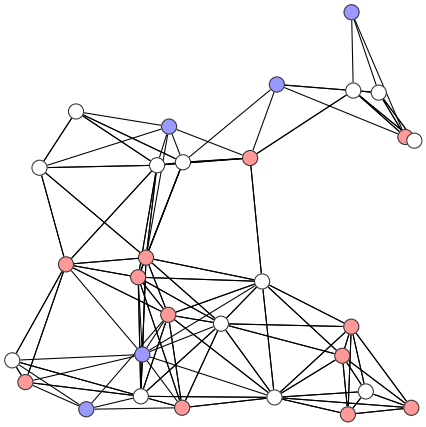

Fig. 4. Network 2

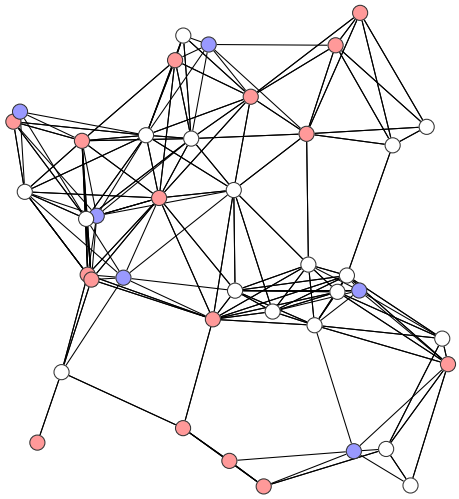

Fig. 5. Network 3

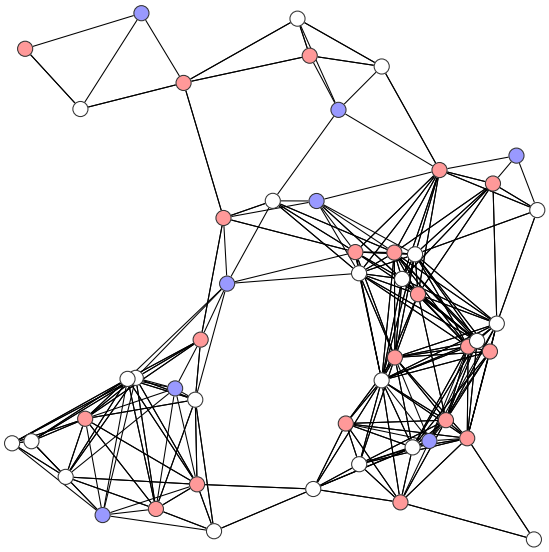

Fig. 6. Network 4

Each link depicted in the figures is bi-directed and represents two oppositely directed arcs between its end nodes. Such a link is provided between two given nodes if, and only if, the distance between these nodes does not exceed the maximal transmission range, which is calculated as follows. First, to obtain the power received at a node we use the propagation model described in [6]: the power (expressed in mW) received at node w when node v is broadcasting is defined as the power of the transmitter (which is the same for each node and equals 100 mW) multiplied by $d(v,w)^{-4}$, where $d(v,w)$ (in m) is the distance between the nodes. Then, assuming noise power equal to -101 dBm, and setting the threshold for the signal to noise ratio (SNR) to 8 dB, we calculate the maximal transmission range, which is equal to 66.8 m.

B. SA parameters setting

The parameter values used in the SA algorithm described in Section III are as follows. In the case of extra small, small, medium, and large networks the initial and the final temperatures are set to $\Theta^0 = 5$ and $\overline{\Theta} = 0.1$; for extra large networks we assume $\Theta^0 = 3$ and $\overline{\Theta} = 0.06$. The temperature is reduced with factor α set to 0.9. Note that since reaching the final temperature is used as a stopping criterion, in both cases the number of iterations is always equal to 38. In each iteration i, $L = 20\,000$ steps of the algorithm are performed, and hence the SA random walk through the solution space is $760\,000$ steps long. As explained in Section III, for better exploration of the solution space when optimizing each network instance, we have used a number of different random initial solution. In the computations we have used 20 initial solutions for each network and we have run the corresponding SA algorithm instances in parallel, each in a separate thread of a logical server.

