The definite version is published in Proc. 7th International Symposium on Wireless Communications Systems (ISWCS) , September 2010.

Fault-Tolerant Averaging for Self-Organizing Synchronization in Wireless Ad Hoc Networks

Robert Leidenfrost
Elektrobit Austria GmbH
Kaiserstrasse 45/2
Austria, 1070 Vienna
Email: Robert.Leidenfrost@elektrobit.com

Wilfried Elmenreich and Christian Bettstetter
Mobile Systems Group and Lakeside Labs
University of Klagenfurt
Austria, 9020 Klagenfurt
Email: *firstname.lastname*@uni-klu.ac.at

***Abstract*— This paper presents a self-organizing robust clock synchronization algorithm based on the Reachback Firefly Algorithm, which is tailored for the use in wireless networks. We adapt a fault-tolerant algorithm from wired networks to cope with nodes deliberately feeding faulty clock readings into the system. The presented algorithm achieves a tight alignment of the firing phases of the non-faulty nodes, which supports duty cycling, communication scheduling, and time synchronization. Results show that the algorithm can cope with up to 1/5 non-silent faulty nodes.**

I. Introduction

Several algorithms for self-organizing time synchronization in wireless ad hoc networks have recently been proposed in the literature (e.g., [1]–[3]). All these algorithms assume that all nodes strictly behave according to the given rules, i.e., there are no faulty nodes. It is an open research issue as to how such self-organizing systems behave in the presence of faulty nodes and how to enhance given algorithms to make them more robust against different types of faults.

This paper addresses this problem by proposing an algorithm that is robust against strongly faulty nodes, namely Byzantine nodes. Whereas robustness against Byzantine faults is a well-studied issue in wired networks, it has received less attention in ad hoc networks. The original definition of a Byzantine fault is that a node can behave arbitrarily [4]. This also means that a group of Byzantine nodes may collude to perform the worst-case damage. The approach presented in this paper establishes a network-wide internal clock synchronization of an ad hoc network. In the scope of this work we consider faulty nodes to broadcast messages with a minimal inter-arrival time, but with the ability to corrupt an ongoing transmission of a neighboring node. The possibility of broadband jamming is not addressed herein.

Many existing synchronization algorithms require a structured network or are based on a centralized approach which makes them prone to different attacks.

Asynchronous Diffusion [5] and Reachback Firefly Algorithm (RFA) [1] are functional in unstructured and dynamic wireless networks. Other protocols (e.g., Synchronizer Ring (SR) [6], Secure Group Synchronization (SGS) [7]) provide robustness against compromised nodes, but are either not completely decentralized or cannot adapt to changing environments. A detailed elaboration and comparison among several protocols is performed in [8].

Wood *et al.* [9] elaborate the possibility of faulty nodes performing broadband jamming and propose approaches where the network is able to recognize and exclude the jammed area from routing and other communication-dependent functions.

Our approach is mainly based on the theory of self-stabilizing *pulse synchronization* [10] to achieve synchrony. RFA was figured out to be the most appropriate self-stabilizing algorithm with respect to this work. As shown in our previous work [2], the original RFA approach suffers from a worse synchronization precision in the order of the average message delay and further is not very robust against compromised nodes. In this paper, we overcome this disadvantage through a combination with a well-known distributed fault-tolerant clock synchronization protocol named Fault-Tolerant Averaging (FTA) [11].

II. System Model and Definitions

We model an ad hoc network using the *bounded delay model* in a *message-passing system* represented by a graph $G = (V, E)$ [12] containing a set of n processors or nodes p_i with $V = \{p_i \mid 0 \leq i < n\}$ where each p_i is capable of a broadcast primitive $broadcast_i(m)$. A node p_i broadcasts information to all neighboring nodes p_j over a bidirectional communication channel (i, j), if and only if $(i, j) \in E$, i.e., p_i and p_j are *neighbors*. The channel (i, j) can suffer from message loss due to collisions and interferences. The pattern of connections between the nodes is the *topology* of the system. Only connected topologies are treated in this paper. We denote the communication network, in short *network*, to be the collection of all connections in the system.

