Theory Comput Syst
https://doi.org/10.1007/s00224-017-9840-3

Self-Stabilizing Byzantine Clock Synchronization with Optimal Precision

Pankaj Khanchandani[1] · **Christoph Lenzen**[2]

Abstract In the *Byzantine-tolerant clock synchronization* problem, the goal is to synchronize the clocks of n fully connected nodes. The clocks run at rates between 1 and $\vartheta > 1$, and messages have a delay (including computation) between $d - U$ and d. Moreover, up to $f < n/3$ of the nodes can fail by deviating arbitrarily from the protocol, i.e., are *Byzantine*. Despite this interference, correct nodes need to generate distinguished events (or *pulses*) almost simultaneously and periodically. The quality of the solution is measured by the *skew*, which is the maximum real time difference between corresponding pulses. In the *self-stabilizing* setting, in addition we allow for transient failures, possibly of all nodes. Once transient faults have ceased and at most f nodes remain faulty, the system should start generating synchronized pulses again. We design a self-stabilizing solution to this problem with asymptotically optimal skew. We achieve our goal by refining and extending the protocol of Lynch and Welch and make the following contributions in the process.

– We give a simple analysis of the Lynch and Welch protocol with improved bounds on skew and tolerable difference in clock rates by rebuilding upon the main ingredient of their protocol, called *approximate agreement*.

This article is part of the Topical Collection on *Special Issue on Stabilization, Safety, and Security of Distributed Systems (SSS 2016)*

✉ Christoph Lenzen
clenzen@mpi-inf.mpg.de

Pankaj Khanchandani
kpankaj@ethz.ch

[1] ETH Zurich, Zurich, Switzerland

[2] Max Planck Institute for Informatics, Saarland Informatics Campus, Saarbrücken, Germany

Published online: 20 January 2018

– We give a modified version of the protocol so that the frequency and amount of communication between the nodes is reduced. The modification adds a step to adjust the clock rates by another application of approximate agreement. The skew bound achieved is asymptotically optimal for suitable choices of parameters.
– We present a method to add self-stabilization to the above protocols while preserving their skew bounds. The heart of the method is a coupling scheme that leverages a self-stabilizing protocol with a larger skew.

Keywords Distributed algorithm · Stabilization time · Phase and frequency correction

1 Introduction

When designing a synchronous distributed system, the most fundamental question is how to generate a system clock at all the n nodes, i.e., how to periodically generate a distinguished event or *pulse* at each node so that the actual time of the i^{th} pulse at each node is close to the actual time of the i^{th} pulse of any other node. This *clock synchronization* problem is easily solved if each node is reliable and equipped with an accurate clock. However, neither is always the case. For instance, in space applications accurate clocks such as quartz oscillators are prone to failure, so less accurate electronic oscillators are preferable, and nodes are subject to radiation-induced transient faults. Thus, nodes have to frequently adjust their clocks by sending and receiving messages and executing a suitable algorithm. The inaccuracy in the clocks is modelled by assigning a clock *rate* or *frequency* that varies at each node, but within fixed bounds. We measure the precision of the algorithm by *skew*, which is the maximum over all pulses i and pairs of (correct) nodes of the time difference between the i^{th} pulses of the respective nodes.

The clock synchronization task is mission critical, both in terms of performance and reliability. Therefore, fault-tolerant distributed clock synchronization algorithms have found their way into real-world systems with high reliability demands. For example, the Time-Triggered Protocol (TTP) [13] and FlexRay [9, 10] tolerate *Byzantine* failure (i.e., arbitrary out-of-spec behavior) of less than $n/3$ nodes and are utilized in cars and airplanes. This means that these algorithms guarantee that correct nodes continue to generate synchronized pulses. They are based on the classic Byzantine clock synchronization algorithm by Lynch and Welch [19].

Another application domain with even more stringent requirements is hardware for spacecraft and satellites. Here, a reliable system clock is in demand despite frequent transient failure of any number of nodes due to radiation. The property to recover from an unknown state once the transient failures have stopped is known as *self-stabilization*. This is essential for the space domain, but also highly desirable in the systems utilizing TTP or FlexRay. This claim is supported by the presence of various mechanisms that monitor the nodes and perform resets in case of observed faulty behavior in both protocols. Thus, it is of interest to devise synchronization algorithms that stabilize on their own, instead of relying on monitoring techniques: these need to

be highly reliable as well, or their failure may bring down the system due to erroneous detection of or response to faults.

Thus, self-stabilizing Byzantine clock synchronization algorithms with small skew have critical and useful applications and, accordingly, have received significant attention in the past (e.g., [2, 8]). However, existing algorithms cannot achieve asymptotically optimal skew. Our key motivation and main goal is to build a self-stabilizing Byzantine clock synchronization algorithm with asymptotically optimal skew.

Our Contribution We achieve our main goal by building upon and extending the approach given by Lynch and Welch [19] to solve the Byzantine clock synchronization problem. The approach uses *approximate agreement* [7] repeatedly to adjust the time of the next clock pulse. In the process of achieving our goal, we make the following contributions.

1. We present a simplified analysis of the Lynch-Welch algorithm. We show that the algorithm converges to a steady-state error $E \in O((\vartheta - 1)d + U)$, where hardware clock rates are between 1 and ϑ and messages take between $d - U$ and d time to arrive at their destination. This works even for very inaccurate clocks: it suffices if $\vartheta \leq 1.1$, although the skew bound goes to infinity as ϑ approaches the critical value.[1] However, for, e.g., $\vartheta \leq 1.01$, Theorem 1 bounds the skew by $E(\vartheta, d, U) \leq 2.222(\vartheta - 1)d + 4.533U$.
2. We give a conceptually simple extension of the previous algorithm that, in addition to changing the (logical) clock values, also adjusts the clock rates using approximate agreement. If the clocks are sufficiently stable, i.e., the maximum rate of change ν of clock rates is sufficiently small, then we can significantly increase the nominal round length T and decrease the frequency of communication without substantially affecting skew. Concretely, if $\vartheta \leq 1.01$, $\max\{F, U\} \ll T$ (where nodes' clocks are initialized within F of each other), and $\max\{(\vartheta - 1)^2 T, \nu T^2\} \ll U$, it is possible to guarantee a skew of $O(U)$ (see Corollary 12 and subsequent explanation), which is asymptotically optimal.
3. We introduce a generic scheme that enables making either of these algorithms self-stabilizing. The scheme couples one of the above (non-stabilizing) algorithms with a self-stabilizing Byzantine clock synchronization algorithm of larger skew $2d$.[2] The coupled algorithm is both self-stabilizing and has the original smaller skew of the non-stabilizing algorithm (Theorem 4 and Theorem 5). The self-stabilizing Byzantine clock synchronization algorithm that we utilize is FATAL [4, 5], which already offers a suitable interface to our coupling mechanism.

[1] For comparison, the critical value in [19] is smaller than 1.025, i.e., we can handle a factor 4 weaker bound on $\vartheta - 1$. Non-quartz oscillators used in space applications, where temperatures vary widely, may have ϑ close to this value, cf. [1].

[2] All prior self-stabilizing algorithms have at least this skew. It should also be noted that d involves computational delay and turns out to be larger for FATAL, due to issues related to implementation.

On the technical side, the first two results require little innovation compared to prior work. However, it proved challenging to obtain simple algorithms that also achieve tight skew bounds. The effort spent was worthwhile for two reasons.

1. A prototype FPGA implementation [12] strongly indicates that these algorithms are also easy to implement in hardware.[3]
2. There is no mathematical analysis of a clock rate or frequency correction scheme in the literature that can be readily applied to yield accurate bounds for simple algorithms. We provide such a tailored analysis of our second algorithm.

To clarify the second point, we first note that the framework in [16, 17] does address frequency correction, but would require substantial specialization, including its mathematical analysis, to achieve good constants in the bounds. Second, the FlexRay algorithm also adjusts frequencies, but differs from our second algorithm in a crucial point. In order to avoid that the approximate agreement scheme is rendered ineffective because nodes reach the imposed limits on adjusting their frequency,[4] we add a correction slowly pulling back nodes' frequencies to the nominal rate. Without this provision, it is straightforward to construct executions in which, e.g., the majority of the nodes run too fast for another node to sufficiently adjust its clock rate to match their speed. This means that, in the worst case, FlexRay's frequency correction is futile.

In contrast to the above contributions, the coupling scheme we use to combine our non-stabilizing algorithms with the FATAL algorithm showcases a novel technique of independent interest. We leverage FATAL's clock "beats" to effectively (re-)initialize the synchronization algorithm we couple it to. Here, care has to be taken to avoid such resets from occurring during regular operation of the non-stabilizing algorithms, as this could result in large skews or even spurious clock pulses. The solution is a feedback mechanism that enables the synchronization algorithm to actively trigger the next beat of FATAL at the appropriate time. FATAL stabilizes regardless of how these feedback signals behave, while actively triggering beats ensures that all nodes pass the checks which, if failed, trigger the respective node being reset. While a specific interface is required from the stabilizing algorithm to permit this approach, it seems likely that most, if not all, self-stabilizing synchronization algorithms could be modified to provide it. Thus, we consider the technique a highly useful separation of the tasks to achieve small skews and to ensure (fast) stabilization.

Organization of the Paper After presenting related work and the model, we proceed in the order of the contributions listed above: simplified phase synchronization (Section 4), frequency synchronization (Section 5), and finally the coupling scheme adding self-stabilization (Section 6). Section 7 concludes the paper.

[3]The prototype implementation achieves 182 ps skew [12], which is suitable for generating a system clock.
[4]Constraining feasible clock rates is necessary to avoid that measurement errors result in clocks speeding up or slowing down arbitrarily over time.

