Received May 17, 2021, accepted May 22, 2021, date of publication June 9, 2021, date of current version June 17, 2021.

*Digital Object Identifier 10.1109/ACCESS.2021.3087861*

# Internetwork Time Synchronization of Mobile Ad Hoc Networks

**HARRI SAARNISAARI, (Senior Member, IEEE), JUHO MARKKULA, (Graduate Student Member, IEEE), TUOMAS PASO, AND TIMO BRÄYSY**
Centre for Wireless Communications, University of Oulu, 90570 Oulu, Finland

Corresponding author: Harri Saarnisaari (harri.saarnisaari@oulu.fi)

This work was supported in part by the European Defence Agency Project ''Multiband efficient networks for ad hoc networks'' (MAENA) under Grant B-1476-IAP4-GP, and in part by the Academy of Finland 6Genesis Flagship under Grant 318927.

**ABSTRACT** Several time synchronous ad hoc networks may operate in the same geographical area. Although network time synchronization within an individual network has been widely discussed, internetwork time synchronization relying on time sources the networks carry has not been addressed. Therefore, this paper proposes an internetwork time synchronization procedure that allows to use one network as a time source and share its time, like a global navigation satellite system time, to other networks. The internetwork time synchronization procedure runs in gateway nodes that could be a member in several networks, and it collaborates with the individual network time synchronization algorithms that operate at the network level. Consequently, the paper proposes a generic, robust hybrid network time synchronization protocol and analyses its performance using a sensible metric called a mutual consensus. It is shown that the common time is achieved with a limited variance in the networks that operate in the master-slave, distributed or hybrid mode. Provided numerical examples illustrate the behavior of the algorithm and confirm the theoretical findings.

**INDEX TERMS** Networks, synchronization, discrete systems, error analysis.

## I. INTRODUCTION

Time synchronous ad hoc networks have various usage including mobile, vehicular and ship ad hoc networks (MANET, VANET and SANET, respectively) but ideas are used also in wireless sensor networks. Consequently, ad hoc network synchronization has been investigated rather deeply over the years. However, it is obvious that a geographical area could contain several radio networks on the same frequency band. In this case, internetwork[1] time synchronous operation would be beneficial, e.g., if dynamic spectrum access or frequency hopping were applied, since that would minimize interference between networks and optimize the use of frequency resources. In addition to communications reasons, several applications would benefit from or even require a common time. Consequently, network time synchronization (NTS) is of great interest both in individual networks but also over different networks though the latter subject is not usually touched in the ad hoc network context. As an additional viewpoint, one of the networks could contain a high quality time source that other networks should or even are required to apply if that is available. An example of such a time source is the one available from global navigation satellite systems (GNSS).

In the individual network level NTS is a well-studied subject and recent references like [1]–[6] without forgetting more distant but often mentioned [7] include the valuable overviews of the progress and proposed methods. There are three well known principles, or strategies, for NTS that are adopted though details, or tactics, vary. These are the master-slave, distributed (or mutual coupling) and hybrid synchronization strategies as addressed in the classics [8]–[10]. Since the literature is rich, just a short overview of the main tactical principles and approaches is provided in the following paragraph.

The associate editor coordinating the review of this manuscript and approving it for publication was Dongxiao Yu.

[1] ''Two or more networks connected by routers and bridges. The largest Internetwork is the Internet'', see https://www.oxfordreference.com/, A Dictionary of the Internet (2 ed.), Edited by Darrel Ince, Publisher Oxford University Press, 2009, eISBN: 9780199571444.

A common feature is that nodes sent their time. Nodes' time may be send as a time stamp or be tied to the signal structure. Some NTS schemes use a virtual clock time that is not touched and a consensus time that is adjusted in each node, and both times are transmitted. Time correction may be based on observed time difference or estimated clock offset and skew. Some schemes assume that propagation delay is negligible, which is true if it is insignificant with respect to the required synchronization accuracy and could usually occur if nodes are in a small geographical area. The propagation delay can be compensated or not. The former requires two-way messaging or known node positions whereas the latter usually applies one-way messaging. NTS schemes could be purely distributed, purely master-slave or hybrids such that they are master-slave down to a certain point in a synchronization tree and distributed thereafter. Some schemes have preselected master (or anchor or root or reference) nodes whereas some dynamically select these. The hybrid form is also possible for robustness such that the distributed mode is activated if the master is lost. Indeed, this kind of robustness is an important factor in military ad hoc networks since GNSS systems, used as a master time source, are known to be vulnerable to jamming and interference.

To this end, it is mentioned that the goal of NTS is to achieve a consensus about time in the network. Indeed, many NTS schemes are named as consensus algorithms since they are alike consensus algorithms. Studies have shown the generic properties of the consensus algorithms with respect to the average consensus value and the mean square metric [11], [12], but especially the former is not the best measure since it does not explain the total picture. Herein, a mutual consensus is proposed as a sensible measure and the analysis in this paper is based on it.

The common time should be achieved in the shortest possible time (convergence rate), have required accuracy derived from the requirements, and the time distribution mechanism should be flexible, adaptable, scalable, resilient and allow dynamicity. Furthermore, it should use communications resources as little as possible, i.e., minimize the required control traffic. With these goals in mind, this paper *introduces an internetwork time synchronization procedure* that utilizes nodes that could be members in several networks and use the time sources available in the involved networks. Indeed, this might be the first time such an idea is presented. A driving force is that the NTS schemes in the different networks are not touched, but instead a new, common time is proposed to them. In the individual network level, this paper *proposes an enhanced hybrid NTS procedure* that is able to use external time source as a master. This part is an extension of our early work [13]. Finally, the paper *shows that the generic hybrid NTS algorithm achieves the mutual consensus* that extends our early work on the average consensus [14] inspired by [15], [16]. Discussion about using higher order filtering within the NTS algorithm is also included.

In practice, wireless communication systems are synchronized in steps rather than in a continuous flow. Consequently, NTS algorithm analysis results do not explain what is exactly occurring at a certain moment but show generic performance and may explain behavior during individual steps and explain how many message exchanges are needed. As a view to this stepwise progress, assume that a set of radios is switched on at the "same time". Since communication systems operate usually at ms or $\mu$s level, humans cannot put them on this precisely such that there will be significant time offsets. At the beginning, the radios can just listen and send hello messages including NTS elements. Once a radio (or a node) observes one NTS message, it can start a synchronization process with the sender, and they may become pair-wise synchronized before they observe other hello messages or others observe their hello messages. Indeed, many subnetworks may have been born during this initial phase. However, gradually all the radios belonging to the same network will be synchronized. Therefore, this paper *considers merging from the NTS point of view*, that is not as an easy task as one might think.

The remaining paper is organized as follows. Section II explains the system and assumptions. The network and internetwork level NTS processes are explained in section III. The hybrid NTS algorithm is analyzed in section IV. Simulation results illustrating some performance aspects of the algorithm are shown in section V and, finally, conclusions are drawn in section VI.

## II. SYSTEM DESCRIPTION

A network is a set of radios (or nodes) that are allowed to communicate and, therefore, should be synchronized if they are in reach. This belonging (to a network) could be managed, e.g., by a node identification (ID). Furthermore, each network is assumed to have a unique network ID. If a network is partitioned, the parts are called subnetworks, even though they are able for independent operation. Initially, nodes do not know whom they could reach, i.e., who are their neighbors. Consequently, initial NTS is closely engaged in neighbor discovery.

Different networks cannot directly communicate but just through dedicated multiradio nodes that could be called network gateways (or bridges). If networks are connected (via multiradio nodes), they are called a joint network and if they are also time synchronized, they are called as a (joint) synchronized network. These multiradio nodes are responsible for internetwork routing as well as other internetwork management functionalities including internetwork time synchronization. Furthermore, some networks may have an ultimate time source that is desired to be used in other networks if possible. Even consumer level GNSS devices provide a time that is a few tens of ns to a few hundreds of ns from the Coordinated Universal Time (UTC) [17]. It might even be a requirement that if the GNSS time is available, it has to be used. This requirement is adopted in this paper.

The network must be connected, i.e., there is a direct or multihop path from each node to all other nodes. However, link failures are allowed, as well as disappearing nodes. If the connectivity is lost, the network may be split into two (or more) subnetworks that manage their own time until they merge.

Nodes send NTS information once per a time synchronization period (TSP), or more often if increased robustness or faster convergence during the initial phase is desired. NTS information includes information that is common with other procedures such as routing, neighbor discovery and so on. Therefore, NTS information could be in the NTS message, distributed into the NTS message and other messages or integrated into one joint message with other control information (e.g., an integrated "hello" message), assuming its frequency is sufficient for NTS purposes. The processing of NTS information is done once per time synchronization period and time is corrected at the end of the period. This allows nodes to receive all NTS information (from several nodes and/or consecutive ones) that is possible to reach during the period resulting in an average consensus and increased robustness. This also avoids possible ping-ponging that may occur if adjustments are done after each reception. If NTS information is not received during a TSP, past decision (i.e., a clock correction value) is used since that is the best available information. In addition, this adds robustness. Naturally, mobility defines how long past values are valid.