A broadcasted message m that is eventually delivered is assumed to have a delay $d(m)$ that is uniformly distributed in the range $d(m) \in [d, d + \varepsilon]$ where d defines the constant part of the delay and $0 \leq \varepsilon \ll d$ the delay jitter. We further neglect the processing time. Each node is equipped with a local drifting phase clock $\varphi_i(t)$ which suffers from some drift $|\rho_i| \leq \rho$ where ρ denotes the *maximum drift rate* among all hardware clocks in the system such that Def. 1 holds.

Definition 1 (Drifting phase clock)**.** *Let T be the nominal cycle duration in real-time. The phase clock $\varphi_j(t)$ of node p_j is a phase variable that has the following properties:*

1) $\varphi_j : t \mapsto [0, 1) = \varphi_j(t)$,
2) $\forall t_0, \Delta t$ *with* $0 < \Delta t \ll T \cdot (1 - \rho)$ *and* $\Delta\varphi = \varphi_j(t_0 + \Delta t) - \varphi_j(t_0) \neq 0$*:*
   - *if* $\Delta\varphi > 0$*:* $(1 + \rho)^{-1} \leq T \cdot \frac{\Delta\varphi}{\Delta t} \leq (1 - \rho)^{-1}$
   - *if* $\Delta\varphi < 0$*:* $(1 + \rho)^{-1} \leq T \cdot \frac{1 + \Delta\varphi}{\Delta t} \leq (1 - \rho)^{-1}$
3) *and* $\varphi_j = 0$ *at the beginning of a cycle.*

We assume that the cycle duration T is short enough such that the *drift variation* $|d\rho_i(t)/dt|$ for any p_i is negligible. That is, we have a *constant drift model*. The *state* $P(t)$ of a system comprising n nodes at real time t is defined as $P(t) = (\varphi_0(t), \varphi_1(t), \ldots, \varphi_{n-1}(t))$. A system is called *phase-synchronized* for a given maximum phase deviation Φ at real time t, if for all possible pairs of nodes $p_i, p_j \in N$, either $|\varphi_j(t) - \varphi_i(t)| \leq \Phi$, or $|\varphi_j(t) - \varphi_i(t)| \geq 1 - \Phi$.

We further say a node is *non-faulty* at time t if for time $t' \geq t$, it behaves according to its algorithm, has a drifting phase clock with a bounded drift, and has a negligible drift variation. A node that is faulty is assumed to be Byzantine faulty. A node is *perfect* if it is non-faulty and has no drift ($\rho = 0$). A network is *non-faulty* at time t if all broadcasted messages m at time $t' \geq t$ are received by the neighboring non-faulty nodes with a bounded delay of $d(m) \in [d, d + \varepsilon]$ time units. A network is *perfect* if it is non-faulty and for each message m, $d(m) = 0$. A system is *coherent* at time t if the network is non-faulty and there are at least $n - f$ non-faulty nodes where $f > 0$ represents our fault hypothesis which defines the maximum number of possibly faulty nodes such that the system is guaranteed to behave correct. A system is *fault-free* at time t if the network is non-faulty and all n nodes are non-faulty. A network is *k-connected*, if there exist at least k disjoint paths between any two non-faulty nodes. A network with diameter 1 is called *fully-meshed*.

III. Problem Statement

A system solves the self-stabilizing phase synchronization problem (in short SSPSP) if the following two conditions hold:

- **Convergence**: Starting from an arbitrary system state, the set of non-faulty nodes of a fault-free system reaches a phase-synchronized state after a finite amount of time.
- **Closure**: If $P(t_0)$ is a phase-synchronized state of the system at real-time t_0, then $\forall t \geq t_0$,
  1) the state $P(t)$ of a coherent system at real-time t is a phase-synchronized state, and
  2) each (non-faulty or faulty) node broadcasts at most one message in any interval $[t, t + T \cdot (1 + \rho)]$.