2 Related Work

TTP [13] and FlexRay [9, 10] are both implemented in software (barring minor hardware components). This is sufficient for their application domain, in which synchronous communication between hardware components at frequencies in the megahertz range is required. Solutions fully implemented in hardware are of interest for two reasons. First, having to implement the full software abstraction dramatically increases the number of potential reasons for a node to fail – at least from the point of view of the synchronization algorithm. A slim hardware implementation is thus likely to result in a substantially higher degree of reliability of the clocking mechanism. Second, if higher precision of synchronization is required, the significantly smaller delays incurred by dedicated hardware make it possible to meet these demands.

Apart from these issues, the complexity of a software solution renders TTP and FlexRay unsuitable as fault-tolerant clocking schemes for VLSI circuits. The DARTS project [3, 11] aimed at developing such a scheme, with the goal of coming up with a robust clocking method for space applications. Instead of being based on the Lynch-Welch approach, it implements the fault-tolerant synchronization algorithm by Srikanth and Toueg [18]. Unfortunately, DARTS falls short of its design goals in two ways. On the one hand, the Srikanth-Toueg primitive achieves skews of $\Theta(d)$, which tend to be significantly larger than those attainable with the Lynch-Welch approach.[5] Accordingly, the operational frequency DARTS can sustain (without large communication buffers and communication delays of multiple logical rounds) is in the range of 100 MHz, i.e., about an order of magnitude smaller than typical system speeds. Moreover, DARTS is not self-stabilizing. This means that DARTS – just like TTP and FlexRay – is unlikely to successfully cope with high rates of transient faults. Worse, the rate of transient faults will scale with the number of nodes (and thus sustainable faulty nodes). For space environments, this implies that adding fault-tolerance without self-stabilization cannot be expected to increase the reliability of the system at all.

These concerns inspired a follow-up work called FATAL, which seeks to overcome the downsides of DARTS. From an abstract point of view, FATAL [4, 5] can be interpreted as another incarnation of the Srikanth-Toueg approach. However, FATAL combines tolerance to Byzantine faults with self-stabilization in $O(n)$ time with probability $1 - 2^{-\Omega(n)}$; after recovery is complete, the algorithm maintains correct operation deterministically. Like DARTS, FATAL and the substantial line of prior work on Byzantine self-stabilizing synchronization algorithms (e.g., [2, 8]) cannot achieve better clock skews than $\Theta(d)$. The key motivation for the present paper is to combine the better precision achieved by the Lynch-Welch approach with the self-stabilization properties of FATAL.

Concerning frequency correction, little related work exists. A notable exception is the extension of the interval-based synchronization framework to rate synchronization [16, 17]. In principle, it seems feasible to derive similar results by specialization

[5]The maximum delay d tends to be at least one or two orders of magnitude larger than the delay uncertainty U.

and minor adaptions of this powerful machinery to our setting. Unfortunately, apart from the technical hurdles involved, an educated guess (based on the amount of necessary specialization and estimates that need to be strengthened) results in worse constants and more involved algorithms, and it is unclear whether our approach to self-stabilization can be fitted to this framework. However, it is worth noting that the overall proof strategies for our (non-stabilizing) phase and frequency correction algorithms bear notable similarities to the generic framework: separately deriving bounds on the precision of measurements, plugging these into a generic convergence argument, and separating the analysis of frequency and phase corrections.

Coming to lower bounds and impossibility results, the following is known.

- impossibility results In a system of n nodes, no algorithm can tolerate $\lceil n/3 \rceil$ Byzantine faults. All mentioned algorithms are optimal in that they tolerate $\lceil n/3 \rceil - 1$ Byzantine faults [6].
- To tolerate this number of faults, $\Omega(n^2)$ communication links are required.[6] All mentioned algorithms assume full connectivity and communicate by broadcasts (faulty nodes may not adhere to this). Less well-connected topologies are outside the scope of this work.
- The worst-case precision of an algorithm cannot be better than $(1 - 1/n)U$ in a network where communication delays may vary by U [15]. In the fault-free case and with $\vartheta - 1$ sufficiently small, this bound can be almost matched (cf. Section 4); all variants of the Lynch-Welch approach match this bound asymptotically, granted sufficiently accurate local clocks.
- Trivially, the worst case precision of any algorithm is at least $(\vartheta - 1)T$ if nodes exchange messages every T time units. Moreover, a simple indistinguishability argument shows a lower bound of $(\vartheta - 1)d$, regardless of T. In the fault-free case, this is essentially matched by our phase correction algorithm as well.
- With faults, the upper bound on the skew of the algorithm increases by factor $1/(1-\alpha)$, where $\alpha \approx 1/2$ if $\vartheta \approx 1$. It appears plausible that this is optimal under the constraint that the algorithm's resilience to Byzantine faults is optimal, due to a lower bound on the convergence rate of approximate agreement [7].

Overall, the resilience of the presented solution to faults is optimal, its precision asymptotically optimal, and it seems reasonable to assume that there is little room for improvement in this regard. In contrast, no non-trivial lower bounds on the stabilization time of self-stabilizing fault-tolerant synchronization algorithms are known. Very recently, it has been shown that stabilization time $O(\log n)$ can be achieved, and that stabilization time polylog$n$ is possible with nodes broadcasting only polylog$n$ bits per time unit [14]. The same coupling strategy as presented in this work could be applied to these algorithms, achieving much faster overall stabilization.

[6] If a node has fewer than $2f + 1$ neighbors in a system tolerating f faults, it cannot distinguish whether it synchronizes to a group of f correct or f faulty neighbors.

3 Model

We assume a fully connected system of n nodes, up to $f := \lfloor (n-1)/3 \rfloor$ of which may be Byzantine faulty (i.e., arbitrarily deviate from the protocol). We denote by V the set of all nodes and by $C \subseteq V$ the subset of *correct* nodes, i.e., those that are not faulty.

Communication is by broadcast of "pulses," which are messages without content: the only information conveyed is when a node transmitted a pulse. Nodes can distinguish between senders; this is used to distinguish the case of multiple pulses being sent by a single (faulty) node from multiple nodes sending one pulse each. Note that faulty nodes are not bound by the broadcast restriction, i.e., may send a pulse to a subset of the nodes only. The system is semi-synchronous. A pulse sent by node $v \in C$ at (Newtonian) time $p_v \in \mathbb{R}_0^+$ is received by node $w \in C$ at time $t_{vw} \in [p_v + d - U, p_v + d]$; we refer to d as the *maximum message delay* (or, chiefly, delay) and to U as the *delay uncertainty* (or, chiefly, uncertainty).

For these timing guarantees to be useful to an algorithm, the nodes must have a means to measure the progress of time. Each node $v \in C$ is equipped with a hardware clock H_v, which is modeled as a strictly increasing function $H_v : \mathbb{R}_0^+ \to \mathbb{R}_0^+$. We require that there is a constant $\vartheta > 1$ such that the following holds for all times $t < t'$.

$$t' - t \leq H_v(t') - H_v(t) \leq \vartheta(t' - t)$$

In other words, the hardware clocks have bounded drift.[7] We remark that our results can be easily translated to the case of discrete and bounded clocks.[8] We refer to $H_v(t)$ as the *local time* of v at time t.

Executions are event-based, where an event at node v is the reception of a message, a previously computed (and stored) local time being reached, or the initialization of the algorithm. A node may then perform computations and possibly send a pulse. For simplicity, we assume that these operations take zero time; adapting our results to account for computation time is straightforward.

Problem A clock synchronization algorithm generates distinguished events or *clock pulses* at times $p_v(r)$ for $r \in \mathbb{N}$ and $v \in C$ so that the following conditions are satisfied for all $r \in \mathbb{N}$.

1. $\forall v, w \in C : |p_v(r) - p_w(r)| \leq e(r)$
2. $\forall v \in C : A_{\min} \leq p_v(r+1) - p_v(r) \leq A_{\max}$

The first requirement is a bound on the synchronization error between the r^{th} clock ticks; naturally, it is desired that $e(r)$ is as small as possible. The second requirement is a bound on the time between consecutive clock ticks, which can be translated to

[7] It is common to define the drift symmetrically, i.e., $(1-\rho)(t'-t) \leq H_v(t') - H_v(t) \leq (1+\rho)(t'-t)$ for some $0 < \rho < 1$. For $\rho \ll 1$ and $\vartheta \approx 1$, up to minor order terms this is equivalent to setting $\rho := (\vartheta - 1)/2$ and rescaling the real time axis by factor $1-\rho$. The one-sided formulation results in less cluttered notation.

[8] Discretization can be handled by re-interpreting the discretization error as part of the delay uncertainty. All our algorithms use the hardware clock exclusively to measure bounded time differences.

a bound on the frequency of the clocks; here, the goal is that $A_{\min}/A_{\max} \approx 1$. The *precision* of the algorithm is measured by the steady state error[9]

$$E := \lim_{r' \to \infty} \sup_{r \geq r'} \{e(r)\}\,.$$

3.1 Model for Frequency Correction Algorithms

In order for frequency corrections to be useful, we need to assume that hardware clock rates do not change faster than the algorithm can adjust to keep the effective frequencies aligned.

Accordingly, in Section 5, we additionally require that clock rates satisfy a Lipschitz condition as well. There, we assume that H_v is differentiable (for all $v \in C$) with derivative h_v, where h_v satisfies for $t, t \in \mathbb{R}_0^+$ that

$$|h_v(t') - h_v(t)| \leq \nu |t' - t| \tag{1}$$

for some $\nu > 0$. Note that we maintain the model assumption that hardware clock rates are close to 1 at all times, i.e., $1 \leq h_v(t) \leq \vartheta$ for all $t \in \mathbb{R}_0^+$.

3.2 Self-stabilization

An algorithm is self-stabilizing, if it (re)establishes correct operation from arbitrary states in bounded time. If there is an upper bound on the time this takes in the worst case, we refer to it as the stabilization time.