C. Results of the delay optimization

Below we present the numerical results on delay optimization. In the considered cases the set of destination nodes for each sensor includes all gateways, and each traffic originating router is associated with just one sensor (and hence is the source of only one packet stream). For solving PDMP, the frame composition, the packet assignment requirement and the multicast routing trees for the traffic originating routers (which are the shortest-path trees, precomputed by means of Dijkstra's

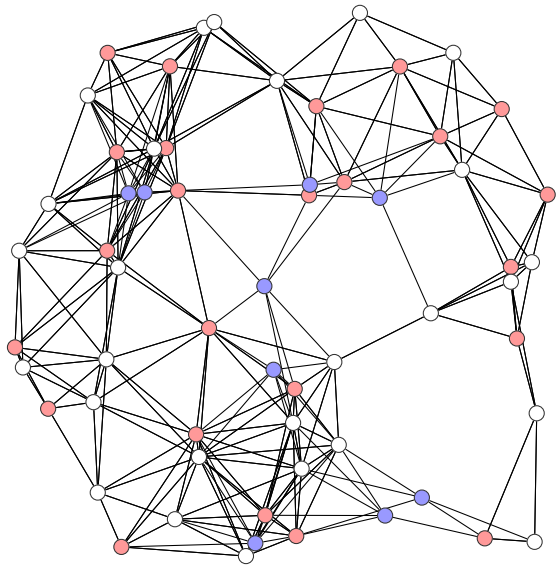

Fig. 7. Network 5

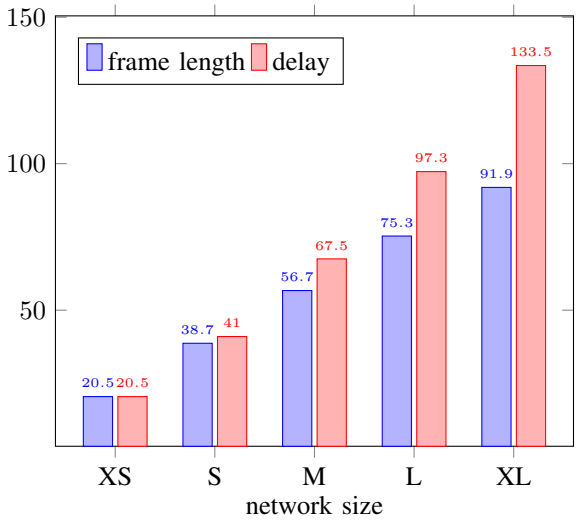

Fig. 8. Frame size and minimized delay as a function of the network size.

TABLE II
NUMERICAL RESULTS – EXPLANATION

Notation	Description
T^*	optimal frame length obtained from frame size minimization
d^*_{MIP}	minimum delay obtained by solving the MIP formulation of PDMP
d^0_{avg}	average initial delay for initial solutions used in concurrent SA algorithm instances
d^*_{avg}	average delay for final solutions of concurrent SA algorithm instances
d^*_{SA}	the best minimum delay found by means of the concurrent SA algorithm
t^f	computation time required for solving frame size minimization
t^d_{MIP}	computation time required for solving the MIP formulation of PDMP (timeout for the computation time equal to 3h was set in this case, and (*) denotes that this timeout was reached)
t^d_{SA}	computation time required for solving all concurrent SA instances

algorithm) obtained from the frame minimization problem (3) are assumed. Notations used in this section are described in Table II.

The results for the networks depicted in Figures 3-7 are presented in Table III. The main observations are as follows. First, the optimization process leads to a significant delay decrease. The optimized delay is up to 3.08 times smaller than the average initial delay (this is the case for Network 5; for Networks $1-4$ these values are equal to 2.63, 2.84, 2.89, and 3.01, respectively). Next, the SA algorithm is able to find solutions that are optimal or close to the lower bound, i.e., the minimum frame size T^*. (Note that the minimum frame size is indeed the lower bound for d since if the minimum d were strictly less than T^*, then all packets would reach their destinations before the frame end, and hence the slots after the slot no. d would not have to be used.) For Network 1 the optimal solution was found, while for Networks $2-5$ the optimized delay is 10.64%, 17.30%, 31.94%, and 39.24% greater than the lower bound, respectively, and this means that all packets are delivered well before the second frame ends.