Informally, the goal of an algorithm solving the SSPSP is to reach a phase-synchronized state in a fault-free system which is then maintained even in a coherent system. Note that we sometimes say that a network achieved *synchrony* which means that it entered a phase-synchronized state. The second closure condition ensures that even faulty nodes do not broadcast more than one message during one cycle. This is feasible since we assume that faulty nodes do not perform jamming or Sybil attacks [13]. Nevertheless, we take into account that a message from a faulty node can disrupt a message from another node.

IV. Robust Synchronization

The robust synchronization approach is based on two algorithms, namely FTA [11] and a robust version of E-RFA [2]. First, the robust E-RFA (R-RFA) is presented and then FTA-RFA as a simple combination of R-RFA and FTA is devised.

A. Enhanced Reachback Firefly Algorithm (E-RFA)

E-RFA requires that every node p_i implements a phase clock φ_i and adheres to the following rules: 1) At the end of every cycle, p_i broadcasts a synchronization message m with some random offset containing its actual phase $\varphi_i(t)$ (*Pre-emptive message staggering*), and 2) based on the set of possible out-of-order gathered messages $M = \{m_1, m_2, \ldots, m_k\}$ during a complete cycle, p_i adjusts its clock through a *phase advance* Δ as a function of M and a pre-defined constant *coupling factor* α by setting the initial phase of the next cycle exactly to Δ (*Reachback response*). The random offset is assumed to be constrained within some pre-defined minimum and maximum relative *message staggering delay* r_{msd}^{min} and r_{msd}^{max}. Upper and lower bounds of these delays are presented in [2].

The calculation of $\Delta = f(M, \alpha)$ is presented in Alg. 1 in Line 9-14. For the case of two nodes, convergence of E-RFA is theoretically analyzed in [2]. Therein, also upper and lower bounds for α are presented. Simulation results for more than two nodes with different topologies also show convergence but without a formal proof.

B. Robust Reachback Firefly Algorithm (R-RFA)

R-RFA applies the fault acceptance mechanism of FTA: Before a node calculates the phase advance, it first removes the f lowest and f highest phase deviations from the message set M. This is illustrated in Alg. 1. In the case of a distributed system with a point-to-point communication, the assumption of $n \geq 3f + 1$ is adequate to guarantee convergence of FTA. However, our system model allows a faulty node to destroy the message broadcast of at most one non-faulty neighbor per cycle (e.g., through short-time radio jamming). Consequently, in the worst case, the f faulty nodes may always prevent f non-faulty neighbors from broadcasting their messages. Thus, if a node receives only $2f$ messages in the case of $n = 3f+1$, we cannot assume that all messages originate from non-faulty nodes. On this account, a node always has to remove the f lowest and f highest phase deviations independent of the number of received messages resulting in the fact that a $(5f + 1)$-connected network is required.

Note that the phase deviations are symmetric and lie within $[-\frac{1}{2}, \frac{1}{2}]$. On this account, we define the set S of *symmetric phase deviations* based on the received phase information in the message set M as

$$S(M) = \{s(\varphi_j) \mid \varphi_j \in M\} \tag{1}$$

where $s(\varphi_j)$ is the *symmetrization function* of φ_j with

$$s(\varphi_j) = \begin{cases} \varphi_j & \text{if } \varphi_j < \frac{1}{2} \\ \varphi_j - 2 & \text{if } \varphi_j \geq \frac{3}{2} \\ \varphi_j - 1 & \text{else} \end{cases}. \tag{2}$$