In Section 6, we will make use of a self-stabilizing pulse synchronization algorithm to "reset" the system from inconsistent initial states. Starting the analysis only from this point, we have a consistent labeling of the pulses (modulo some $M \in \mathbb{N}$) that is shared by all correct nodes. For this special case, we can still apply the above problem formulation (w.r.t. this labeling).

4 Phase Synchronization Algorithm

In this section, we give a basic algorithm for byzantine clock synchronization and show its guarantees in Theorem 1. The basic algorithm is a variant of the one by Lynch and Welch [19], which synchronizes clocks by simulating perpetual synchronous approximate agreement [7] on the times when clock pulses should be generated. We diverge only in terms of communication: instead of round numbers, nodes broadcast content-free pulses. Due to sufficient waiting times between pulses, during regular operation received messages from correct nodes can be correctly attributed to the respective round. In fact, the primary purpose of transmitting round numbers in the Lynch-Welch algorithm is to add recovery properties. Our technique for adding self-stabilization (presented in Section 6) leverages the pulse synchronization algorithm from [4, 5] instead, which requires to broadcast constant-sized messages only.

[9]Typically, $e(r)$ is a monotone sequence, implying that simply $E = \lim_{r \to \infty} e(r)$.

Before presenting the algorithm and its analysis in Sections 4.2 and 4.3, respectively, we revisit some basic properties of the approximate agreement technique [7]. The results in this section are derivatives of the ones from [7, 19], but adapting them to our setting and notation is essential for deriving our main results in Sections 5 and 6.

4.1 Properties of Approximate Agreement Steps

Abstractly speaking, the synchronization performs approximate agreement steps in each (simulated synchronous) round. In approximate agreement, each node is given an input value and the goal is to let nodes determine values that are close to each other and within the interval spanned by the correct nodes' inputs.

In the clock synchronization setting, there is the additional obstacle that the communicated values are points in time. Due to delay uncertainty and drifting clocks, the communicated values are subject to a (worst-case) perturbation of at most some $\delta \in \mathbb{R}_0^+$. We will determine δ later in our analysis of the clock synchronization algorithms; we assume it to be given for now. The effect of these disturbances is straightforward: they may shift outputs by at most δ in each direction, increasing the range of the outputs by an additive 2δ in each step (in the worst case).

Algorithm 1 describes an approximate agreement step from the point of view of node $v \in C$. When implementing this later on, we need to make use of timing constraints to ensure that (i) correct nodes receive each other's messages in time to perform the associated computations and (ii) correct nodes' messages can be correctly attributed to the round to which they belong. Figure 1 depicts how a round unfolds assuming that these timing constraints are satisfied.

Algorithm 1 Approximate agreement step at node $v \in \mathbf{C}$ (with synchronous message exchange)

1 // node v is given input value x_v;
2 broadcast x_v to all nodes (including self);
3 // if $w \in C$, the received value $\hat{x}_{wv} \in [x_w - \delta, x_w + \delta]$;
4 receive first value $\hat{x}_{wv}$ from each node w ($\hat{x}_{wv} := x_v$ if no message from w received);
5 $S_v \leftarrow \{\hat{x}_{wv} \mid w \in V\}$;
6 denote by S_v^k the k^{th} element of S_v w.r.t. ascending order;
7 $y_v \leftarrow \dfrac{S_v^{f+1} + S_v^{n-f}}{2}$;
8 **return** y_v;

Denote by $\vec{x}$ the $|C|$-dimensional vector of correct nodes' inputs, i.e., $(\vec{x})_v = x_v$ for $v \in C$. The *diameter* $\|\vec{x}\|$ of $\vec{x}$ is the difference between the maximum and minimum components of $\vec{x}$. Formally,

$$\|\vec{x}\| := \max_{v \in C}\{x_v\} - \min_{v \in C}\{x_v\}.$$

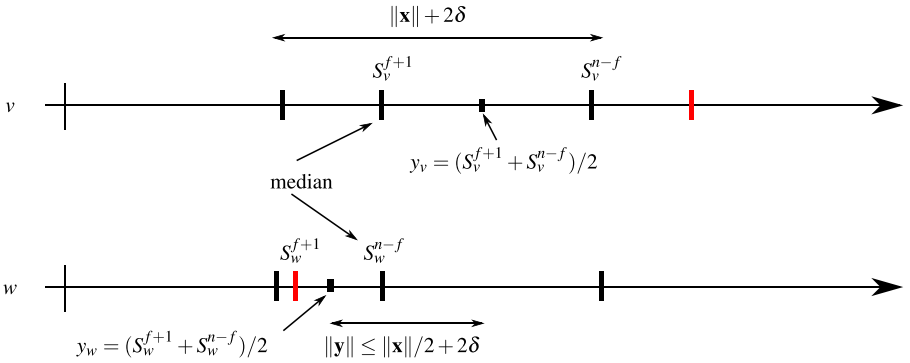

Fig. 1 An execution of Algorithm 1 at nodes v and w of a system consisting of $n = 4$ nodes. There is a single faulty node and its values are indicated in red. Note that the ranges spanned by the values received from non-faulty nodes are *almost* identical; the difference originates in the perturbations of up to δ

We will use the same notation for other values, e.g. $\vec{y}$ and $\|\vec{y}\|$. For simplicity, we assume that $|C| = n - f$ in the following; all statements can be adapted by replacing $n - f$ with $|C|$ where appropriate.

Consider the special case of $\delta = 0$. Intuitively, Algorithm 1 discards the smallest and largest f values each to ensure that values from faulty nodes cannot cause outputs to lie outside the range spanned by the correct nodes' values. Afterwards, y_v is determined as the midpoint of the interval spanned by the remaining values. Since $f < n/3$, i.e., $n - f \geq 2f + 1$, the median of correct nodes' values is part of all intervals computed by correct nodes. From this, it is easy to see that $\|\vec{y}\| \leq \|\vec{x}\|/2$, see Fig. 1. For $\delta > 0$, we simply observe that the resulting values y_v, $v \in C$, are shifted by at most δ compared to the case where $\delta = 0$, resulting in $\|\vec{y}\| \leq \|\vec{x}\|/2 + 2\delta$. We now prove these properties.

Lemma 1

$$\forall v \in C : \min_{w \in C}\{x_w\} - \delta \leq y_v \leq \max_{w \in C}\{x_w\} + \delta\,.$$

Proof As there are at most f faulty nodes, for $v \in C$ we have that

$$S_v^{f+1} \geq \min_{w \in C}\{\hat{x}_{wv}\} \geq \min_{w \in C}\{x_w\} - \delta\,.$$

Analogously, $S_v^{n-f} \leq \max_{w \in C}\{x_w\} + \delta$. We conclude that

$$\min_{w \in C}\{x_w\} - \delta \leq S_v^{f+1} \leq \frac{S_v^{f+1} + S_v^{n-f}}{2} = y_v \leq S_v^{n-f} \leq \max_{w \in C}\{x_w\} + \delta\,.$$

□

Corollary 1 $\max_{v \in C}\{|y_v - x_v|\} \leq \|\vec{x}\| + \delta$.

Lemma 2 $\|\vec{y}\| \leq \|\vec{x}\|/2 + 2\delta$.

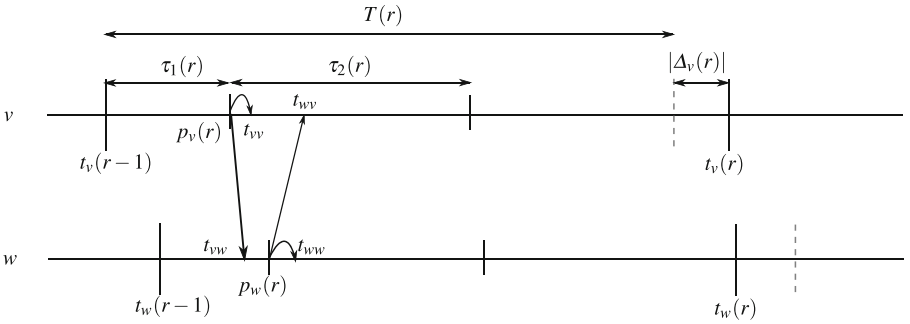

Fig. 2 A round of Algorithm 2 from the point of view of nodes v and w. Note that the durations marked on the horizontal axis are measured using the local hardware clock

Proof We show the claim for $\delta = 0$ first, i.e., $\hat{x}_{wv} = x_w$ for all $v, w \in C$. Denote by x^k the k^{th} element of $\vec{x}$ w.r.t. ascending order. Since $f < n/3$, we have that $n - f \geq 2f + 1$. Hence, for all $v \in C$,

$$x^1 \leq S_v^{f+1} \leq x^{f+1} \leq S_v^{2f+1} \leq S_v^{n-f} \leq x^{n-f} .$$

For any $v, w \in C$, it follows that

$$\begin{aligned} y_v - y_w &= \frac{S_v^{f+1} - S_w^{f+1} + S_v^{n-f} - S_w^{n-f}}{2} \\ &\leq \frac{x^{f+1} - x^1 + x^{n-f} - x^{f+1}}{2} = \frac{x^{n-f} - x^1}{2} \\ &= \frac{\|\vec{x}\|}{2} . \end{aligned}$$

Symmetrically, we have that $y_w - y_v \leq \|\vec{x}\|/2$ and thus $|y_v - y_w| \leq \|\vec{x}\|/2$. As $v, w \in C$ were arbitrary, this yields $\|\vec{y}\| \leq \|\vec{x}\|/2$ (under the assumption that $\delta = 0$).

For the general case, observe that S_v^{f+1}, S_w^{f+1}, S_v^{n-f}, and S_w^{n-f} each can be changed by at most δ. This can affect $(S_v^{f+1} - S_w^{f+1} + S_v^{n-f} - S_w^{n-f})/2$ by at most $4\delta/2 = 2\delta$; the claim follows. □

4.2 Algorithm

Algorithm 2 shows the pseudocode of the phase synchronization algorithm at node $v \in C$. It implements iterative approximate agreement steps on the times when to send pulses. The algorithm assumes that the nodes are initialized within a (local) time window of size F. In each round $r \in \mathbb{N}$, the nodes estimate the phase offset of their pulses[10] and then compute an according phase correction $\Delta_v(r)$. Figure 2 illustrates how a round of the algorithm plays out.