## A. INDIVIDUAL NETWORKS

Once radios are synchronized, they still have to transmit and listen hello messages to allow late or re-entry and merging after the split or loss of synchronization. It is its own art to design suitable transmit and listening epochs (of hellos) such that all this is possible in a reasonable time, but that is not touched in this paper. The network's NTS protocol describes the overall time synchronization process flow, i.e., how related messages are formed, transmitted and processed once received. The network's NTS algorithm describes how its time is adjusted in individual nodes. The NTS protocol handles time synchronization but it is also related to late entry and merging as discussed in this paper.

## B. MULTIPLE NETWORKS

Different networks can be connected via network gateway nodes that host an extended NTS (ENTS) protocol that manages internetwork synchronization. This protocol will be described in this paper. Furthermore, such nodes are called ENTS nodes. The gateway concept is illustrated in Fig. 1. Each radio (A, B, C and D in the figure) belong to a different network. What makes the situation a challenge is that there could be several gateway nodes in a network and the decision about the time to be used must be unambiguous in all of them. Finally, distinct multiradio nodes may have different joint networks. This complexity is illustrated in an example shown in Fig. 2. Therein, three ENTS nodes are connected to a different number of networks (A, B and C) but the time sharing mechanism must allow time distribution into all networks through the ENTS nodes.

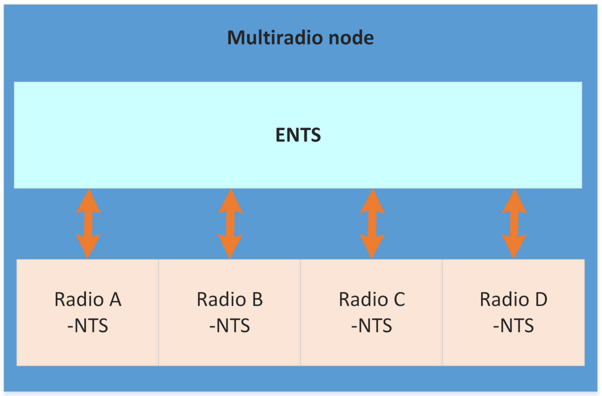


**FIGURE 1. A multiradio node with the ENTS protocol.**

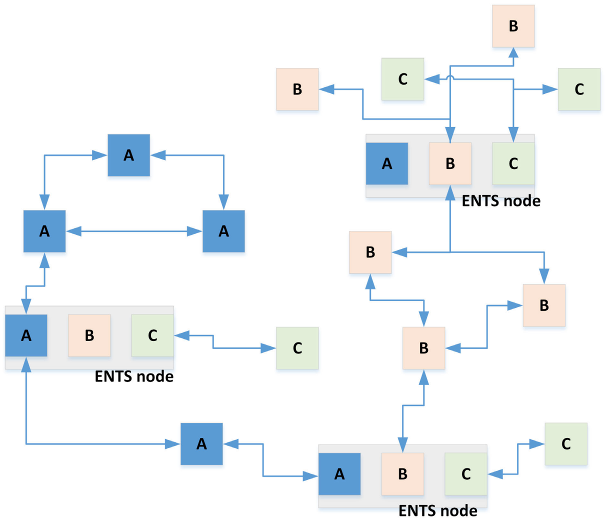


**FIGURE 2. An example multiple networks case with three ENTS nodes and three networks.**

## C. CLOCK

Let the time of the $i$th node be $T_i(k) = T_0(k) + t_i(k)$, where $T_0(k)$ is the common reference ("true") time and $t_i(k)$ deviation from that. The deviation of a free running clock may be modeled as [18]

$$t_i(k) = T_{\text{init},i} + \beta_i t(k) + 0.5\alpha_i t^2(k) + \dots, \tag{1}$$

where $t(k)$ refers to the (used) true time and $T_{\text{init},i}$ denotes the initial time offset. The skew of the clock $\beta_i = f_{r,i}/f_0 - 1$, where $f_0$ and $f_{r,i}$ are the nominal and actual frequency of the clock and, finally, $\alpha_i$ denotes the frequency drift of the clock. However, the time synchronization period is usually selected so short that the higher order effects are insignificant such that just the first two terms, namely clock offset and skew, are needed in modeling clocks in many practical systems.

A physical clock is hardware that provides ticks at certain intervals. Different clocks have different frequency accuracy (with respect to the nominal frequency $f_0$) resulting in skew and thus their time intervals differ. Virtual clocks read ticks to advance their time. Often, virtual clocks or their derivatives are adjusted in NTS processes.

### D. TIME STAMP

The time reading or the time stamp of a virtual clock could be sent as actual clock reading, usually performed at the MAC layer in order to reduce uncertainties as explained, e.g., in [5]. Alternatively, in synchronous communication systems transmission is allowed only at certain moments (beginning of the slot) such that the exact time is tied to transmit moments and the remaining task is to send the slot level timing, often as a combination of superframe, frame and slot numbers.

### E. ABOUT NTS ACCURACY REQUIREMENT

Let $R$ be the (maximum) NTS error after convergence and directly after clock adjustment at the end of the time synchronization period with duration $T_A$. Let $S = \beta T_A$ denote the (max) drift during $T_A$ due to skew of the physical clock and let $G$ denote the goal NTS accuracy after $T_A$. Obviously, $R + S < G$. Consequently, clock quality impacts the parameter selection in the NTS processes.

Let $D_{\max}$ be the maximum expected propagation delay or communication range in seconds. The guard interval between transmission must be at least $D_{\max}$ to prevent overlaps, but actually it must be $D_{\max} + R + S < D_{\max} + G$.

Storywise, it would not be the correct place to talk about the achieved accuracy $R$ before analysis results, but since basic results are readily known from the earlier works, this does not matter. Consequently, since NTS accuracy affects the requirements, the main conclusions about expected accuracy are shown here.

Ultimate accuracy will be the time delivery accuracy. If the time stamp were read, written and sent accurately, the only remaining error is time stamp reading accuracy at the receiver, i.e., time-of-arrival estimation (TOA) error $E$. This is inversely proportional to the signal bandwidth $B$ such that accuracy is typically better than $1/4B$. For 1 MHz signal this means better than 0.25 $\mu$s and for 10 MHz signal better than 25 ns accuracy such that time delivery accuracy is usually rather insignificant, except if you are sharing a high-quality time. If time stamps cannot be expressed accurately, these errors should be added to this value.

In the fully distributed mode, propagation delay cancellation is not necessary since mutual consensus averages delays away and the NTS error will be in the order of skew effect, i.e., $\beta T_A$ or, if $E$ is not insignificant, $\sqrt{(\beta T_A)^2 + E^2}$.

In the master-slave systems or in master-slave links in the hybrid systems, uncompensated propagation delays accumulate in each hop along the timing tree. Consequently, if the total delay is $x$ $\mu$s to the furthest node, its accuracy with respect to the master is in the order of $x$ $\mu$s, if other effects are assumed to be insignificant. Therefore, if propagation delays play a remarkable role in the requirement budget, they should be compensated in the master-slave or hybrid systems.

## III. NTS PROTOCOLS

### A. NTS IN INDIVIDUAL NETWORKS

The design principles are: i) GNSS time has to be used as a master time source if available, ii) external master time must be allowed and used if available, iii) if a master time is not (anymore) available, the distributed mode is adapted, iv) the master-slave chain does not need to go down the bottom, but after some hops from a master the distributed mode is allowed. Further driving force is: v) delay compensation could be used in the master-slave mode, if so desired. Obviously, nodes having master time do not adjust their time. Sometimes it might be good if a node was set as a master time source, even temporarily, e.g., for faster initial convergence though principle ii) was intended for the ENTS usage. If the network has multiple master time sources, it is assumed that they have a common time.