Let $j_{min} = \min_{\forall \varphi_j}\{j \mid s(\varphi_j) = \min(S)\}$ and $j_{max} = \max_{\forall \varphi_j}\{j \mid s(\varphi_j) = \max(S)\}$ be the smallest and greatest array indices of the subset of phase deviations that have the same minimum and maximum symmetric phase deviation. Note that the algorithm uses the array indices to refer to the distinct received phases. This does not mean that each node requires a unique identifier which defines our algorithm to be *anonymous*. Consequently, we define the following function:

$$reduce(M) = M \setminus \{\varphi_{j_{min}}, \varphi_{j_{max}}\}.$$

Algorithm 1: R-RFA: code $\forall p_i : 0 \leq i < n,\ n \geq 5f+1$

```
1  Init: M := ∅, Δ_i := 0, φ_i := 0,
   offset_i := random(r_msd^max − r_msd^min) + r_msd^min
2  upon event φ_i(t) = 1 − offset_i do                    // preponed tx
3      trigger broadcast_i(φ_i(t))                        // tx current phase
4  upon event recv_i(φ_j) from p_j do                     // rx sync-message
5      add (φ_i(t) + 1 − φ_j) to M
6  upon event φ_i(t) = 1 do                               // threshold reached
7      φ_last := δ_last := Δ_i := 0                       // clean up
8      reduce^f(M)                                        // introduce robustness
9      for each φ_j ∈ M in increasing order do            // E-RFA
10         if Δ_i + φ_j < 1 and φ_last + δ_last < φ_j then
11             δ_last := min(1, (φ_j + Δ_i) · α) − (φ_j + Δ_i)
12             Δ_i := Δ_i + δ_last
13             φ_last := φ_j
14     φ_i(t) := Δ_i                                      // Apply reachback response
15     offset_i := random(r_msd^max − r_msd^min) + r_msd^min
16     M := ∅
```

Algorithm 2: FTA-RFA: code $\forall p_i : 0 \leq i < n,\ n \geq 5f+1$

```
1  Init: M := ∅, Δ_i := 0, φ_i := 0,
   offset_i := random(r_msd^max − r_msd^min) + r_msd^min
2  upon event φ_i(t) = 1 − offset_i do                    // preponed tx
3      trigger broadcast_i(φ_i(t))                        // tx current phase
4  upon event recv_i(φ_j) from p_j do                     // rx sync-message
5      M := M ∪ {φ_i(t) + 1 − φ_j}
6  upon event φ_i(t) = 1 do                               // threshold reached
7      φ_last := δ_last := Δ_i := 0                       // clean up
8      ftaset := M ∪ {1}                                  // copy set for FTA concept
9      reduce^f(M)                                        // introduce robustness
10     dev_max := max(S(M))
11     dev_min := min(S(M))
12     dev := max(dev_max − dev_min, |dev_max|, |dev_min|)
13     if dev ≥ 1/L then                                  // execute E-RFA
14         for each φ_j ∈ M in increasing order do
15             if Δ_i + φ_j < 1 and φ_last + δ_last < φ_j then
16                 δ_last := min(1, (φ_j + Δ_i) · α) − (φ_j + Δ_i)
17                 Δ_i := Δ_i + δ_last
18                 φ_last := φ_j
19     else                                               // execute FTA
20         Δ_i := −avg(S(reduce^f(ftaset)))
21     φ_i(t) := Δ_i                                      // Apply reachback response
22     offset_i := random(r_msd^max − r_msd^min) + r_msd^min
23     M := ∅
```

Since we want to remove a set of lowest and highest deviations, we use $reduce^f$ to denote the f-fold iteration of the function $reduce$.

Our experimental studies [8] show that this approach likely converges in the presence of simple omission failures, if the parameters are correctly chosen according to the upper and lower bounds introduced in [2].