[10]Note that we divide the measured local time differences by factor $(\vartheta + 1)/2$, the average of the minimum and maximum clock rates. This is an artifact of our more notation-friendly "one-sided" definition of hardware clock rates from $[1, \vartheta]$; in an implementation, one simply reads the hardware clocks (which exhibit symmetric error) without any scaling.

Algorithm 2 Phase synchronization algorithm, code for node $v \in C$. Time $t_v(r)$, $r \in \mathbb{N}_0$, is the time when round $r+1$ starts

```
1  // H_w(0) ∈ [0, F) for all w ∈ V
2  wait until time t_v(0) with H_v(t_v(0)) = F;
3  foreach round r ∈ ℕ do
4      start listening for messages;
5      wait until local time H_v(t_v(r − 1)) + τ_1(r);        // all nodes are in round r
6      broadcast clock pulse to all nodes (including self);
7      wait until local time H_v(t_v(r − 1)) + τ_1(r) + τ_2(r);        // correct nodes' messages arrived
8      for each node w ∈ V do
9          τ_wv := H_v(t_wv), where first message from w received at t_wv (τ_wv := ∞ if none received);
10     S_v ← {2(τ_wv − τ_vv)/(ϑ + 1) | w ∈ V} (as multiset);
11     let S_v^k denote the k^th smallest element of S_v;
12     Δ_v(r) ← (S_v^{f+1} + S_v^{n−f}) / 2;
13     // T(r) denotes the nominal length of round r
14     wait until time t_v(r) with H_v(t_v(r)) = H_v(t_v(r − 1)) + T(r) − Δ_v(r);
```

To fully specify the algorithm, we need to determine how long the waiting periods in each round are (in terms of local time), which will be given as $\tau_1(r)$, $\tau_2(r)$, and $T(r) - \Delta(r) - \tau_1(r) - \tau_2(r)$. Here, we must ensure for all $r \in \mathbb{N}$ that

1. for all $v, w \in C$, the message that v broadcasts at time $t_v(r-1)+\tau_1(r)$ is received by w at a local time from $[H_w(t_w(r-1)), H_w(t_w(r-1)) + \tau_1(r) + \tau_2(r)]$ and
2. for all $v \in C$, $T(r) - \Delta_v(r) \geq \tau_1(r) + \tau_2(r)$, i.e., v computes $H_v(t_v(r))$ *before* time $t_v(r)$.

If these conditions are satisfied at all correct nodes, we say that *round r is executed correctly*, and we can interpret the round as an approximate agreement step in the sense of Section 4.1. We will show in the next section that the following condition is sufficient for all rounds to be executed correctly.

Condition 1 *Define* $e(1) := F + (1 - 1/\vartheta)\tau_1(1)$ *and inductively for all* $r \in \mathbb{N}$ *that*

$$e(r+1) := \frac{2\vartheta^2 + 5\vartheta - 5}{2(\vartheta + 1)} e(r) + (3\vartheta - 1)U + \left(1 - \frac{1}{\vartheta}\right)(T(r) + \tau_1(r+1) - \tau_1(r)) .$$

We require for all $r \in \mathbb{N}$ *that*

$$\begin{aligned} \tau_1(r) &\geq \vartheta e(r) \\ \tau_2(r) &\geq \vartheta(e(r) + d) \\ T(r) &\geq \tau_1(r) + \tau_2(r) + \vartheta(e(r) + U) . \end{aligned}$$

Here, $e(r)$ is a bound on the synchronization error in round r, i.e., we will show that $\|\vec{p}(r)\| \leq e(r)$ for all $r \in \mathbb{N}$, provided Condition 1 is satisfied. Condition 1

cannot be satisfied for arbitrary $\vartheta > 1$ such that $e(r)$ is bounded independently of r. The intuition is that rounds must be long enough to ensure that all pulses from correct nodes are received (i.e., at least $\vartheta e(r)$), but during this time additional error is built up by drifting clocks; if the approximate agreement step cannot overcome this *relative* skew increase, round $r+1$ has to be even longer, and so on. However, any $\vartheta \leq 1.1$ can be sustained.

Lemma 3 *Condition 1 can be satisfied such that* $\lim_{r\to\infty} e(r) < \infty$ *if*

$$\alpha := \frac{6\vartheta^2 + 5\vartheta - 9}{2(\vartheta+1)(2-\vartheta)} < 1\,.$$

In this case, we can achieve

$$\lim_{r\to\infty} e(r) = \frac{(\vartheta-1)d + (4\vartheta-2)U}{(2-\vartheta)(1-\alpha)}\,.$$

Proof By plugging $e(1)$ into the inequality for $\tau_1(1)$, we see that we may choose $\tau_1(1) < \infty$ if and only if $\vartheta < 2$. Assuming that this is the case, we choose to satisfy all inequalities with equality, yielding for $r \in \mathbb{N}$ that

$$\begin{aligned}
\tau_1(r) &= \vartheta e(r)\\
T(r) &= \vartheta(3e(r) + d + U)\\
e(r+1) &= \frac{6\vartheta^2 + 5\vartheta - 9}{2(\vartheta+1)(2-\vartheta)}\, e(r) + \frac{(\vartheta-1)d}{2-\vartheta} + \frac{(4\vartheta-2)U}{2-\vartheta}\\
&= \alpha e(r) + \frac{(\vartheta-1)d + (4\vartheta-2)U}{2-\vartheta}\,.
\end{aligned}$$

Thus,

$$\begin{aligned}
\lim_{r\to\infty} e(r) &= \lim_{r\to\infty}\left(\alpha^{r-1}e(1) + \sum_{r'=0}^{r-1}\alpha^{r'}\left(\frac{(\vartheta-1)d + (4\vartheta-2)U}{2-\vartheta}\right)\right)\\
&= \frac{(\vartheta-1)d + (4\vartheta-2)U}{(2-\vartheta)(1-\alpha)}\,,
\end{aligned}$$

where the second equality holds because $\alpha < 1$. Because $\alpha < 1$ is a stricter constraint on ϑ than $\vartheta < 2$, this completes the proof. □

Several remarks are in order.

- α goes to $1/2$ as ϑ goes to 1. For $\vartheta = 1.01$, we already have that $\alpha \approx 0.55$. Thus, the approach can support fairly large phase drifts.
- For $\vartheta \approx 1$, we have that $\lim_{r\to\infty} e(r) \approx 4U + 2(\vartheta-1)d$. From Corollary 2, one can see that if $(\vartheta-1)d \ll U$, this can be reduced to $\lim_{r\to\infty} e(r) \approx 2U$.
- The lower bound by Lynch and Welch [15] shows that this is optimal up to factor 2. It is straightforward to verify that in the fault-free case with $\vartheta = 1$, the algorithm attains the lower bound.
- The convergence is exponential, i.e., for any $\varepsilon > 0$ we have that $e(r) \leq (1+\varepsilon)\lim_{r\to\infty} e(r)$ for all $r \geq r_\varepsilon \in \Theta(\log F/(\varepsilon \lim_{r\to\infty} e(r)))$.

4.3 Analysis

In this section, we prove that Condition 1 is indeed sufficient to ensure that $\|\vec{p}(r)\| \leq e(r)$ for all $r \in \mathbb{N}$. In the following, denote by $\vec{p}(r)$, $r \in \mathbb{N}_0$, the vector of times when nodes $v \in C$ broadcast their r^{th} pulse, i.e., $H_v(p_v(r)) = H_v(t_v(r-1)) + \tau_1(r)$. If $v \in C$ takes note of the pulse from $w \in C$ in round r, the corresponding value $\tau_{wv} - \tau_{vv}$ can be interpreted as inexact measurement of $p_w(r) - p_v(r)$. This is captured by the following lemma, which provides precise bounds on the incurred error.

Lemma 4 *Suppose $v \in C$ receives the pulses from both $w \in C$ and itself in round r at a time from $[H_v(t_v(r-1)), H_v(t_v(r-1)) + \tau_1(r) + \tau_2(r)]$. Then*

$$\left| \frac{2(\tau_{wv} - \tau_{vv})}{\vartheta + 1} - (p_w(r) - p_v(r)) \right| < \vartheta U + \frac{\vartheta - 1}{\vartheta + 1} \|\vec{p}(r)\| ,$$

where τ_{wv} and τ_{vv} denote the values of the respective variables in the algorithm in round r.