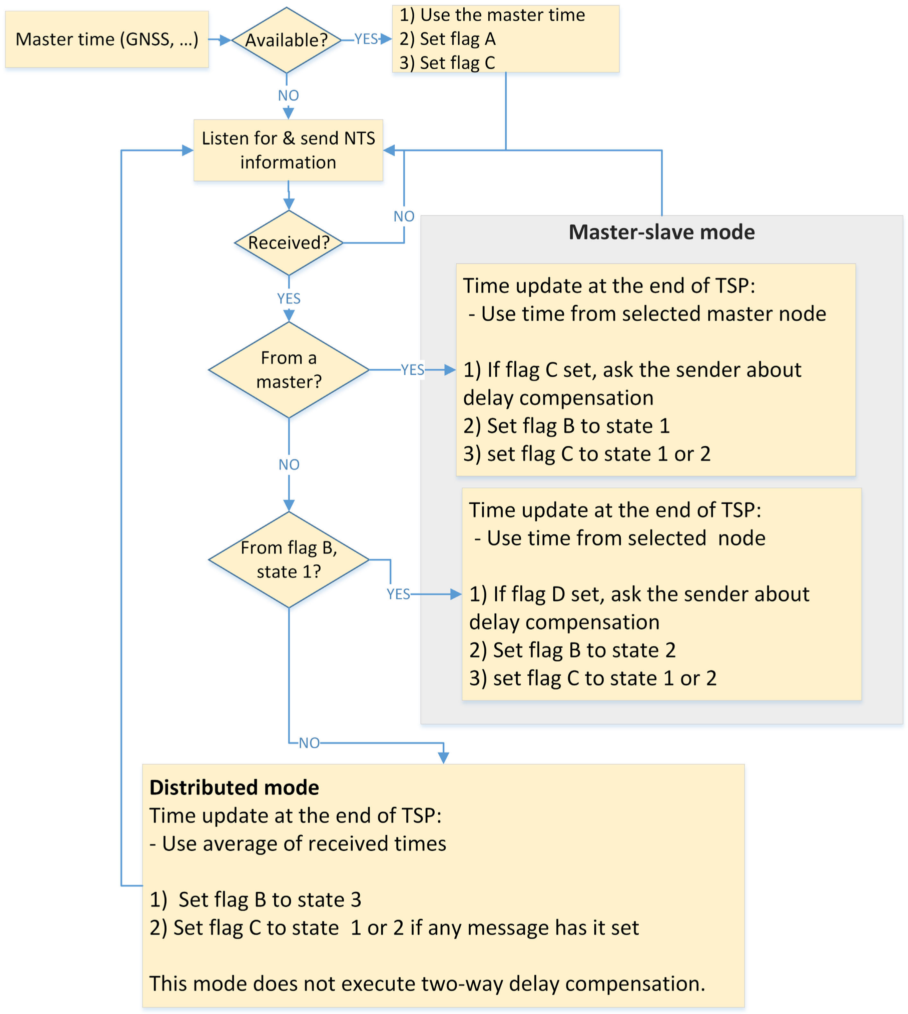
To make the NTS protocol behave the above described way, a set of flags shown in table 1 is introduced. These flags are sent as the NTS information along with the time information. Flag A defines if a node is a master time source or not. Flag B defines how far from the master the node is. It has been selected so that after 2 hops the nodes are in the distributed mode even though the master time is available, but this value can be freely changed. Flag C is used to distribute the availability of the master time source within the network that is useful property during merging, as will be seen. Finally, preset flag D describes if master-slave links are allowed to use two-way delay compensation. If this is allowed, such a mechanism must be defined. Altogether, these take just seven bits, but allow many things.

**TABLE 1. The information flags of the NTS message.**

| NTS flag | Meaning | Comments |
|---|---|---|
| Flag A states | Master time in a node | |
| 1 | GNSS master time | |
| 2 | other master time | |
| 3 | no master time | initial value |
| 4 | | |
| Flag B states | Master-slave/distributed | |
| 1 | one hop from a master | |
| 2 | two hops from a master | |
| 3 | distributed mode | initial value |
| 4 | | |
| Flag C states | Master time in the network | Note 1 |
| 1 | GNSS master time | |
| 2 | other master time | |
| 3 | no master time | initial value |
| 4 | | |
| Flag D states | Delay compensation | Set in advance |
| 1 | one-way mode | initial value |
| 2 | two-way mode | |

Note 1: Indicates if the network has GNSS time or not, even if part of it is using the distributed mode.

The NTS protocol is shown in Fig. 3 and it is quite self-explanatory. However, a few details are explained. Before that an important observation must be mentioned. If a node receives NTS information from multiple master time sources, it has to decide which one to use. This may be simply based on the node ID. Interestingly, this phenomena allows also master



Master time (GNSS, ...)
Available?
YES
1) Use the master time
2) Set flag A
3) Set flag C
NO
Listen for & send NTS information
Received?
NO
YES
From a master?
YES
NO
From flag B, state 1?
YES
NO
Master-slave mode
Time update at the end of TSP:
- Use time from selected master node
1) If flag C set, ask the sender about delay compensation
2) Set flag B to state 1
3) set flag C to state 1 or 2
Time update at the end of TSP:
- Use time from selected node
1) If flag D set, ask the sender about delay compensation
2) Set flag B to state 2
3) set flag C to state 1 or 2
Distributed mode
Time update at the end of TSP:
- Use average of received times
1) Set flag B to state 3
2) Set flag C to state 1 or 2 if any message has it set
This mode does not execute two-way delay compensation.

**FIGURE 3.** The NTS protocol.

time integrity checking as proposed in [13] since times of the master time sources can be compared.

### 1) DELAY COMPENSATION

Delay compensation can be done in many ways and one alternative is shown here. Fig. 4 shows a master and a slave node that participate the process. First, the slave receives regular NTS information and then it initiates the delay compensation process. The shown clock times are local times. The clocks have time offset $O$ such that time in the slave node is master time $+O$, and propagation delay is $D$. Consequently, the slave receives a message from the master send at $T_1$ a bit later at $T_2 = T_1 + O + D$ and, consequently, the master receives message send by the slave at $T_3$ at $T_4 = T_3 - O + D$. Since the slave can read $T_1$ from the message, it can calculate $T_1 - T_2 = -O + D$. At time $T_3$ it transmits this information to the master that can calculate $T_3 - T_4 = 0 + D$, and subsequently also $(T_3 - T_4) + (T_1 - T_2) = 2D$, from which it can solve $D$ and transmit it to the slave. The delay compensation process thus includes two additional messages that, in principle, should be executed within the TSP. However, if mobility is low with respect to the requirements, it could take longer, and updates are not needed so often. In the distributed mode, it is possible to achieve delay compensation just by sending measured time errors $T_1 - T_2$ back to senders as will be shown in the analysis section IV.

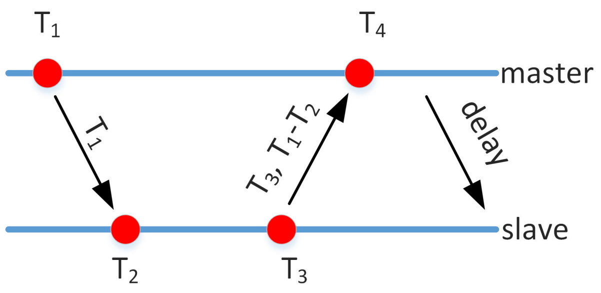


**FIGURE 4.** A two-way delay compensation method.

### 2) MERGING DISCUSSION

Merging is often overlooked when NTS is considered. However, if two subnetworks join, they have to decide which time to use. Naturally, it is possible to let the NTS protocol run as it is, but this may lead to splits, lost connections and so on, such that it is better to do merging in a coordinated way so that the whole subnetwork changes its time at the same moment. As such, a "merging needed" or "network time change at the moment T" message is needed, and the remaining task is to define how it is initiated and what are the decision rules for the time selection. The situation would be easy if just one node observed other subnetworks, but it cannot be ensured. Therefore, to avoid conflicting time change orders, a conflict resolution procedure is needed. Indeed, two processes are needed; one to decide the leading node of time change process within a subnetwork and another to decide the time sourcing subnetwork. It is assumed that in addition to the node IDs also subnetwork sizes are distributed in "hello" messages. It is emphasized that merging is not just a NTS process, but other processes are usually involved. Therefore, additional features or interprocess cooperation may be needed.

#### *a: DECIDE SUBNETWORK*

The decision process about which subnetwork is used as a time source is described in Fig. 5. The flag contents are compared. If the flag states are equal, the smaller subnetwork has to change its time and if the sizes are equal, the node ID is used to stop the process in an unambiguous way (e.g., the larger value). The process includes late-entry and may mean that a single GNSS timed node is forcing a bigger subnetwork to change its time. In practice, GNSS timed subnetworks will never be compared since they are readily synchronized.

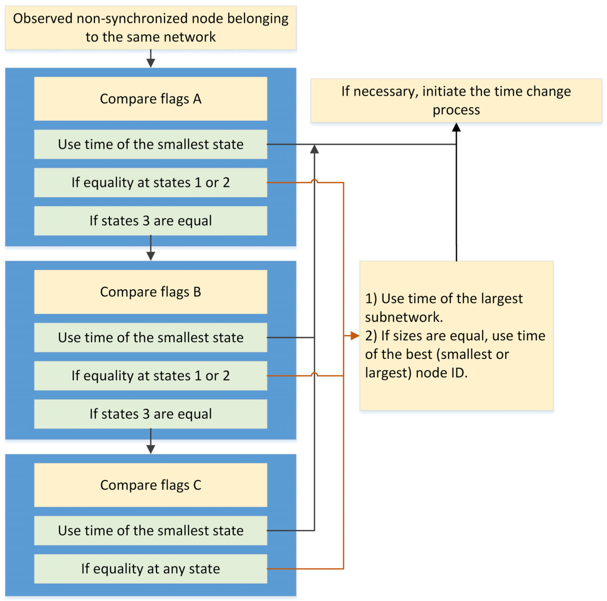

Observed non-synchronized node belonging to the same network
Compare flags A
Use time of the smallest state
If equality at states 1 or 2
If states 3 are equal
Compare flags B
Use time of the smallest state
If equality at states 1 or 2
If states 3 are equal
Compare flags C
Use time of the smallest state
If equality at any state
If necessary, initiate the time change process
1) Use time of the largest subnetwork.
2) If sizes are equal, use time of the best (smallest or largest) node ID.