However, if the faulty nodes act in an adversary manner, they are always able to prevent the system from converging. For instance, consider the configuration in Fig. 1. Therein, a group of nodes is already synchronized and node p_i is outside the group. Assume that p_i has a higher drift (i.e., p_i is much faster) compared to the other nodes in the group and consequently diverges from the group. Assume $f = 1$ and let p_f be a Byzantine node which transmits a message to each node in the group in each round with the information as it would be situated exactly c phase units in front the group. Further let Δ_i be the phase advance performed by node p_i in some round. Then p_f can chose a different $c > 1/L$ for each node of the group such that the phase advance Δ_g of all these nodes is the same and equals Δ_i. This leads to the fact that the phase difference between each group node and p_i never changes over time and convergence is never achieved. However, even the closure condition does not hold in a simple

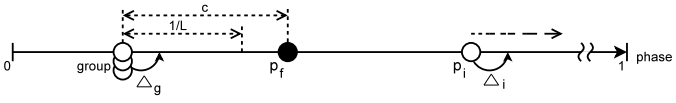


Fig. 1. Demonstration of a configuration where R-RFA will never converge in a coherent system for all admissible executions.

coherent system where all faulty nodes behave correct.

Corollary 1. *In any coherent system with a fully-meshed network comprising $n \geq 5f+1$ nodes for some $f > 0$ where all faulty nodes behave correct, Alg. 1 cannot maintain the closure condition of SSPSP. [8]*

Thus, if the algorithm maintains the convergence condition in a fault-free system, the nodes will periodically enter a phase-synchronized state, keep therein for some time, and then become unsynchronized until the fastest node again comes close to the other nodes.

C. Merging Fault-Tolerant Averaging and the Reachback Firefly Algorithm

To maintain the phase-synchronized state, we extended R-RFA by the FTA approach. The main advantage of the combination is that both algorithms calculate the clock adjustments solely on the message set M. Furthermore, we can reuse the formal results of FTA to determine the worst case precision. In other words, R-RFA provides convergence with a coarse synchronization precision in a fault-free system and FTA maintains the closure condition with a fine precision in a coherent system. Clearly, the coarse precision must be small enough to validate the assumption of initially synchronized nodes for maintaining synchrony with FTA, but must be large enough to have enough time until all nodes switched to the FTA approach. The inherent advantage of FTA compared to RFA is that the precision is in the order of the delay jitter and improves with an increasing number of nodes. In contrast, the worst case precision of E-RFA or R-RFA is about the maximum message delay and independent of the number of nodes.

Alg. 2 illustrates the cooperation of both approaches. Therein, we make use of the symmetrization function $s(\varphi_j)$ and the symmetrized set $S(M)$ as defined in Equ. (1). The switching condition in Line 13 depends on the maximum deviation a node identified and a pre-defined threshold value L named *FTA convergence threshold*. If the maximum deviation exceeds $1/L$, then the R-RFA approach is chosen. Otherwise, FTA is chosen. The parameter L is defined by the worst case scenario where the FTA approach may never converge. For this, we first define the term of a Basic Rest Circle (BRC) in graph G. Let $P(p_i, p_j)$ be the set of all paths from p_i to p_j with $p_i, p_j \in V$ and $p_i \neq p_j$, and $l(p)$ denotes the length of

a path or circle p. A BRC of $G = (V, E)$ is a closed simple path $C = (p_0, p_1, \ldots, p_k, p_0)$ that satisfies: $\forall 0 \leq i < j \leq k : \nexists p \in P(p_j, p_i) : l(p) < \min(j - i, k + 1 - (j - i))$. Let C_{max} denote the maximum BRC of G such that there is no other BRC C_i with $l(C_i) > l(C_{max})$.

Theorem 1. *Let C_{max} be the maximum BRC of a system. In the case $n \geq 5f+1$ for some $f > 0$, Alg. 2 may never converge if the convergence threshold $L \leq \max(l(C_{max}), 4)/2$ even if the f faulty nodes behave correct. [8]*

To incorporate the effect of inaccuracies and drift, we finally set the FTA convergence threshold to $L = \max(l(C_{max}), 4)$. However, in the next section we will see that there exists a tighter lower bound for L which was identified for ring topologies but also applies to all topologies which contain a maximum BRC C_{max} with $l(C_{max}) > 0$. Further bounds on L that increase the probability of reaching synchrony are devised in [8].