Proof Denote by t_{uv} the time when v receives the pulse from $u \in \{v, w\}$. The communication model guarantees that $t_{uv} \in [p_u(r) + d - U, p_u(r) + d]$. Thus,

$$\tau_{uv} = H_v(t_{uv}) \in [H_v(p_u(r)+d-U), H_v(p_u(r)+d)] \subseteq H_v(p_u(r)+d-U/2) \pm \frac{\vartheta U}{2} . \tag{2}$$

Moreover, if $p_w(r) - p_v(r) \geq 0$, the bounds on the hardware clock speed guarantee that

$$\begin{aligned} \frac{2(p_w(r) - p_v(r))}{\vartheta + 1} &\leq \frac{2(H_v(p_w(r) + d - U/2) - H_v(p_v(r) + d - U/2))}{\vartheta + 1} \\ &\leq \frac{2\vartheta(p_w(r) - p_v(r))}{\vartheta + 1} \end{aligned}$$

and thus

$$\begin{aligned} & \frac{(1 - \vartheta)(p_w(r) - p_v(r))}{\vartheta + 1} \\ \leq\ & \frac{2(H_v(p_w(r) + d - U/2) - H_v(p_v(r) + d - U/2))}{\vartheta + 1} - (p_w(r) - p_v(r)) \\ \leq\ & \frac{(\vartheta - 1)(p_w(r) - p_v(r))}{\vartheta + 1} . \end{aligned}$$

Since $|p_w(r) - p_v(r)| \leq \|\vec{p}(r)\|$ by definition, this yields that

$$\begin{aligned} & \left| \frac{2(H_v(p_w(r) + d - U/2) - H_v(p_v(r) + d - U/2))}{\vartheta + 1} - (p_w(r) - p_v(r)) \right| \\ \leq\ & \frac{\vartheta - 1}{\vartheta + 1} \|\vec{p}(r)\| . \end{aligned} \tag{3}$$

This bound also holds in case $p_w(r) - p_v(r) < 0$, as we can switch the roles of v and w in the above inequalities. We conclude that

$$\begin{aligned}
&\left|\frac{2(\tau_{wv}-\tau_{vv})}{\vartheta+1} - (p_w(r)-p_v(r))\right| \\
&\quad\le \frac{2}{\vartheta+1}(|\tau_{wv} - H_v(p_w(r)+d-U/2)| + |\tau_{vv} - H_v(p_v(r)+d-U/2)|) \\
&\qquad + \left|\frac{2(H_v(p_w(r)+d-U/2) - H_v(p_v(r)+d-U/2))}{\vartheta+1} - (p_w(r)-p_v(r))\right| \\
&\quad\overset{(2),(3)}{<} \vartheta U + \frac{\vartheta-1}{\vartheta+1}\|\vec{p}(r)\|\,.
\end{aligned}$$

□

We remark that if $(\vartheta-1)d < U$ and U is known, it is beneficial to refrain from having v send a message to itself. Instead it estimates the arrival time of the message using its hardware clock, yielding the following corollary.

Corollary 2 *Suppose $v \in C$ receives the pulse from $w \in C$ in round r at a time from $[H_v(t_v(r-1)), H_v(t_v(r-1)) + \tau_1(r) + \tau_2(r)]$. Then*

$$\left|\frac{2(\tau_{wv} - H_v(p_v(r)))}{\vartheta+1} - \left(d-\frac{U}{2}\right) - (p_w(r)-p_v(r))\right| < \frac{\vartheta U}{2} + \frac{\vartheta-1}{\vartheta+1}(\|\vec{p}(r)\|+d)\,,$$

where τ_{wv} denotes the value of the respective variable in the algorithm in round r.

Proof By repeating the proof of Lemma 4, where the term $|\tau_{vv} - H_v(p_v(r)+d-U/2)|$ gets replaced by

$$\begin{aligned}
&\left|H_v(p_v(r)) + \frac{(\vartheta+1)(d-U/2)}{2} - H_v\left(p_v(r)+d-\frac{U}{2}\right)\right| \\
&\le \max\left\{\left|\frac{\vartheta+1}{2}-1\right|, \left|\frac{\vartheta+1}{2}-\vartheta\right|\right\}\left(d-\frac{U}{2}\right) \\
&= \frac{\vartheta-1}{\vartheta+1}\left(d-\frac{U}{2}\right) \\
&< \frac{\vartheta-1}{\vartheta+1}\,d\,.
\end{aligned}$$

□

In the sequel, we use the bounds provided by Lemma 4. However, the reader should keep in mind that in case $(\vartheta-1)d \ll U$ and sufficiently precise bounds on U are known, Corollary 2 shows how to effectively cut the influence of the uncertainty in half.

Using Lemma 4, we can interpret the phase shifts $\Delta_v(r)$ as outcomes of an approximate agreement step, yielding the following corollary.

Corollary 3 *Suppose in round $r \in \mathbb{N}$, it holds for all $v, w \in C$ that v receives the pulse from $w \in C$ and itself in round r during $[H_v(t_v(r-1)), H_v(t_v(r-1)) + \tau_1(r) + \tau_2(r)]$. Then*

1. $|\Delta_v(r)| < \vartheta(\|\vec{p}(r)\| + U)$ *and*
2. $\max_{v,w \in C}\{p_v(r) - \Delta_v(r) - p_w(r) + \Delta_w(r)\} \leq (5\vartheta - 3)\|\vec{p}(r)\|/(2(\vartheta + 1)) + 2\vartheta U$.

Proof By Lemma 4, we can interpret the values $2(\tau_{wv} - \tau_{vv})/(\vartheta + 1)$ as measurements of $p_w(r) - p_v(r)$ with error $\delta = \vartheta U + (\vartheta - 1)\|\vec{p}(r)\|/(\vartheta + 1)$. Note that shifting all values by $p_v(r)$ in an approximate agreement step changes the result by exactly $p_v(r)$, implying that $p_v(r) - \Delta_v(r)$ equals the result of an approximate agreement step with inputs $p_w(r)$, $w \in C$, and error δ at node v. Thus, the claims follow from Corollary 1 and Lemma 2, noting that $1/2 + 2(\vartheta - 1)/(\vartheta + 1) = (5\vartheta - 3)/(2(\vartheta + 1))$. □

To derive a bound on $\|\vec{p}(r + 1)\|$, it remains to analyze the effect of the clock drift between the pulses. To this end, we examine how an established timing relation between actions of two correct nodes deteriorates due to measuring time using the inaccurate hardware clocks.

Lemma 5 *Suppose $H_v(t'_v) - H_v(t_v) = h_v \geq 0$ and $H_w(t'_w) - H_v(t_w) = h_w \geq 0$. Then*

$$t_v - t_w + \frac{h_v}{\vartheta} - h_w \leq t'_v - t'_w \leq t_v - t_w + h_v - \frac{h_w}{\vartheta}\,.$$

Proof Since hardware clocks are increasing, $t'_v \geq t_v$ and $t'_w \geq t_w$. The inequalities follow because hardware clock rates are between 1 and $\vartheta \geq 1$. □

This readily yields a bound on $\|\vec{p}(r + 1)\|$ – provided that all nodes can compute when to send the next pulse on time.

Corollary 4 *Assume that round $r \in \mathbb{N}$ is executed correctly. Then*

$$\|\vec{p}(r + 1)\| \leq \frac{2\vartheta^2 + 5\vartheta - 5}{2(\vartheta + 1)}\|\vec{p}(r)\| + (3\vartheta - 1)U + \left(1 - \frac{1}{\vartheta}\right)T(r)\,.$$

Proof For $v, w \in C$, assume w.l.o.g. that $p_v(r+1) - p_w(r+1) \geq 0$. By Lemma 5 and Corollary 3, we have that

$$\begin{aligned}
&p_v(r+1) - p_w(r+1) \\
&\leq p_v(r) - p_w(r) + T(r) - \Delta_v(r) + \tau_1(r+1) - \tau_1(r) \\
&\quad - \frac{T(r) - \Delta_w(r) + \tau_1(r+1) - \tau_1(r)}{\vartheta} \\
&\leq p_v(r) - \Delta_v(r) - (p_w(r) - \Delta_w(r)) + \left(1 - \frac{1}{\vartheta}\right)(T(r) + \tau_1(r+1) - \tau_1(r) + |\Delta_w(r)|) \\
&\leq \frac{2\vartheta^2 + 5\vartheta - 5}{2(\vartheta+1)} \|\vec{p}(r)\| + (3\vartheta - 1)U + \left(1 - \frac{1}{\vartheta}\right)(T(r) + \tau_1(r+1) - \tau_1(r)) \,.
\end{aligned}$$

□

This bound hinges on the assumption that the round is executed correctly. We next establish sufficient conditions for this to be the case.

Lemma 6 *Suppose that*

$$\begin{aligned}
\tau_1(r) &\geq \vartheta(\|\vec{p}(r)\| - (d - U)) \\
\tau_2(r) &\geq \vartheta(\|\vec{p}(r)\| + d) \\
T(r) &\geq \tau_1(r) + \tau_2(r) + \vartheta(\|\vec{p}(r)\| + U) \,.
\end{aligned}$$

Then round r is executed correctly.

Proof Suppose $v, w \in C$. Denote by $t_{vw} \in [p_v(r) + d - U, p_v(r) + d]$ the time when this message is received by w. We have that

$$\begin{aligned}
t_{vw} &\geq p_v(r) + d - U \geq p_w(r) - \|\vec{p}(r)\| + d - U \\
&\geq t_w(r-1) + \frac{\tau_1(r)}{\vartheta} - (\|\vec{p}(r)\| - (d - U)) \\
&\geq t_w(r-1) \,,
\end{aligned}$$

showing that $H_w(t_{vw}) \geq H_w(t_w(r-1))$, i.e., w starts listening for the pulse of v on time. Similarly,

$$t_{vw} \leq p_v(r) + d \leq p_w(r) + \|\vec{p}(r)\| + d \leq p_w(r) + \frac{\tau_2(r)}{\vartheta} \,,$$

implying that $H_w(t_{vw}) \leq H_w(p_w(r)) + \tau_2(r) = H_w(t_w(r-1)) + \tau_1(r) + \tau_2(r)$. Thus, w receives the pulse from v before it stops listening, and the first requirement of correct execution of round r is met for all $v, w \in C$.

It remains to prove that for each $v \in C$, it holds that $T(r) - \Delta_v(r) \geq \tau_1(r) + \tau_2(r)$. By the preconditions of the lemma, this is satisfied if $\Delta_v(r) \leq \vartheta(\|\vec{p}(r)\| + U)$. As we already established the precondition of Corollary 3 for round r, the corollary shows that this inequality is satisfied. □

We have almost all pieces in place to inductively bound $\|\vec{p}(r)\|$ and determine suitable values for $\tau_1(r)$, $\tau_2(r)$, and $T(r)$. The last missing bit is an anchor for the induction, i.e., a bound on $\|\vec{p}(1)\|$.

Corollary 5 $\|\vec{p}(1)\| \leq F + (1 - 1/\vartheta)\tau_1(1) = e(1)$.