**FIGURE 5. The process to decide the subnetwork whose time is used in the merging case.**

#### *b: DECIDE LEADER*

A straightforward solution for conflict resolution is to use an NTS conflict timer along with a broadcast message (within the subnetwork) about needed merging. The message contains the received NTS information to allow decision. The sending node is waiting until the timer elapses. If it will get only "go ahead" replies, it initiates and broadcasts the time change message. If it got "conflict" replies (along with other "merging needed" messages), it goes through a decision process. The process is exactly the same in all conflicting nodes since information has been shared in the broadcast messages. The process ends to an unambiguous conclusion about the node who initiates the time change process. The merging procedure in Fig. 5 is followed except that it is decided which node will initiate the time change message. To guarantee an unambiguous process, it could be required that a reply ("go ahead" or "conflict") from all nodes has to be received before the time change decision such that re-requests may be needed. Alternatively, the message is sent with the "time change request at the moment T" message, but all the nodes have to select which of these requests is used, if there are several. The decision is based on the unambiguous process described above.

One open issue is the handling of non-GNSS masters. This might be an issue if subnetworks have ENTSs containing nodes that are joined to different networks and therefore use different non-GNSS master time sources. The simplest solution is not to allow non-GNSS master time sources; see the ENTS section III-B for details about this.

### 3) NTS INFORMATION

As a summary, the needed NTS information is: i) Flags A, B, C, D, ii) Network ID (to see the cohesiveness, but also for the ENTS process), iii) Node ID, iv) Network size, v) Time. In addition, the possible two-way delay compensation requires additional information (and messages).

## *B. INTERNETWORK ENTS*

The multiradio nodes that have the ENTS algorithm share their existence to keep up a table of these ENTS nodes within the network. This ENTS table contains the ENTS nodes' IDs. Updates are not needed very often such that the ENTS update period could be rather long. The "network time change at the moment T" message introduced in the previous section is used to adopt possible new time within the networks such that new messages are not required.

One network could have more than just one ENTS node, and these may find different networks. Initially, just one radio network is active in the ENTS nodes in a network. Thereafter, one or more networks may become active in different ENTS nodes in the network. This means that these ENTS nodes have detected active networks and joined to them. However, these networks may have different times. Therefore, it is important to avoid time change conflicts and have possibility to make unambiguous decisions in all involved networks about what time will be used in the joint network.

The ENTS nodes that observe new networks inform the other ENTS nodes (within the network) that this network ID with this network size has been observed and a timing change process may be needed. At the same time, it starts an ENTS conflict timer. The replies are "go ahead" or "conflict". If all ENTS nodes do not reply (they are known due to the ENTS table) during the conflict timer, the request is resent until those "missing" nodes are doomed to be lost and the ENTS table updated accordingly.

If it gets "go ahead" replies, it means that there was just one time change request during the conflict timer period. The ENTS node decides the adopted time source based on the following order:

1) A GNSS time source is set as a master time for all non-GNSS timed networks.
2) Non-GNSS time sources are categorized as:
    a) time of the largest synchronized network size will be used*,
    b) if the sizes are equal, time of the network with the best network ID (e.g., the largest or the smallest) will be used.
    * Initially, the synchronized network size is the size of a network, but after two or more networks are synchronized involved ENTS nodes set the synchronized network size as the sum of network sizes.

If it gets "conflict" replies, this means that the same network has been observed at least twice, or different networks have been observed. In this case, the conflicting ENTS nodes can individually decide (without message exchange) that the ENTS node with the best ID (e.g., the smallest or the largest) will lead the process and the rest will do nothing. Once the decision is made, the ENTS node asks corresponding networks to change their time using their time change mechanism and set the NTS protocol flags accordingly. Note that this can be applied for different kind of NTS algorithms, not just the one described in this paper. The requirement is that they can handle an outside time source.

A summary of ENTS requirements:

1) The ENTS table and its updating once per the ENTS update period.
2) The ENTS conflict timer.
3) "New network observed, size X, ID Y" message to other ENTS nodes.
4) "go ahead" and "conflict" messages to the originating ENTS node(s).
5) Decision about adopted common time (in the best ENTS node).
6) Order corresponding networks to change their time.

The ENTS process can be used to synchronize networks even if they do not operate at the same frequency band. This is especially significant if a GNSS time is shared. However, it is less sure if a non-GNSS time should be shared in this case since the networks in different frequency bands are not interfering with each other.

Indeed, distributing a GNSS time always when it is available might be a policy set by the user organization. In addition, there might be other time-sharing policies that the ENTS protocol has to follow. However, distributing a GNSS time or any other time among networks that use different waveform is not always straightforward. The reason is that other waveform's NTS requirements and its time distribution mechanism may be too inaccurate for another waveform. As an example, a narrowband waveform has a GNSS timed node, but it is far away from the ENTS node such that its timing accuracy is not excellent even though it has a GNSS time source. This NTS accuracy may not be sufficient for a wideband waveform. Consequently, the wideband waveform should not adopt this GNSS time even though it would be a policy, since its master time source (the ENTS node) would be too inaccurate (time varies too much). Therefore, one must be careful when planning the rules for the ENTS protocol and these rules should include available and required NTS accuracies in different waveforms. If the GNSS time source was in the ENTS node, there would be no problems at all.

## IV. NTS ALGORITHM ANALYSIS

### *A. ALGORITHM*

The purpose is to correct the time readings of the clocks to be equal at different nodes, i.e., achieve a consensus about time. This is done adjusting the time of the clocks in the nodes by a correcting term $g_i(k)$ as

$$\begin{aligned} T_i(k) &\equiv T_i(k) + g_i(k) \\ &= T_0(k) + d_i(k), \end{aligned} \tag{2}$$

where $d_i(k) = t_i(k) + g_i(k)$ is the deviation from the common time. The adjustment term is a function of time comparisons between the local time in the node $i$ and received times with or without delay compensation. Obviously, the time deviation sequence obeys the recursion

$$d_i(k+1) = d_i(k) + g_i(k+1). \tag{3}$$

In what follows, the behavior of the deviation $d_i(k)$ is analyzed.

It is assumed that during the $k$th interval, a node $i$ has received $N_i(k) \leq N$ messages, where $N$ denotes the number of nodes. This means that the node $i$ has connections (in the synchronization sense) to $N_i(k)$ nodes, not necessarily to all nodes and the number of connections may be time varying. The connections need not be two way, i.e., if $i$ can receive a (NTS) message from $j$, $j$ does not need to be able to receive a message from $i$.

The estimated TOAs in the node $i$ are

$$D_{ij}(k) = T_0(k) + d_j(k) + \tau_{ij}(k) + n_{ij}(k), \tag{4}$$

where $\tau_{ij}$ is the propagation delay between nodes $i$ and $j$ and $n_{ij}$ is the time delivery error that includes both random clock reading accuracy at the transmitter and TOA accuracy in the receiver. Different nodes may have different noise variance, which is quite natural assumption since the master nodes usually have better time quality than other nodes have and they do not need to take care of measurement noise.

The timing information may be tied to the signal structure, e.g., to the start of time division multiple access (TDMA) slot or frequency hop. In that case $T_0(k) + d_j(k)$ is included in the time-of-arrival measurement. In the time stamping (clock sampling) case the value $T_0(k) + d_j(k)$ is read from the message.