Let us observe that the MIP formulation (2) of PDMP was successfully solved (before the 3 hour timeout was reached) only for the two smallest networks, and provided the exact minimum delay $d^* = 23$ (equal to the minimum frame size in this case) for Network 1 and $d^* = 48$ for Network 2. Notice that for the case of Network 1 SA algorithm found the optimal solution as well, while for Network 2 the difference between SA solution and optimal solution is equal to only 8.33%. Note also that within the timeout, only feasible PDMP solutions for Network 3 were found, while even this was not possible for Network 4 and Network 5. Yet, using the SA algorithm we were able to obtain the near optimal solutions even in those computationally demanding cases in a reasonable time. Note that the SA solution ($d^*_{SA} = 61$) for Network 3 found in 16 minutes and 35 seconds is substantially better than the best feasible solution ($d^*_{MIP} = 102$) achieved by the MIP solver in 3 hours.

The results obtained with the SA algorithm averaged over 10 randomly generated networks for each of the considered sizes are presented in Table IV and depicted in Figure 8 (where the letters XS, S, M, L, and XL denote extra small, small, medium, large, and extra large networks, respectively). The results confirm satisfactory effectiveness of the SA approach. For all extra small networks the optimal solutions were found. For small, medium, large, and extra large networks, the optimized delay is on the average 5.94%, 19.04%, 29.21%, and 45.26% greater than the lower bound, respectively.

D. Convergence of the SA delay minimization process

Convergence of a selected SA algorithm run for Network 3 (the one that found the best solution) is illustrated in Figure 9.

TABLE III
DELAY OPTIMIZATION RESULTS

	T^*	d^*_{MIP}	d^0_{avg}	d^*_{avg}	d^*_{SA}	t^f	t^d_{MIP}	t^d_{SA}
Network 1	23	23	60.55	24.85	23	1.5s	31s	3m28s
Network 2	47	48	147.55	52.9	52	10s	55m47s	9m19s
Network 3	52	102	176.7	64.35	61	1m5s	3h (*)	16m35s
Network 4	72	-	286.3	100.55	95	1m15s	3h (*)	35m50s
Network 5	79	-	339.2	114.95	110	1m50s	3h (*)	45m33s

TABLE IV
AVERAGED DELAY OPTIMIZATION RESULTS

	T^*	d^0_{avg}	d^*_{avg}	d^*	t^f	t^d_{SA}
XS	20.5	48.1	20.84	20.5	2.54s	2m38s
S	38.7	141.11	43.42	41	11.31s	8m19s
M	56.7	215.66	72.17	67.5	42.5s	18m26s
L	75.3	302.97	102.05	97.3	1m39s	38m15s
XL	91.9	423.5	141.27	133.5	2m48s	58m12s

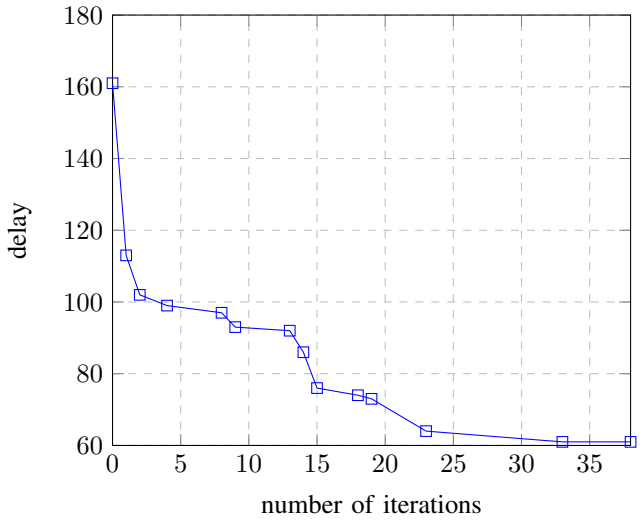

Fig. 9. Convergence of the delay minimization process.