Proof Since $H_v(0) \in [0, F)$ for all $v \in C$, $t_v(0) \in [0, F)$ for all $v \in C$. The claim follows by applying Lemma 5. □

Theorem 1 *Suppose that Condition 1 is satisfied. Then, for all $r \in \mathbb{N}$, it holds that $\|\vec{p}(r)\| \leq e(r)$. If $\alpha = (6\vartheta^2 + 5\vartheta - 9)/(2(\vartheta + 1)(2 - \vartheta)) < 1$ (which holds for $\vartheta \leq 1.1$), we can choose the parameters such that the condition holds and Algorithm 2 has steady state error*

$$E = \lim_{r \to \infty} e(r) = \frac{(\vartheta - 1)d + (4\vartheta - 2)U}{(2 - \vartheta)(1 - \alpha)}.$$

Proof To show the first part, inductively use Lemma 6 and Lemma 4 to show that round r is executed correctly and that $\|\vec{p}(r + 1)\| \leq e(r + 1)$, respectively; the induction anchor is given by $\|\vec{p}(1)\| \leq e(1)$ according to Corollary 5. The second part directly follows from Lemma 3. □

5 Phase and Frequency Synchronization Algorithm

In this section, we extend the phase synchronization algorithm to also synchronize frequencies and give the guarantees of the extended algorithm in Theorem 3; a simplified statement is provided by Corollary 12. The basic idea is to apply the approximate agreement not only to phase offsets, but also to frequency offsets. To this end, in each round the phase difference is measured twice, applying any phase correction only after the second measurement. This enables nodes to obtain an estimate of the relative clock speeds, which in turn is used to obtain an estimate of the differences in clock speeds.

Ensuring that this procedure is executed correctly is straightforward by limiting $|\mu_v(r) - 1|$ to be small, where $\mu_v(r)$ is the factor by which node v changes its clock rate during round r. However, constraining this multiplier means that approximate agreement steps cannot be performed correctly in case $\mu_v(r + 1)$ would lie outside the valid range of multipliers. This is fixed by introducing a correction that "pulls" frequencies back to the default rate.

Of course, for all this to be meaningful, we need to assume that hardware clock rates do not change faster than the algorithm can adjust the multipliers to keep the effective frequencies aligned. We recall the additional model assumption stated in Section 3.1: we assume that H_v is differentiable (for all $v \in C$) with derivative h_v, where h_v satisfies for $t, t \in \mathbb{R}_0^+$ that $|h_v(t') - h_v(t)| \leq \nu|t' - t|$ for some $\nu > 0$.

5.1 Algorithm

Algorithm 3 Phase and frequency synchronization algorithm, code for node $v \in C$. Time $t_v(r)$, $r \in \mathbb{N}_0$, is the time when round $r+1$ starts

```
// H_w(0) ∈ [0, F) for all w ∈ V
wait until time t_v(0) with H_v(t_v(0)) = F;
// initialize clock rate multiplier
μ_v(0) := μ_v(1) := ϑ;
foreach round r ∈ ℕ do
    // phase correction step
    start listening for messages;
    wait until local time H_v(t_v(r − 1)) + τ_1/μ_v(r − 1);
    broadcast clock pulse to all nodes (including self);
    wait until local time H_v(t_v(r − 1)) + (τ_1 + τ_2)/μ_v(r);
    for each node w ∈ V do
        τ_wv := H_v(t_wv) (first message from w while listening at time t_wv;
        τ_wv := ∞ if none);
    S_v ← {2(τ_wv − τ_vv)/(ϑ + 1) | w ∈ V} (as multiset);
    let S_v^k denote the k^th smallest element of S_v;
    Δ_v(r) ← (S_v^{f+1} + S_v^{n−f})/2;
    // frequency correction step
    start listening for messages;
    wait until local time H_v(t_v(r − 1)) + (τ_1 + τ_2 + τ_3)/μ_v(r);
    broadcast clock pulse to all nodes (including self);
    wait until local time H_v(t_v(r − 1)) + (τ_1 + τ_2 + τ_3 + τ_4)/μ_v(r);
    for each node w ∈ V do
        τ'_wv := H_v(t'_wv) (first message from w while listening at time t'_wv;
        τ_wv := ∞ if none);
        Δ_wv := H_v(t'_wv) − H_v(t_wv);
    S_v ← {1 − μ_v(r)Δ_wv/(τ_2 + τ_3) | w ∈ V} (as multiset);
    let S_v^k denote the k^th smallest element of S_v;
    ξ_v(r) ← (S_v^{f+1} + S_v^{n−f})/2;
    μ̂_v(r + 1) ← μ_v(r) + 2ξ_v(r)/(ϑ + 1);
    // pull back towards nominal frequency by ε, ensure minimum and
    maximum rate
    if μ̂_v(r + 1) ≤ ϑ then
        μ_v(r + 1) ← max{μ̂_v(r + 1) + ε, 1};
    else
        μ_v(r + 1) ← min{μ̂_v(r + 1) − ε, ϑ^2};
    wait until time t_v(r) with H_v(t_v(r)) + (T − Δ_v(r))/μ_v(r);  // nominal
    round length is T
```

Algorithm 3 gives the pseudocode of our approach. Mostly, the algorithm can be seen as a variant of Algorithm 2 that allows for speeding up clocks by factors $\mu_v(r) \in [1, \vartheta^2]$, where $\vartheta h_v(t)$ is considered the nominal rate at time t.[11] For simplicity, we fix all local waiting times independently of the round length.

The main difference to Algorithm 2 is that a second pulse signal is sent before the phase correction is applied, enabling to determine the rate multipliers for the next round by an approximate agreement step as well. A frequency measurement is obtained by comparing the (observed) relative rate of the clock of node w during a local time interval of length $\tau_2 + \tau_3$ to the desired relative clock rate of 1. Since the clock of node v is considered to run at speed $\mu_v(r)h_v(t)$ during the measurement period, the former takes the form $\mu_v(r)\Delta_{wv}/(\tau_2 + \tau_3)$, where Δ_{wv} is the time difference between the arrival times of the two pulses from w measured with H_v. The approximate agreement step results in a new multiplier $\hat{\mu}_v(r+1)$ at node v; we then move this result by a (small) value ε in direction of the nominal rate multiplier ϑ and ensure that we remain within the acceptable multiplier range $[1, \vartheta^2]$.

To fully specify the algorithm, we need to determine how long the waiting periods are (in terms of local time) and choose ε. Here, we must ensure for all $r \in \mathbb{N}$ that

1. for all $v, w \in C$, the message v broadcasts at time $t_v(r-1) + \tau_1/\mu_v(r-1)$ is received by w at a local time from $[H_w(t_w(r-1)), H_w(t_w(r-1)) + \tau_1/\mu_v(r-1) + \tau_2/\mu_w(r)]$,
2. for all $v, w \in C$, the message v broadcasts at time $t_v(r-1) + \tau_1/\mu_v(r-1) + (\tau_2+\tau_3)/\mu_v(r)$ is received by w at a local time from $[H_w(t_w(r-1))+\tau_1/\mu_v(r-1) + \tau_2/\mu_w(r), H_w(t_w(r-1)) + \tau_1/\mu_v(r-1) + (\tau_2 + \tau_3 + \tau_4)/\mu_w(r)]$, and
3. for all $v \in C$, $T - \Delta_v(r) \geq \tau_1/\mu_v(r-1) + (\tau_2+\tau_3+\tau_4)/\mu_v(r)$, i.e., v computes $H_v(t_v(r))$ *before* time $t_v(r)$.

If these conditions are satisfied for $r \in \mathbb{N}$, we say that *round r was executed correctly.*

We now specify the constraints our choices for the parameters must satisfy to ensure that all rounds are executed correctly and both phase and frequency errors converge to small values.

Condition 2 *Set* $\bar{\vartheta} := \vartheta^3$. *Define*

$$e(1) := \max\left\{F + \left(1 - \frac{1}{\bar{\vartheta}}\right)\tau_1, \frac{(1 - 1/\bar{\vartheta})T + (3\bar{\vartheta} - 1)U}{1 - \bar{\beta}}\right\}$$

and, inductively for $r \in \mathbb{N}$,

$$e(r+1) := \frac{2\bar{\vartheta}^2 + 5\bar{\vartheta} - 5}{2(\bar{\vartheta} + 1)}\, e(r) + (3\bar{\vartheta} - 1)U + \left(1 - \frac{1}{\vartheta}\right) T\,.$$

[11] Given that hardware clock speeds may differ by at most factor ϑ, nodes need to be able to increase or decrease their rates by factor ϑ: a single deviating node may be considered faulty by the algorithm, so each node must be able to bridge this speed difference on its own.

We require that

$$\begin{aligned}
\tau_1 &\geq \bar{\vartheta} e(1) \\
\tau_2 &\geq \bar{\vartheta}(e(1) + d) \\
\tau_3 &\geq \bar{\vartheta}\left(e(1) + \left(1 - \frac{1}{\bar{\vartheta}}\right)(\tau_1 + \tau_2)\right) \\
\tau_4 &\geq \bar{\vartheta}\left(e(1) + d + \left(1 - \frac{1}{\bar{\vartheta}}\right)(\tau_1 + \tau_2)\right) \\
T &\geq \tau_1 + \tau_2 + \tau_3 + \tau_4 + \bar{\vartheta}(e(1) + U) \\
\varepsilon &\geq 2\left((\vartheta - 1)(\vartheta^3 - 1) + 2\vartheta^3\left(1 - \frac{1}{\vartheta^3}\right)^2 + \frac{2\vartheta^3 U}{\tau_2 + \tau_3} + 2(\vartheta^3 + 1)\nu T\right).
\end{aligned}$$

Here, all but the last conditions mimic Condition 1, where the bounds on τ_3 and τ_4 account for the fact that between the first and the second pulse of each round, the nodes' opinion on the "synchronized time" drift apart slowly. The lower bound on ε ensures that the pull-back of multipliers to the nominal ones is sufficiently strong to guarantee that, in fact, multipliers will never leave the valid range of $[1, \vartheta^2]$. We now show that these constraints can be satisfied provided that ϑ is not too large.