V. Performance

We use the probabilistic wireless sensor network simulator JProwler which is basically configured to simulate the behavior of the Berkeley Mica Motes with the B-MAC protocol [14]. Some MAC-specific attributes are modified according to Table I to simulate the behavior of an IEEE 802.15.4 MAC layer with uniformly distributed message delay $d(m) \in [2.2, 2.7]$ ms in the case a message m is delivered. The backoff scheme of CSMA/CA is deactivated due to its strong influence on the delay jitter. The configuration of the transmission power and noise is the same as originally implemented for the Mica2 motes and corresponds to a transmission range of about 25 m. JProwler simulates message collisions based on a calculated SINR. Every node p_i initially has a random phase and a constant random drift-rate $|\rho_i| \leq \rho$. Every simulation configuration is performed 100 times with a virtual time duration of at least 10000 s.

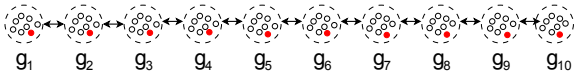


Fig. 2. Example of a grouped multi-hop topology (faulty nodes are depicted as filled red circles).

All simulations are based on a *grouped multi-hop network*, consisting of several groups ordered in a chain-like topology such that any two nodes within the same or neighboring group are neighbors. Figure 2 visualizes this topology with $k = 10$ groups g_i, $1 \leq i \leq k$, each having a group size of $g = 8$ nodes, and thus realizes an 8-connected network. To avoid a high rate of message collisions due to noise and interference, two neighboring groups are geographically situated 15 m apart. Nodes within the same group are situated at most 1 m apart. The parameters used for all simulations are shown in Table I. Other topologies and metrics are evaluated and discussed in [8].

A. Simulation of a Fault-Free System

Figure 3 compares the phase deviations (denoted as group spread) by the use of box plots between R-RFA and FTA-RFA in grouped multi-hop networks with different group sizes for a fault-free case ($f = 0$). All simulated systems entered a phase-synchronized state with, in average, 200 rounds (1 round equals 1 second in the simulation). Independent of the group size, there is a significant difference between the phase deviation for R-RFA and FTA-RFA. This is due to the fact that R-RFA is based on E-RFA [2], which cannot synchronize better than the maximum message delay in a single-hop network. As shown in [2], the worst case precision in the fault-free case of R-RFA in this network equals $\Pi = 9 \cdot 2.9$ ms. Figure 3(a) shows that this upper precision bound is never exceeded, even in the presence of many message omissions and collisions. The main advantage of R-RFA is the fact that the nodes typically perform very small clock adjustments (in our experiments in the order of some μs).

TABLE I
General Simulation Configuration

Parameter	Symbol	Value
Granularity		1 μs
Transmission time	t_{tx}	1 ms
Minimum tx waiting time	$d - t_{tx}$	1.2 ms
Delay jitter	ε	0.5 ms
Cycle duration	T	1 s
Maximum drift rate	ρ	100 ppm
Maximum phase deviation	Φ	0.01
Coupling factor	α	1.01
FTA convergence threshold	L	4

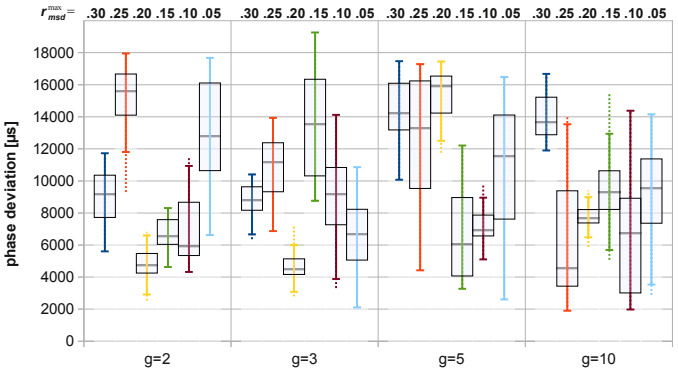
The convergence behavior of FTA-RFA (Fig. 3(b)) in this network equals that of R-RFA, but in contrast to R-RFA, FTA-RFA adjusts the nodes to the average of the received phase information after synchrony is reached. Since incoming messages are delayed by d and due to the fact that $\varepsilon \ll d$, every node performs large clock adjustments in the order of d, but globally the nodes will deviate only in the order of ε except in the vicinity of the beginning and at the end of a cycle. An increasing group size slightly improves the synchronization precision due to the nature of FTA, but increases the probability of message collisions until the precision degrades. However, in the presence of high message omissions and collisions, FTA-RFA is more robust than R-RFA with respect to the phase deviation.