The node $i$ subtracts its own time from these and compensates unknown propagation time by $\hat{\tau}_{ij}$. Naturally, in the uncompensated systems $\hat{\tau}_{ij} = 0$. The result is

$$\begin{aligned}\epsilon_{ij}(k) &= D_{ij}(k) - (T_i(k) + \hat{\tau}_{ij}(k))\\ &= d_j(k) - d_i(k) + \tau_{ij}(k) - \hat{\tau}_{ij}(k) + n_{ij}(k).\end{aligned} \quad (5)$$

The bias term $\triangle\tau_{ij}(k) = \tau_{ij}(k) - \hat{\tau}_{ij}(k)$ should be negligible in the delay compensated systems. Finally, the measurements are averaged such that the input to the correction term is

$$\epsilon_i(k+1) = \frac{1}{N_i(k)} \sum_{j=1}^{N_i(k)} \epsilon_{ij}(k). \quad (6)$$

Let $\mathbf{d}_k = [d_1(k) \ \ldots \ d_N(k)]^{\mathrm{T}}$ be the vector of time differences from the reference time. The measurements $\epsilon_i(k)$ at all nodes may be expressed in a vector $\boldsymbol{\epsilon}_k = [\epsilon_1(k) \ \ldots \ \epsilon_N(k)]^{\mathrm{T}}$ as

$$\boldsymbol{\epsilon}_{k+1} = \mathbf{C}_k\mathbf{d}_k + \boldsymbol{\tau}_k + \mathbf{n}_k, \quad (7)$$

where the diagonal elements of the connection matrix $\mathbf{C}(k)$ are -1's and the off diagonal elements $\mathbf{C}_{i,j}(k)$ are $1/N_i(k)$ if node $i$ is connected to node $j$ and zero otherwise. If a node $i$ is a master, the corresponding row in $\mathbf{C}(k)$ is all zeroes. Notice that the row sums of $\mathbf{C}(k)$ are zero. The vector $\mathbf{n}_k$ contains the measurement errors, i.e., $n_i(k) = \frac{1}{N_i(k)} \sum_{j=1}^{N_i(k)} n_{ij}(k)$. The vector $\boldsymbol{\tau}_k$ contains the effects of uncompensated delays $\triangle\tau_{ij}$.

The correction term $g_i(k) = h_i\epsilon_i(k)$ such that the deviation update becomes

$$\mathbf{d}_{k+1} = \mathbf{d}_k + \mathbf{H}\boldsymbol{\epsilon}_k, \quad (8)$$

where the diagonal matrix $\mathbf{H}$ has the coupling strengths $h_i$ as elements.

## *B. ANALYSIS*

This section provides the analytical results, where the starting point is the analysis with respect to the average consensus followed by novel mutual consensus analysis. The former analysis includes all the tools for the latter.

Since it is difficult to analyze the non-stationary case although some convergence results may be derived [11], the analysis is restricted to the stationary case wherein $\mathbf{C}_k = \mathbf{C}$. Even this restricted viewpoint provides valuable information for the behavior of the algorithms with bias and noise.

### 1) AVERAGE CONSENSUS VALUE

For simplicity, let $h_i = h$ such that the system has recursion

$$\begin{aligned}\mathbf{d}_{k+1} &= (\mathbf{I} + h\mathbf{C})\mathbf{d}_k + h\boldsymbol{\tau}_k + h\mathbf{n}_k\\ &= \mathbf{P}\mathbf{d}_k + h\big(\boldsymbol{\tau}_k + \mathbf{n}_k\big),\end{aligned} \quad (9)$$

where $\mathbf{I}$ is an identity matrix. The matrix $\mathbf{P} = \mathbf{I} + h\mathbf{C}$ may be called as a Perron matrix with parameter $h$ [11].

Assuming that $\mathbf{n}(t)$ are independent, identically distributed and zero mean variables, the properties of recursion (9) are well known [19]. If the bias is ignored for a moment, the expected value is $\mathbf{d}_k \xrightarrow{k\to\infty} \mathbf{P}^{\infty}\mathbf{d}_0$, where $\mathbf{d}_0$ is an initial value. If $0 < h < 1$, the consensus scheme is stable and the matrix $\mathbf{P}$ is a stochastic matrix [20]. Therefore, $\mathbf{d}_{k+1} = \mathbf{P}\mathbf{d}_k$ is a Markov chain. Stochastic matrices have interesting properties that are valuable in the analysis herein. First, stochastic matrices have an eigenvalue equal to one and, therefore, $\mathbf{P}^{\infty} \neq \mathbf{0}$. As a consequence, the consensus property cannot be proven trivially. To this end, observe that the distributed networks form primitive Markov chains and the master-slave or hybrid networks form reducible Markov chains [21].

Primitive Markov chains have a property that

$$\lim_{k\to\infty} \mathbf{P}^k = \mathbf{1}\mathbf{y}^{\mathrm{T}}, \quad (10)$$

where superscript T denotes transpose, $\mathbf{1}$ is the column vector of ones and $\mathbf{y}$ is the left eigenvector of $\mathbf{P}$ corresponding to eigenvalue one normalized such that $\mathbf{1}^{\mathrm{T}}\mathbf{y} = 1$ [20, Theorem 8.5.1]. This result means that all the rows of $\lim_{k\to\infty} \mathbf{P}^k$ are equal. Consequently, the distributed networks attain a consensus. They attain the limit (10) geometrically fast [22, Theorem 4.4.2]. If the network contains master nodes, the matrix $\mathbf{P}$ has form

$$\mathbf{P} = \begin{bmatrix}\mathbf{P}_1 & \mathbf{0}\\ \mathbf{R} & \mathbf{Q}\end{bmatrix}, \quad (11)$$

where $\mathbf{R} \neq \mathbf{0}$ and $\mathbf{P}_1$ is irreducible such that $\lim_{k\to\infty} \mathbf{P}_1^k = \mathbf{1}\mathbf{y}^{\mathrm{T}}$. It can be shown [22, Section 4.4.2] that

$$\lim_{k\to\infty} \mathbf{P}^k = \mathbf{1}[\mathbf{y}^{\mathrm{T}} \ \mathbf{0}]. \quad (12)$$

The result means that a consensus is attained also in the master-slave case. It also means that the time of masters is followed. The limit is attained geometrically fast. The sole master case is a special case with $\mathbf{P}_1 = 1$.

### 2) BIAS

The bias propagates as well. In general, it causes a sequence $\mathbf{P}^{k-1}\boldsymbol{\tau}_1 + \ldots \mathbf{P}\boldsymbol{\tau}_{k-1} + \boldsymbol{\tau}_k$, which is typically nonzero. In order to get more insight, let the bias be a constant. Then the previous sequence becomes $\left(\sum_{i=1}^{k-1} \mathbf{P}^i\right)\boldsymbol{\tau}$. Since $\mathbf{P}^i$ converges to a constant matrix, even a constant bias causes ever increasing deviation from the average consensus value. Similar behavior is expected from a non-constant bias.

If an instant bias, which is nonzero only for $k = 0$, is considered instead of continuous bias, it causes term $\mathbf{P}\boldsymbol{\tau}_0$, i.e., a deviation from the average consensus value but not ever increasing one.

### 3) COVARIANCE

The covariance of the Markov process is [19] $\mathbf{R}_k = \mathbf{P}^k\mathbf{R}_0(\mathbf{P}^k)^{\mathrm{T}} + h^2\sum_{n=1}^{k}\mathbf{P}^n\mathbf{Q}(\mathbf{P}^n)^{\mathrm{T}}$, where $\mathbf{Q}$ is the covariance of the noise process $\mathbf{n}(t)$ and $\mathbf{R}_0$ is the covariance of the initial uncertainty $\mathbf{d}_0$. In the mutual coupling networks all the rows of $\mathbf{P}^\infty$ are equal, say $\boldsymbol{\pi}$. Therefore, the elements of $\mathbf{P}^\infty\mathbf{Q}\mathbf{P}^{\infty\mathrm{T}}$ are $\boldsymbol{\pi}^{\mathrm{T}}\mathbf{Q}\boldsymbol{\pi}$. This means that the states are fully correlated. Therefore, a state change in a node causes corresponding change in the other nodes. In other words, the algorithm provides a consensus over the network, but that consensus value varies with an increasing magnitude around the average consensus value $\mathbf{P}^\infty\mathbf{d}_0$. Furthermore, the covariance will depend on the coupling strength $h$ and basically the larger it is, the larger is the covariance. However, also convergence rate depends on it such that trade-offs are needed.

### 4) MUTUAL CONSENSUS

Since the states are *fully correlated* it might be more sensible to compare the states of other nodes with the state of an arbitrary node, which in the master based systems should naturally be a master node. It is emphasized that the selection of the reference node does not affect the result, i.e., results cannot be improved by clever selection. The aim is just to show that nodes attain the mutual consensus with a limited variance although the variance increases with respect to the average consensus value as shown above.