Lemma 7 *Condition 2 can be satisfied such that* $\lim_{r\to\infty} e(r) < \infty$ *if*

$$\bar{\alpha} := \bar{\beta} + (4\bar{\vartheta} + 3)(\bar{\vartheta} - 1) < 1\,,$$

where $\bar{\beta} := (2\bar{\vartheta}^2 + 5\bar{\vartheta} - 5)/(2(\bar{\vartheta} + 1))$. *Here, we may choose any* $T \geq T_0 \in O(F + d + U)$. *In this case,*

$$\lim_{r\to\infty} e(r) = \frac{(1 - 1/\bar{\vartheta})T + (3\bar{\vartheta} - 1)U}{1 - \bar{\beta}}\,.$$

Proof We choose τ_1, τ_2, τ_3, and τ_4 minimal such that the respective constraints are satisfied, and pick any feasible ε. Hence, the remaining constraints are that

$$T \geq \bar{\vartheta}((4\bar{\vartheta} + 3)e(1) + (2\bar{\vartheta} + 1)d + U) \tag{4}$$

and

$$e(1) = \max\left\{F + \left(1 + \frac{1}{\bar{\vartheta}}\right)e(1), \frac{(1 - 1/\bar{\vartheta})T + (3\bar{\vartheta} - 1)U}{1 - \bar{\beta}}\right\}.$$

Using that $2 - \bar{\vartheta} > 0$ (which is a weaker constraint than $\bar{\alpha} < 1$), assuming that $e(1)$ equals the first term of the maximum would yield that

$$e(1) = \frac{F}{2 - \bar{\vartheta}}\,,$$

and clearly there is a $T_0 \in O(F + d + U)$ such that (4) is satisfied for any $T \geq T_0$. Assuming that $e(1)$ equals the second term in the maximum, (4) becomes

$$T \geq \bar{\vartheta}\left((4\bar{\vartheta} + 3)\left(\frac{(1 - 1/\bar{\vartheta})T + (3\bar{\vartheta} - 1)U}{1 - \bar{\beta}}\right) + (2\bar{\vartheta} + 1)d + U)\right).$$

Using that $\bar{\alpha} < 1$, we can resolve this to

$$T \geq \bar{\vartheta} \cdot \frac{(4\bar{\vartheta}+3)(3\bar{\vartheta}+1)U + (1+\bar{\beta})((2\bar{\vartheta}+1)d+U)}{1-\bar{\alpha}} \in O(U+d)\,.$$

For the final claim, observe that by induction on r, we have that

$$\begin{aligned}\lim_{r\to\infty} e(r) &= \lim_{r\to\infty}\left(\bar{\beta}^{r-1}e(1) + \sum_{i=1}^{r-1}\bar{\beta}^{i-1}\left((3\bar{\vartheta}-1)U + \left(1-\frac{1}{\vartheta}\right)T\right)\right)\\ &= \frac{(1-1/\bar{\vartheta})T + (3\bar{\vartheta}-1)U}{1-\bar{\beta}}\,.\end{aligned}$$

□

5.2 Analysis

In the following, denote by $\vec{p}(r)$ and $\vec{q}(r)$, $r \in \mathbb{N}$, the vectors of times when nodes $v \in C$ broadcast their first and second pulse in round r, respectively. Thus, we have that $H_v(p_v(r)) = H_v(t_v(r-1)) + \tau_1/\mu_v(r-1)$ and $H_v(q_v(r)) = H_v(t_v(r-1)) + \tau_1/\mu_v(r-1) + (\tau_2+\tau_3)/\mu_v(r)$.

We will first make use of the analysis we performed for the phase correction algorithm to show that all rounds are executed correctly. Then we will refine the analysis by examining the impact of the frequency correction steps.

5.2.1 Phase Correction Steps

Observe that because for all $r \in \mathbb{N}_0$ and $v \in C$, we have that $1 \leq \mu_v(r) \leq \vartheta^2$, for all times t we have that $1 \leq \mu_v(r)h_v(t) \leq \vartheta^3 = \bar{\vartheta}$. Thus, we may interpret the waiting periods of Algorithm 3 as nodes waiting for τ_1, τ_2, etc. local time with hardware clocks of drift $\bar{\vartheta} = \vartheta^3$. Thus, we can make use of the same arguments as in Section 4.3 to obtain a series of results.

Corollary 6 *For all $r \in \mathbb{N}$, $\|\vec{q}(r)\| \leq \|\vec{p}(r)\| + (1-1/\bar{\vartheta})(\tau_1+\tau_2)$.*

Proof By application of Lemma 5. □

Corollary 7 *Suppose that*

$$\begin{aligned}\tau_1 &\geq \vartheta(\|\vec{p}(r)\| - (d-U))\\ \tau_2 &\geq \vartheta(\|\vec{p}(r)\| + d)\\ \tau_3 &\geq \vartheta(\|\vec{q}(r)\| - (d-U))\\ \tau_4 &\geq \vartheta(\|\vec{q}(r)\| + d)\\ T &\geq \tau_1+\tau_2+\tau_3+\tau_4+\vartheta(\|\vec{p}(r)\|+U)\,.\end{aligned}$$

Then round r is executed correctly.

Proof As for Lemma 6, where the pulse in the frequency correction step is analyzed analogously. □

Theorem 2 *Suppose that Condition 2 is satisfied and that*

$$\bar{\alpha} := \bar{\beta} + (4\bar{\vartheta} + 3)(\bar{\vartheta} - 1) < 1\,,$$

where $\bar{\beta} := (2\bar{\vartheta}^2 + 5\bar{\vartheta} - 5)/(2(\bar{\vartheta} + 1))$ *(this is the case for* $\vartheta \leq 1.011$*). Then, for all* $r \in \mathbb{N}$*, it holds that* $\|\vec{p}(r)\| \leq e(r)$ *and the algorithm has steady state error*

$$E \leq \frac{(1 - 1/\bar{\vartheta})T + (3\bar{\vartheta} - 1)U}{1 - \bar{\beta}}\,.$$

In particular, all rounds $r \in \mathbb{N}$ *are executed correctly.*

Proof As for Theorem 1, where we replace ϑ with $\bar{\vartheta}$, Lemma 6 with Corollary 7 and Lemma 3 with Lemma 7. However, the induction step requires that we can apply Lemma 6 again in step $r + 1$ if we could do so in step $r \in \mathbb{N}$. This readily follows from Condition 2 if $e(r + 1) \leq e(r)$ for all $r \in \mathbb{N}$.

We show this by induction on r. Abbreviate $x := (3\bar{\vartheta} - 1)U + (1 - 1/\bar{\vartheta})T$. Our claim is that (i) for $r \in \mathbb{N}$, $e(r) \geq x/(1 - \bar{\beta})$ and (ii) for $r \geq 2$, $e(r) \leq e(r - 1)$. The base case $r = 1$ requires (i) only, which holds by definition of $e(1)$. For the step from r to $r + 1$, we bound

$$e(r + 1) = \bar{\beta}e(r) + x \geq \frac{\bar{\beta}x}{1 - \bar{\beta}} + x = \frac{x}{1 - \bar{\beta}}$$

and

$$e(r) - e(r + 1) = (1 - \bar{\beta})e(r) - x \geq x - x = 0\,.$$

Finally, observe that our reasoning shows as part of the inductive argument that all rounds are executed correctly. □

5.2.2 Frequency Correction Steps

In the following, we assume that the prerequisites of Theorem 2 are satisfied. In particular, all rounds are executed correctly, i.e., we can assume that correct nodes receive each others' pulses. We introduce some notation to capture the behavior of the (logical) rates of the nodes' clocks. This notation may seem somewhat cumbersome; basically, the reader may think of the clock rates $h_v(t)$ as being almost constant, implying that all considered values for a given node $v \in C$ are essentially the same, slowly deviating at rate at most ν.

By $\vec{\rho}(r)$, we denote the vector whose entries are the intervals of clock rate ranges of nodes $v \in C$ between the first pulses in rounds $r \in \mathbb{N}$ and $r + 1$. Concretely,

$$\vec{\rho}(r)_v := \left[\min_{p_v(r) \leq t \leq p_v(r+1)} \{\mu_v(r)h_v(t)\}, \max_{p_v(r) \leq t \leq p_v(r+1)} \{\mu_v(r)h_v(t)\}\right].$$

By $\|\vec{\rho}(r)\|$, we denote the difference between maximum and minimum rate in $\vec{\rho}(r)$, i.e.,

$$\|\vec{\rho}(r)\| := \max_{v\in C} \max_{p_v(r)\le t\le p_v(r+1)} \{\mu_v(r)h_v(t)\} - \min_{v\in C} \min_{p_v(r)\le t\le p_v(r+1)} \{\mu_v(r)h_v(t)\}\,.$$

Furthermore, we denote by $\bar{\rho}(r)_v := \mu_v(r)h_v((p_v(r) + p_v(r+1))/2)$, by $\bar{\rho}(r)$ the respective vector, and by $\|\bar{\rho}(r)\| := \max_{v\in C}\{\bar{\rho}(r)\} - \min_{v\in C}\{\bar{\rho}(r)\}$. Note that $\bar{\rho}(r)_v \in \vec{\rho}(r)_v$ by definition.

We start by showing that $\bar{\rho}(r)_v$ approximates $\mu_v(r)h_v(t)$ well for times t between pulse r and $r+1$ of $v \in C$, i.e., we may see $\bar{\rho}(r)_v$ as "the" clock rate of v in round r.