B. Simulation of a Coherent System

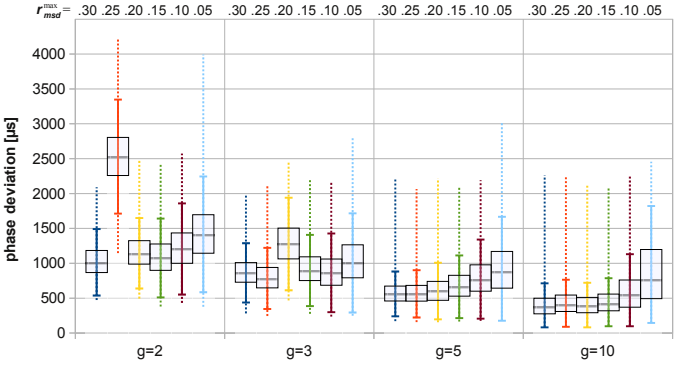
Faulty nodes now transmit different random values within special ranges to the distinct neighbors (two-faced malicious). Simulation results based on such a system yield estimations about the robustness of the applied algorithm.

Recall that FTA-RFA requires at least a $(5f + 1)$-connected network, which is fulfilled by a grouped multi-hop network with $g = 8$ for $f = 1$. If we assume that each node knows the exact number of maximum neighboring faulty nodes f_g per group, our definition of a grouped multi-hop network comprising k groups allows the presence of at most $k \cdot f_g$ faulty nodes (which may behave faulty only after convergence) with the constraint of at most f_g faulty nodes in each group. In contrast, $(5f + 1)$-connected networks are resilient to at most f faulty nodes. This means that in our topology example of one faulty node in each group ($f_g = 1$), all nodes of the border groups assume $f = 2$ and the remaining nodes assume $f = 3$.

Simulation results of FTA-RFA in such a grouped multi-hop topology are presented in Figure 4. Results regarding

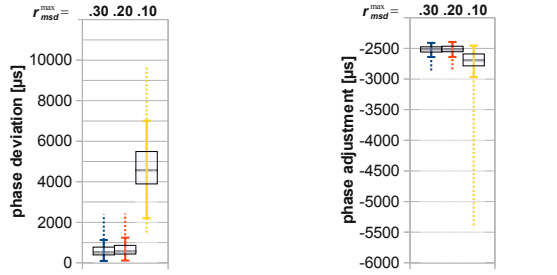


(a) R-RFA: Phase deviation diagram



(b) FTA-RFA: Phase deviation diagram

Fig. 3. Phase deviations in fault-free grouped multi-hop networks.



(a) Phase deviation diagram (b) Phase adjustment diagram

Fig. 4. Quality aspects of FTA-RFA in a grouped multi-hop network with one faulty node ($f = 1$) per group and 10 groups with group size $g = 8$.

$r_{msd}^{max} = 0.1$ are significantly worse than results for $r_{msd}^{max} = 0.3$ and $r_{msd}^{max} = 0.2$ due to the increased number of message omissions.

The average time to synchrony for the two other systems is about 250 rounds. The phase deviations never exceeded the upper bound for the worst case precision of FTA in the fault-free case ($9 \cdot 0.7\,\text{ms}$) [2]. The average clock adjustment is in the order of the average message delay (2.5 ms). The results show that FTA-RFA provides a robust convergence and a strongly improved precision even in the presence of two-faced malicious faulty nodes.