Without the loss of generality, let the node 1 serve as the selected reference node. Then, the time difference to the reference node's time can be written as

$$\mathbf{r}_{k+1} = \mathbf{d}_{k+1} - \begin{bmatrix}-1 & \mathbf{0}\end{bmatrix}\mathbf{d}_{k+1} = \underbrace{\begin{bmatrix}0 & \mathbf{0}^{\mathrm{T}}\\ -\mathbf{1} & \mathbf{I}\end{bmatrix}}_{\mathbf{F}}\mathbf{d}_{k+1}. \quad (13)$$

As a consequence,

$$\mathbf{r}_{k+1} = \mathbf{F}\mathbf{P}\mathbf{d}_k + h\mathbf{F}\big(\boldsymbol{\tau}_k + \mathbf{n}_k\big). \quad (14)$$

Obviously, the mean propagates as

$$\mathbf{r}_{k+1} = \mathbf{F}\mathbf{P}^k\mathbf{d}_0 \quad (15)$$

and the covariance as

$$\mathbf{R}_k = \mathbf{F}\mathbf{P}^k\mathbf{R}_0(\mathbf{F}\mathbf{P}^k)^{\mathrm{T}} + h^2\sum_{n=1}^{k}\mathbf{F}\mathbf{P}^n\mathbf{F}\mathbf{Q}\mathbf{F}^{\mathrm{T}}(\mathbf{F}\mathbf{P}^n)^{\mathrm{T}}. \quad (16)$$

The convergence properties definitely depend on the matrix $\mathbf{F}\mathbf{P}^n$. It has already been shown that in all cases, see (10) and (12), $\mathbf{P}^n$ converges to $\mathbf{1}\mathbf{y}^{\mathrm{T}}$. Partition this as

$$\mathbf{1}\mathbf{y}^{\mathrm{T}} = \begin{bmatrix} y_1 & \mathbf{y}_L^{\mathrm{T}}\\ \mathbf{y}_1 & \mathbf{Y}_L\end{bmatrix}. \quad (17)$$

The elements of $\mathbf{y}_1$ are $y_1$ and the rows of $\mathbf{Y}_L$ are $\mathbf{y}_L$. It is straightforward to see that

$$\mathbf{F}(\mathbf{1}\mathbf{y}^{\mathrm{T}}) = \begin{bmatrix} 0 & \mathbf{0}^{\mathrm{T}}\\ \underbrace{-\mathbf{1}y_1 + \mathbf{y}_1}_{\mathbf{0}} & \underbrace{-\mathbf{1}\mathbf{y}_L^{\mathrm{T}} + \mathbf{Y}_L}_{\mathbf{0}}\end{bmatrix} \quad (18)$$

such that $\mathbf{F}\mathbf{P}^n$ converges to zero. Therefore, the sequence (14) converges to zero independent of the initial value and the covariance remains limited. In other words, it has been shown that the considered generic NTS algorithms attain a mutual consensus and the covariance around this common state remains limited.

It has already been shown that an initial bias is not a problem since the algorithms attain consensus (after convergence) for all initial values. However, the system may change its state after such an event since the steady-state value is not unique to all initial values. This is discussed also in [16]. Effects of a stationary bias are studied herein.

A stationary bias causes the recursion (14) to be

$$\mathbf{r}_{k+1} = \mathbf{F}\mathbf{P}\mathbf{d}_k + h\mathbf{F}\boldsymbol{\tau} + h\mathbf{F}\mathbf{n}_k. \quad (19)$$

As a result, a constant bias causes a state bias

$$\mathbf{b}'(t) = h\Big(\sum_{i=0}^{k}\mathbf{F}\mathbf{P}^i\Big)\boldsymbol{\tau} \quad (20)$$

to the $t$th state. It has been shown (18) that $\lim_{n\to\infty}\mathbf{F}\mathbf{P}^n = \mathbf{0}$. Therefore, the bias remains limited.

## C. DOUBLE-ENDED ALGORITHM

A constant bias may be compensated using a two-way or double-ended algorithm [8], [16]. In this scheme early measured state differences are distributed back to the origin. Since typically recent information is not available, node $i$ receives information $\epsilon_{ji}(t-m)$ at the moment $t$ from the other nodes. The node $i$ forms measurements

$$\begin{aligned}&\epsilon_{ij}(t) - \epsilon_{ji}(t-m)\\ &\quad = d_j(t) - d_i(t) + [d_j(t-m) - d_i(t-m)]\\ &\quad + \underbrace{\tau_{ij}(t) - \tau_{ji}(t-m)}_{\triangle\tau_{ij}(t,t-m)} + n_{ij}(t) - n_{ji}(t-m). \quad (21)\end{aligned}$$

As with the single-ended algorithm, these measurements are averaged in a node and then filtered, i.e., these averages replace $\epsilon_i(t)$ in the consensus algorithm (8). In addition, it is obvious that a constant bias is compensated in the double-ended algorithm since $\triangle\tau_{ij}(t, t-m)$ becomes zero.

If the connections are symmetric,

$$\boldsymbol{\epsilon}_k = \mathbf{C}\mathbf{d}_k + \mathbf{C}\mathbf{d}_{k-m} + \boldsymbol{\tau}_k + \mathbf{n}_k. \quad (22)$$

Consequently, the recursion becomes (without noise and bias)

$$\mathbf{d}_{k+1} = [\mathbf{I} + h\mathbf{C}]\mathbf{d}_k + h\mathbf{C}\mathbf{d}_{k-m}. \quad (23)$$

At the steady state this becomes

$$\mathbf{d} = [\mathbf{I} + 2h\mathbf{C}]\mathbf{d}. \quad (24)$$

This is similar result to that obtained for the single-ended system. However, now it is required that $0 < h < 0.5$ for the algorithm to be stochastic (stable). The covariance analysis would be more complicated than in the single-ended case. However, the results in the single-ended case explain the behavior of the double-ended algorithm.

### D. HIGHER ORDER FILTERS

So far, a zero order filter with a coefficient $h_i$ has been used though higher order filters would be possible too.

Let the deviation state, bias, noise and filters have z-transforms $\mathbf{D}(z)$, $\mathbf{B}(z)$, $\mathbf{N}(z)$ and $H_i(z)$. Then, after z-transforms and some rearranging, one obtains

$$\big[(z-1)\mathbf{I} - \mathbf{H}(z)\mathbf{C}\big]\mathbf{D}(z) = \mathbf{H}(z)[\mathbf{B}(z) + \mathbf{N}(z)], \qquad (25)$$

where $\mathbf{H}(z)$ is a diagonal matrix having the z-transforms $H_i(z)$ as its diagonal elements. Let the filter have feedback and feedforward parts, or $\mathbf{H}(z) = \mathbf{A}^{-1}(z)\mathbf{Q}(z)$. Assuming zero noise and bias in (25) results

$$\big[(z-1)\mathbf{A}(z) + \mathbf{Q}(z)\big]\mathbf{X}(z) = \mathbf{0}. \qquad (26)$$

Assuming second order filters $\mathbf{A}(z) = (a_0 z^2 + a_1 z + a_2)\mathbf{I}$, $(a_0 = 1)$ and $\mathbf{Q}(z) = (q_0 z^2 + q_1 z + q_2)\mathbf{I}$ and taking the inverse z-transform, (26) results in the temporal time deviation sequence

$$\begin{aligned}\mathbf{d}_{k+1} = \big[(a_0 - a_1)\mathbf{I} + q_0\mathbf{C}\big]\mathbf{d}_k + \big[(a_1 - a_2)\mathbf{I} \\ + q_1\mathbf{C}\big]\mathbf{d}_{k-1} + \big[a_2\mathbf{I} + q_2\mathbf{C}\big]\mathbf{d}_{k-2}.\end{aligned} \qquad (27)$$

At the consensus (if it is attained) the state remains constant between the adjustment instants and, therefore, the recursion (27) becomes the steady-state recursion

$$\begin{aligned}\mathbf{d} &= \big[(a_0 - a_1)\mathbf{I} + q_0\mathbf{C}\big]\mathbf{d} + \big[(a_1 - a_2)\mathbf{I} + q_1\mathbf{C}\big]\mathbf{d} \\ &\quad + \big[(a_2)\mathbf{I} + q_2\mathbf{C}\big]\mathbf{d}, \\ &= \big[a_0\mathbf{I} + (q_0 + q_1 + q_2)\mathbf{C}\big]\mathbf{d}.\end{aligned} \qquad (28)$$

Possible bias terms are: i) an instant bias like an initial offset $\delta(t)$, ii) a constant bias like an uncompensated delay $u(t)$, iii) a linear bias $tu(t)$ (a ramp) and iv) a higher order bias $t^2\,u(t)$, where $\delta(t)$ and $u(t)$ are the unit sample sequence (one at $t = 0$, zero otherwise) and the unit step signal (one for $t \geq 0$, zero otherwise). These have z-transforms 1, $z/(z-1)$, $z/(z-1)^2$ and $z(z+1)/(z-1)^3$, respectively. It has readily been shown that the bias i) can be compensated. If it would be possible to counter the bias ii), $Q(z)$ would be of form $z-1$, i.e., $q_0 = -q_1$. In this case the latter term in (28) vanish and the steady state solution becomes $\mathbf{x} = \mathbf{x}$. In other words, the algorithm would not converge at all. The same occurs also if the bias iii) is tried to counter since then $Q(z)$ would be of form $(z-1)^2 = z^2 - 2z + 1$, i.e., $q_0 + q_1 + q_2 = 0$. This result means that even though a filter in a single node can be adjusted to counter the bias ii) and iii), a network of such nodes cannot do that. However, as observed in [14], a bit of feedback could help to reduce the variance.