In this case the gap between the initial solution ($d^0 = 161$) and the final solution ($d^*_{SA} = 61$) was equal to 100. This gap was reduced by 48% already in the first iteration, and became equal to 15% of the initial gap after 15 iterations. Then, in the next 8 iterations, the gap was further decreased to 3% (this took place in iteration number 23). Finally, it took 10 extra iterations (when the algorithm terminated) to reduce the gap 0. The observed behaviour suggests that if the computation time is important and slightly worse solutions can be accepted, the SA iterative process could be terminated substantially earlier than indicated by the stopping criterion.

V. CONCLUDING REMARKS

The presented paper deals with a difficult issue of optimizing packet delay in multi-hop wireless TDMA networks with periodic traffic, and in particular with min-max packet delay optimization for the case when the frame composition, the packet assignment requirement and the multicast routing trees are known and fixed, and then only an optimal permutation of the c-sets composing the frame and a corresponding packet broadcast schedule are to be found. This problem turns out to be difficult ($\mathcal{NP}$-hard), and its exact mixed-integer formulation is in general applicable only for small networks. Because of that we have developed a simulated annealing based heuristic algorithm that, as shown by means of a numerical study, gives near-optimal solutions in reasonable time.

As far as future work is concerned, we are going to investigate the extensions of the packet delay minimization problem discussed in Section II-C, especially those dealing with enlarging the frame size in order to decrease the delay. This is planned to be done within the grant listed in the acknowledgment.

REFERENCES

[1] I. Akyildiz, X. Wang, and W. Wang, "Wireless Mesh Networks: a Survey," *Computer Networks*, vol. 47, no. 4, pp. 445–487, 2005.

[2] D. Benyamina, A. Hafid, and M. Gendreau, "Wireless Mesh Networks Design: a Survey," *IEEE Communications Surveys and Tutorials*, vol. 14, no. 2, pp. 299–310, 2012.

[3] P. Rawat, K. Singh, H. Chaouchi, and J. Bonnin, "Wireless Sensor Networks: a Survey on Recent Developments and Potential Synergies," *J Supercomput*, vol. 68, pp. 1–48, 2014.

[4] O. Khader, A. Willig, and A. Wolisz, "WirelessHART TDMA Protocol Performance Evaluation Using Response Surface Methodology," in *IEEE BWCCA*, October 2011, pp. 197–206, published online, DOI: 10.1109/BWCCA.2011.32.

[5] M. Nixon, *A Comparison of WirelessHART and ISA100.11*. White Paper: HCF_SPEC-xxx, July 2012, revision 1.0, Preliminary A.

[6] M. Pióro, A. Tomaszewski, and A. Capone, "Maximization of multicast periodic traffic throughput in multi-hop wireless networks with broadcast transmissions," *Ad Hoc Networks*, vol. 77, pp. 119–142, 2018.

[7] Y. Li, A. Capone, and D. Yuan, "On End-to-end Delay Minimization in Wireless Network under Physical Interference Model," in *IEEE INFOCOM*, April 2015, pp. 2020–2028, published online, DOI: 10.1109/INFOCOM.2015.7218586.

[8] W. Ogryczak, M. Pióro, and A. Tomaszewski, "Telecommunications Network Design and Max-Min Optimization Problem," *Journal of Telecommunications and Information Technology*, vol. 3, pp. 43–56, 2005.

[9] A. Tomaszewski and M. Pióro, "Packet Routing and Frame Length Optimization in Wireless Mesh Networks with Multicast Communications," in *Proc. of the 17th International Network Strategy and Planning Symposium (NETWORKS 2016)*, September 2016, pp. 1–8.

[10] M. Pióro and D. Medhi, *Routing, Flow, and Capacity Design in Communication and Computer Networks*. Morgan Kaufmann, 2004.

[11] E. Fitzgerald, "Wireless network scenario generator," https://bitbucket.org/EIT_networking/network_generator, accessed 2020-01-07.