VI. Conclusions

In this paper we propose FTA-RFA, a self-organizing synchronization algorithm. Evaluations by simulation show that it works well in $(5f + 1)$-connected networks in the presence of at most f faulty nodes which may behave two-faced maliciously but are not assumed to perform radio jamming attacks or to collude in order to behave in an adversary manner. The presented approach is best suited for the use in networks suffering from significant communication delays but relatively small delay jitter and provides a high probability of achieving network-wide synchrony even in large multi-hop networks.

Future research will rely on the establishment of a robust self-organizing drift-correction algorithm. A node may also estimate the average message delay in order to combine the advantages of both FTA and RFA, namely an improved synchronization precision and small clock adjustment values.

Acknowledgment

This work was partly performed in the research cluster Lakeside Labs and was partly funded by the European Regional Development Fund, the Carinthian Economic Promotion Fund (KWF), and the state of Austria under grants 20214/17094/24770 (Triple-S) and 20214/18128/26673 (DEMESOS). We would like to thank Kornelia Lienbacher for proofreading the paper.

References

[1] G. Werner-Allen, G. Tewari, A. Patel, M. Welsh, and R. Nagpal, "Firefly-inspired sensor network synchronicity with realistic radio effects," in *SenSys '05: Proceedings of the 3rd international conference on Embedded networked sensor systems*. New York, NY, USA: ACM, 2005, pp. 142–153.

[2] R. Leidenfrost and W. Elmenreich, "Firefly clock synchronization in an 802.15.4 wireless network," *EURASIP J. Embedded Syst.*, vol. 2009, pp. 1–17, 2009.

[3] A. Tyrrell, G. Auer, and C. Bettstetter, "Emergent slot synchronization in wireless networks," *IEEE Trans. Mobile Comput.*, vol. 9, no. 5, pp. 719–732, May 2010.

[4] L. Lamport, R. Shostak, and M. Pease, "The Byzantine generals problem," *ACM Transactions on Programming Languages and Systems*, vol. 4, no. 3, pp. 382–401, Jul. 1982.

[5] Q. Li and D. Rus, "Global clock synchronization in sensor networks," *IEEE Transactions on Computers*, vol. 55, no. 2, pp. 214–226, 2006.

[6] K. Sun, "Fault-tolerant cluster-wise clock synchronization for wireless sensor networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 2, no. 3, pp. 177–189, 2005.

[7] S. Ganeriwal, C. Pöpper, S. Čapkun, and M. Srivastava, "Secure time synchronization in sensor networks," *ACM Transactions on Information and System Security*, vol. 11, no. 4, pp. 1–35, 2008.

[8] R. Leidenfrost, "Robust self-organizing pulse synchronization in wireless sensor networks," Master's Thesis, Vienna University of Technology, Vienna, Austria, 2009.

[9] A. D. Wood, J. A. Stankovic, and S. H. Son, "Jam: A jammed-area mapping service for sensor networks," in *RTSS '03: Proceedings of the 24th IEEE International Real-Time Systems Symposium*. Washington, DC, USA: IEEE Computer Society, 2003, p. 286.

[10] S. Dolev and J. L. Welch, "Self-stabilizing clock synchronization in the presence of Byzantine faults," *J. ACM*, vol. 51, no. 5, pp. 780–799, 2004.

[11] H. Kopetz and W. Ochsenreiter, "Clock synchronization in distributed real-time systems," *IEEE Transactions on Computers*, vol. C-36, no. 8, pp. 933–940, 1987.

[12] H. Attiya and J. L. Welch, *Distributed Computing: Fundamentals, Simulations and Advanced Topics (Second Ed.)*. McGraw-Hill, 2004.

[13] J. Douceur, "The sybil attack," in *IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems*. London, UK: Springer-Verlag, 2002, pp. 251–260.

[14] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in *SenSys '04: Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*. New York, NY, USA: ACM, 2004, pp. 95–107.