### E. ABOUT CLOCK FREQUENCY

Time delivery mechanism can be used to estimate clock frequencies, or skews. Indeed, the difference $\mathbf{s}_{k+1} = \mathbf{d}_{k+1} - \mathbf{d}_k$ divided by time duration between the events is the frequency estimate. Since the NTS algorithm makes $d_k$ a constant sequence, the difference $\mathbf{s}_k$ becomes zero, or that the NTS algorithm makes the virtual frequencies in the nodes equal. This is obvious since the NTS algorithms makes time epochs to start at the same moment and have equal duration, and these correspond to the phase and frequency of a clock.

### F. SUMMARY OF ANALYSIS RESULTS

It was shown that the investigated NTS schemes achieve mutual synchronization with a limited variance. However, these algorithms are prone to uncompensated delays and clock skews. It was shown that in those situations the well-known two-way schemes, which are insensitive to a constant bias, may be used. The two-way schemes can easily be adopted in practice since many systems have bidirectional communication links. It was also shown that the higher order filters cannot be used to reduce linear or higher order bias, though they may reduce variance. Furthermore, the lower bound for the accuracy is the time delivery accuracy. Uncompensated delays and skews decrease accuracy. In multihop master-slave chains the bias cumulates, i.e., the error is larger at the end of the chain than at the beginning. In general, the convergence rate of the algorithms is geometrically fast but the rate depends on the mode (centralized, distributed), topology (the number of nodes) and the number of connections and hops. Finally, it was shown that the NTS algorithm makes the virtual clock frequencies equal.

## V. SIMULATIONS

The simulations in this section illustrate what previous analysis findings could mean in practice. In the sake of generality, everything is expressed with respect to time delivery accuracy that is called the unit and it includes both time stamp accuracy (or transmit time accuracy) and time-of-arrival measurement accuracy. The unit can be converted to seconds and meters by selecting a sensible, investigated system dependent value for it. If a good GNSS time source and MHz class signal bandwidth are used, a unit could be about 100 ns (30 m in distance). Obtained results should be compared to required accuracy, that was considered in section III. If the numerical example is continued, 5 $\mu$s maximum allowed error corresponds to 50 units. In what follows, numerical values based on this example are presented in parenthesis after unit values to give a view what this could mean.

The NTS algorithm is otherwise as the described generic algorithm but flag B goes only one hop from the master. The clock skew is in the range $\pm 30$ units meaning that during the adjustment interval $T_A$ the skew $\beta$ causes $\pm 30$ units maximum time error ($\pm 3$ $\mu$s). If the clock has $\beta = 1$ ppm (pulse per million) accuracy, this means that the used adjustment period is 3 s. The maximum communication distance is 30 units (900 m according to the example) whereas the area is $60 \times 60$ units square (1.8 km by 1.8 km square). Results are shown for the first 200 adjustment periods and averaged over 100 Monte Carlo rounds where initial time offsets, time delivery errors and skews are varied randomly for a static, randomly formed node topology. Due to the short communication range, some nodes are not directly connected with the master nodes, i.e., there are multihop situations.

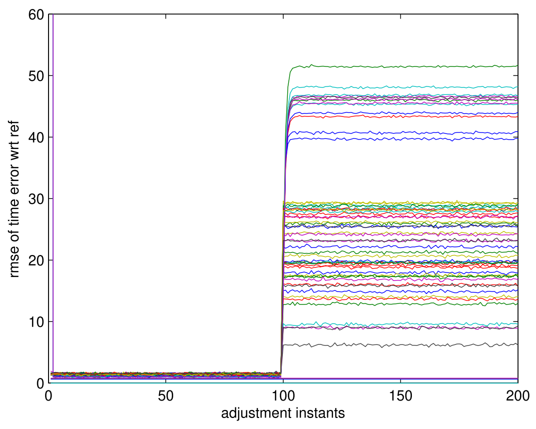


**FIGURE 6.** Root mean square errors (rmse) of individual nodes with respect to the reference node. Uncompensated delays start to affect at point 100.

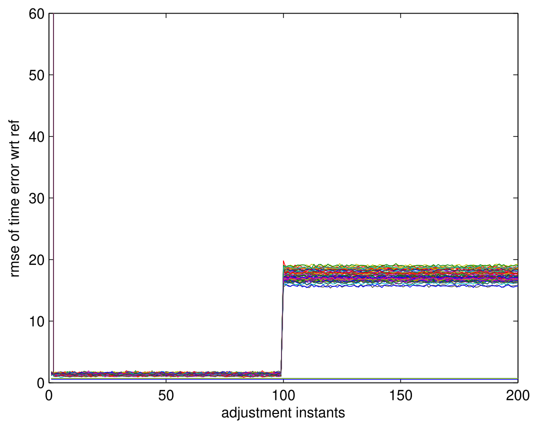


**FIGURE 7.** Root mean square errors (rmse) of individual nodes with respect to the reference node. Skews start to affect at point 100.

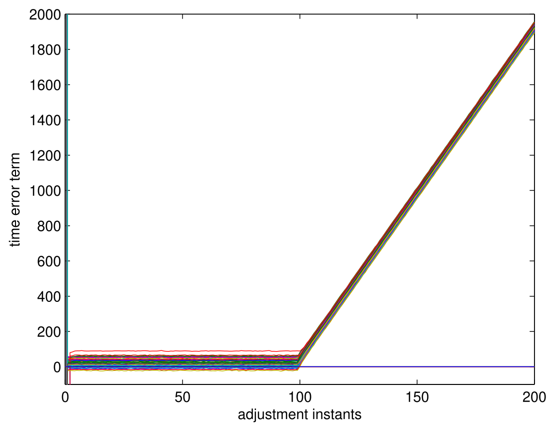


**FIGURE 8.** Time deviations of individual nodes when masters are lost at point 100.

The number of randomly deployed nodes is 64 of which 5 are master nodes. It is assumed that masters' clocks do not drift (i.e., they maintain their accuracy) but their initial time accuracy is $\pm 0.83$ units ($\pm 83$ ns). Furthermore, at each adjustment point their accuracy is 0.83 units. The other nodes are initially set within $\pm 6 \times 10^8$ units (one minute according to the example, quite good wristwatch accuracy). Their time reading accuracy (the variance of it) is one unit (by definition). Finally, the results are shown for all the nodes making figures somewhat fuzzy. This is done so because the interest is not on exact values but on trends like the convergence rate and that all time differences from a reference time stay limited.

In the first results, the skew and uncompensated delays are set on at point 100. This is done to demonstrate the ultimate accuracy (seen before point 100) and effects of those nonidealities (after point 100). In practice, nonidealities are usually present from the start. The results are with respect to a reference node, i.e., mutual consensus is considered. Naturally, one of the master nodes was selected as a reference.

The first results in Fig. 6 show effects of uncompensated delays. The ultimate limit is about 1.6 units, i.e., close to the time delivery accuracy that could be achieved if delays could be perfectly compensated. The convergence is after two rounds since there are two hops from a master to some nodes. Since some nodes are two hops from the masters, NTS errors larger than the maximum propagation range can be observed, as expected in the uncompensated delay case in the master-slave systems. And again, after point 100 the convergence to a stable state is very fast.

Clock skew effects are shown in Fig. 7. Skew obviously increases NTS error as predicted. The average error levels stay below the maximum skew (i.e. 30 units). Convergence is fast taking only two rounds.

In the next results, the skew and uncompensated delays are present from the start. First, it is demonstrated what occurs when all the masters are totally lost at point 100. The results in Fig. 8 show time errors from a single run since averaged results do not show the effect. Before point 100 the error is around the expected net error $\sqrt{20^2 + 50^2}$ coming from the skew and uncompensated delays shown in the previous case. After the point 100, the remaining non-GNSS timed nodes stay synchronized but their common time deviates from the global time (time error zero). As such, it has been shown that the proposed NTS protocol operates as it should and is robust against the lost of masters. In addition, it has been illustrated the fact that in the mutual coupling case the GNSS time is not necessarily followed. Simulations without lost of masters but, instead, with link failures after point 100 didn't show any visible effect when the link failure rate was 60%. This means that the algorithm, and especially the use of the previous control term therein, forms a very robust system. Naturally, if there is mobility such that relative distances of nodes change, then the use of the previous term may lead to increased error if mobility is larger or comparable to usual error (50 units now) during one or a few (if there are repeated fails) adjustment periods. Furthermore, if failures occur during the initialization phase, the convergence is slower.

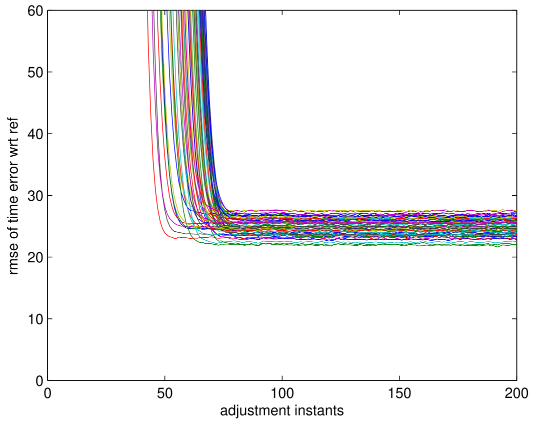


**FIGURE 9.** **Root mean square errors (rmse) of individual nodes with respect to the reference node in distributed mode.**

The final simulation results shown in Fig. 9 demonstrate the behavior of the distributed mode. First, it is observed that the convergence rate may be slow since it takes about 80 adjustment intervals. Consequently, large distributive synchronized networks call for speed up methods. Second, it is shown that the NTS error remains limited and that uncompensated delays are averaged away. This is so since the errors of all the nodes are at the same level though their distance from a selected reference node vary and could be up to two hops (compare to results in Fig. 6 where error levels vary depending on distance).

## VI. CONCLUSION

The paper introduced the internetwork time synchronization procedure and discussed its operation also in the challenging multipoint merging case. In addition, the paper presented a generic, highly flexible and robust NTS protocol for individual networks and showed its operation in the merging case. The protocol uses a precise time master, like a GNSS time, if that is available but otherwise it operates in the distributed mode. Novel NTS algorithm performance analysis was provided to show that in all modes the nodes are mutually synchronized, and the synchronization error remains limited. The provided simulations results confirmed these observations.

One identified future work topic was to develop efficient processes to handle non-GNSS masters in both NTS and internetwork time synchronization. One question is how to decide which source is used as the master in this case. Another future topic is to implement the algorithms using more detailed network level simulator that implements also other aspects such as medium access and routing functions, since NTS and ENTS functionalities can be integrated with their messages. It would be very interesting to study how this would be done in practice.

## REFERENCES

[1] H. Aissaoua, M. Aliouat, A. Bounceur, and R. Euler, “A distributed consensus-based clock synchronization protocol for wireless sensor networks,” *Wireless Pers. Commun.*, vol. 95, no. 4, pp. 4579–4600, Aug. 2017.

[2] J. A. Ansere, G. Han, and H. Wang, “A novel reliable adaptive beacon time synchronization algorithm for large-scale vehicular ad hoc networks,” *IEEE Trans. Veh. Technol.*, vol. 68, no. 12, pp. 11565–11576, Dec. 2019.

[3] S. Huang, N. Zheng, Y. Wu, and M. Xu, “Resilient consensus-based time synchronization in asynchronous sensor networks,” *IEEE Access*, vol. 7, pp. 115650–115661, 2019.

[4] X. Su, B. Hui, and K. Chang, “Multi-hop clock synchronization based on robust reference node selection for ship ad-hoc network,” *J. Commun. Netw.*, vol. 18, no. 1, pp. 65–74, Feb. 2016.

[5] W. Sun, E. G. Ström, F. Brännström, and M. R. Gholami, “Random broadcast based distributed consensus clock synchronization for mobile networks,” *IEEE Trans. Wireless Commun.*, vol. 14, no. 6, pp. 3378–3389, Jun. 2015.

[6] G. C. Copeland and S. Y. Seidel, “Synchronization of time in a mobile ad-hoc network,” U.S. Patent 8 300 615 B2. Accessed: May 3, 2021. [Online]. Available: https://patents.google.com/patent/US8300615B2/en

[7] K. Römer, P. Blum, and L. Meier, “Time synchronization and calibration in wireless sensor networks,” in *Handbook of Sensor Networks: Algorithms and Architechtures*, I. Stojmenovic, Ed. Hoboken, NJ, USA: Wiley, 2005.

[8] W. Lindsey, F. Ghazvinian, W. C. Hagmann, and K. Dessouky, “Network synchronization,” *Proc. IEEE*, vol. 73, no. 10, pp. 1445–1467, Oct. 1985.

[9] D. Mitra, “Network synchronization: Analysis of a hybrid of master-slave and mutual synchronization,” *IEEE Trans. Commun.*, vol. COM-28, no. 8, pp. 1245–1259, Aug. 1980.

[10] H. Stover, “Network timing/synchronization for defense communications,” *IEEE Trans. Commun.*, vol. COM-28, no. 8, pp. 1234–1244, Aug. 1980.

[11] R. Olfati-Saber, J. A. Fax, and R. M. Murray, “Consensus and cooperation in networked multi-agent systems,” *Proc. IEEE*, vol. 95, no. 1, pp. 215–233, Jan. 2007.

[12] S. S. Pereira and A. Pages-Zamora, “Mean square convergence of consensus algorithms in random WSNs,” *IEEE Trans. Signal Process.*, vol. 58, no. 5, pp. 2866–2874, May 2010.

[13] H. Saarnisaari and T. Vanninen, “Hybrid network synchronization for manets,” in *Proc. Mil. Commun. Inf. Syst. Conf. (MCC)*, 2012, pp. 1–6.

[14] H. Saarnisaari, “Analysis of a discrete network synchronization algorithm,” in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*, Atlantic City, NJ, USA, Oct. 2005, pp. 740–746.

[15] A. Davies, “Discrete-time synchronization of digital data networks,” *IEEE Trans. Circuits Syst.*, vol. CAS-22, no. 7, pp. 610–618, Jul. 1975.

[16] A. Davies, “Discrete-time synchronization of communications networks having variable delays,” *IEEE Trans. Commun.*, vol. COM-23, no. 7, pp. 782–785, Jul. 1975.

[17] K. F. Hasan, Y. Feng, and Y.-C. Tian, “GNSS time synchronization in vehicular ad-hoc networks: Benefits and feasibility,” *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 12, pp. 3915–3924, Dec. 2018.

[18] D. W. Allan, “Time and frequency (time-domain) characterization, estimation, and prediction of precision clocks and oscillators,” *IEEE Trans. Ultrason., Ferroelectr., Freq. Control*, vol. UFFC-34, no. 6, pp. 647–654, Nov. 1987.

[19] S. M. Kay, *Fundamendals of Statistical Signal Prosessing: Estimation Theory*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1993.

[20] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge, U.K.: Campbridge Univ. Press, 1985.

[21] D. Cox and H. Miller, *The Theory of Stochastic Processes*. London, U.K.: Methuen, 1970.

[22] E. Seneta, *Non-Negative Matrices: An Introduction to Theory and Applications*. London, U.K.: George Allen & Unwin, 1973.

**HARRI SAARNISAARI** (Senior Member, IEEE) received the Ph.D. degree from the University of Oulu, Finland, in 2000. He is currently an Adjunct Professor with the Centre for Wireless Communications (CWC) Research Group, University of Oulu, where he has worked since 1994. He is working on projects aiming to improve remote area connections using newest and future mobile cellular systems. His research interests include signal processing for receivers, spectrum sensing, *ad hoc* networks, and network synchronization.

**JUHO MARKKULA** (Graduate Student Member, IEEE) received the M.Sc. degree in telecommunications from the University of Oulu, Finland, in 2009. Since 2008, he has been working with the Centre for Wireless Communications, University of Oulu, where he currently works as a Research Scientist. He has worked in several research projects concerning military communications system developments. His research interests include smart grid communications, especially concerning LTE, wireless sensor networks, and low-power wide-area networks.

**TUOMAS PASO** received the M.Sc. degree in telecommunications engineering from the University of Oulu, Finland, in 2010, where he is currently pursuing the Ph.D. degree in telecommunications engineering with a focus on medium access control (MAC) protocols for wireless healthcare scenarios.

Since 2009, he has been working as a Research Scientist and the Project Manager of the Centre for Wireless Communications, University of Oulu. He has authored or coauthored 18 articles and conference papers and he holds three patents. His research interests include MAC and NET protocols and waveform design in general, MANETs, and wireless sensor networks. The main verticals he is involved in are healthcare/medical ICT and defense with an emphasis in tactical and other military communications.

Mr. Paso has served as a reviewer for several IEEE journals and conferences and as a Technical Program Committee Member for IEEE PIMRC, since 2014. He is a member of the European Telecommunications Standard Institute (ETSI) Technical Committee Smart Body Area Network (SmartBAN), in which he is currently a Rapporteur for three different work items.

**TIMO BRÄYSY** received the M.Sc. and Ph.D. degrees in physics from the University of Oulu, Finland, in 1991 and 2000, respectively. He has a wide experience in scientific research in space physics and has worked in satellite instrument and software development projects. He is currently a Postdoctoral Researcher with the Centre for Wireless Communications-Networks and System (CWC-NS), University of Oulu. His current research interest includes cognitive and autonomous wireless networks with special emphasis on constraints set by security and defense application areas.