Lemma 8 *Let $t \in [p_v(r), p_v(r+1)]$ for some $v \in C$ and $r \in \mathbb{N}$. Then*

$$|\mu_v(r)h_v(t) - \bar{\rho}(r)_v| < \nu\,\frac{T+\tau_2}{2}\,.$$

Proof Using that hardware clock rates are at least 1 and that $|\Delta_v(r)| < \max\{\tau_1, \tau_2\} = \tau_2$, we see that

$$\left|t - \frac{p_v(r+1)+p_v(r)}{2}\right| \le \frac{|p_v(r+1)-p_v(r)|}{2} \le \frac{|T-\Delta_v(r)|}{2\mu_v(r)} < \frac{T+\tau_2}{2\mu_v(r)}\,.$$

By our assumptions on the hardware clocks, this yields that

$$\begin{aligned}\left|\mu_v(r)\left(h_v(t) - h_v\left(\frac{p_v(r+1)+p_v(r)}{2}\right)\right)\right| &\le \mu_v(r)\cdot\nu\left|t - \frac{p_v(r+1)+p_v(r)}{2}\right| \\ &< \nu\,\frac{T+\tau_2}{2}\,.\end{aligned}$$

□

Two corollaries relate the progress of the hardware clocks between (i) $p_v(r)$ and $q_v(r)$ and (ii) t'_{wv} and t_{wv} to $\bar{\rho}(r)_v$, respectively.

Corollary 8 *For $v \in C$ and $r \in \mathbb{N}$, we have that*

$$|\bar{\rho}(r)_v(q_v(r) - p_v(r)) - (\tau_2+\tau_3)| < \nu T(\tau_2+\tau_3)\,.$$

Proof Let $\rho \in \vec{\rho}(r)_v$ such that $\rho(q_v(r) - p_v(r)) = \tau_2 + \tau_3$. By definition of $\vec{\rho}(r)_v$ and the mean value theorem, such a ρ exists and $\rho = \mu_v(r)h_v(t)$ for some $t \in [p_v(r), p_v(r+1)]$. By Lemma 8, $|\rho - \bar{\rho}(r)_v| < \nu T$. Thus,

$$\begin{aligned}|\bar{\rho}(r)_v(q_v(r) - p_v(r)) - (\tau_2+\tau_3)| &= |\rho - \bar{\rho}(r)_v|(q_v(r) - p_v(r)) \\ &= |\rho - \bar{\rho}(r)_v|\,\frac{\tau_2+\tau_3}{\rho} \\ &< \nu T(\tau_2+\tau_3)\,.\end{aligned}$$

□

Corollary 9 *For $v, w \in C$ and $r \in \mathbb{N}$, we have that*

$$|\mu_v(r)(H_v(t'_{wv}) - H_v(t_{wv})) - \bar{\rho}(r)_v(t'_{wv} - t_{wv})| < \nu T(\tau_2 + \tau_3)\,.$$

Proof Let $\bar{\rho} \in \vec{\rho}(r)_v$ such that $t'_{wv} - t_{wv} = \mu_v(r)(H_v(t'_{wv}) - H_v(t_{wv}))$. By definition of $\vec{\rho}(r)_v$ and the mean value theorem, such a ρ exists and $\rho = \mu_v(r)h_v(t)$ for some $t \in [t_{wv}, t'_{wv}] \subseteq [p_v(r),\, p_v(r+1)]$. By Lemma 8, $|\rho - \bar{\rho}(r)_v| < \nu(T + \tau_2)/2$. Thus,

$$\begin{aligned}|\mu_v(r)(H_v(t'_{wv}) - H_v(t_{wv})) - \bar{\rho}(r)_v(t'_{wv} - t_{wv})| &= |\rho - \bar{\rho}(r)_v|(t'_{wv} - t_{wv}) \\ &< \nu\,\frac{T + \tau_2}{2}(\tau_2 + \tau_3 + U) \\ &< \nu T(\tau_2 + \tau_3)\,,\end{aligned}$$

where the second last step exploits that $t'_{wv} - t_{wv} \leq q_w(r) + d - (p_w(r) + d - U) \leq \tau_2 + \tau_3 + U$, since clock rates are at least 1, and the final inequality easily follows from Condition 2. □

These results put us in the position to prove that $1 - \mu_v(r)\Delta_{wv}/(\tau_2 + \tau_3)$ is indeed a good estimate of $\bar{\rho}(r)_w - \bar{\rho}(r)_v$. Thus, this (computable) value can serve as a proxy for the difference between "the" clock rates of w and v in round r.

Lemma 9 *For $v, w \in C$ and $r \in \mathbb{N}$, we have that*

$$\left|\bar{\rho}(r)_w - \bar{\rho}(r)_v - \left(1 - \frac{\mu_v(r)\Delta_{wv}}{\tau_2 + \tau_3}\right)\right| \leq \vartheta^3\left(1 - \frac{1}{\vartheta^3}\right)^2 + \frac{\vartheta^3 U}{\tau_2 + \tau_3} + (\vartheta^3 + 1)\nu T\,.$$

Proof We have

$$|t'_{wv} - t_{wv} - (q_w(r) - p_w(r))| \leq U \tag{5}$$

and by Corollaries 8 and 9 that

$$\left|\frac{q_w(r) - p_w(r)}{\tau_2 + \tau_3} - \frac{1}{\bar{\rho}(r)_w}\right| < \frac{\nu T}{\bar{\rho}(r)_w} \leq \nu T \tag{6}$$

$$\left|\frac{\mu_v(r)\Delta_{wv}}{t'_{wv} - t_{wv}} - \bar{\rho}(r)_v\right| < \nu T\,. \tag{7}$$

Note that $|\mu_v(r)\Delta_{wv}/(t'_{wv} - t_{wv})| \leq \vartheta^3$. Therefore,

$$\begin{aligned}\left|\frac{\bar{\rho}(r)_v}{\bar{\rho}(r)_w} - \frac{\mu_v(r)\Delta_{wv}}{\tau_2 + \tau_3}\right| &= \left|\frac{\bar{\rho}(r)_v}{\bar{\rho}(r)_w} - \frac{\mu_v(r)\Delta_{wv}}{t'_{wv} - t_{wv}} \cdot \frac{t'_{wv} - t_{wv}}{q_w(r) - p_w(r)} \cdot \frac{q_w(r) - p_w(r)}{\tau_2 + \tau_3}\right| \\ &\overset{(5)}{\leq} \left|\frac{\bar{\rho}(r)_v}{\bar{\rho}(r)_w} - \frac{\mu_v(r)\Delta_{wv}}{t'_{wv} - t_{wv}} \cdot \frac{q_w(r) - p_w(r)}{\tau_2 + \tau_3}\right| + \frac{\vartheta^3 U}{\tau_2 + \tau_3} \\ &\overset{(6)}{\leq} \left|\frac{\bar{\rho}(r)_v}{\bar{\rho}(r)_w} - \frac{\mu_v(r)\Delta_{wv}}{t'_{wv} - t_{wv}} \cdot \frac{1}{\bar{\rho}(r)_w}\right| + \frac{\vartheta^3 U}{\tau_2 + \tau_3} + \vartheta^3 \nu T \\ &\overset{(7)}{\leq} \frac{\vartheta^3 U}{\tau_2 + \tau_3} + (\vartheta^3 + 1)\nu T\,.\end{aligned}$$

Moreover,

$$\left|\bar{\rho}(r)_w - \bar{\rho}(r)_v - \left(1 - \frac{\bar{\rho}(r)_v}{\bar{\rho}(r)_w}\right)\right| = \left(1 - \frac{1}{\bar{\rho}(r)_w}\right)|\bar{\rho}(r)_w - \bar{\rho}(r)_v| \leq \left(1 - \frac{1}{\vartheta^3}\right)(\vartheta^3 - 1)\,.$$

We conclude that

$$\left|\bar{\rho}(r)_w - \bar{\rho}(r)_v - \left(1 - \frac{\mu_v(r)\Delta_{wv}}{\tau_2 + \tau_3}\right)\right| \leq \vartheta^3\left(1 - \frac{1}{\vartheta^3}\right)^2 + \frac{\vartheta^3 U}{\tau_2 + \tau_3} + (\vartheta^3 + 1)\nu T\,.$$

□

We remark that the $\Theta((1 - 1/\vartheta^3)^2)$ factor is, more precisely, bounded as $\Theta((1 - 1/\vartheta^3)\|\bar{\rho}(r)\|)$. However, for this to be of use, we would have to choose ε depending on r. Since rule-of-thumb calculations show that this term is unlikely to be significant in any real system and the improvement would not extend to the self-stabilizing variant of the algorithm, we refrained from adding this additional complication.

Given that we can bound the "measurement error" of the frequency correction step by Lemma 9, the results from Section 4.1 can be invoked to show convergence. First, we analyze the properties of $\hat{\mu}_v(r+1)$, which Lemma 11 then uses to control $\mu_v(r+1)$.

Lemma 10 *For $v \in C$ and $r \in \mathbb{N}$, abbreviate $\bar{t}_v := (p_v(r) + p_v(r+1))/2$, i.e., $\bar{\rho}(r)_v = \mu_v(r)h_v(\bar{t}_v)$. Then, for all $v, w \in C$,*

$$|\hat{\mu}_v(r+1)h_v(\bar{t}_v) - \hat{\mu}_w(r+1)h_w(\bar{t}_w)| \leq \frac{2\vartheta - 1}{2}\|\bar{\rho}(r)\| + \vartheta\varepsilon\,.$$

Furthermore,

$$(\hat{\mu}_v(r+1) - \varepsilon)h_v(\bar{t}_v) \leq \max_{u \in C}\left\{\mu_u(r)h_u(\bar{t}_u)\right\} - \frac{\varepsilon}{2}$$
$$(\hat{\mu}_v(r+1) + \varepsilon)h_v(\bar{t}_v) \geq \min_{u \in C}\left\{\mu_u(r)h_u(\bar{t}_u)\right\} + \frac{\varepsilon}{2}\,.$$

Proof Set $\delta := \vartheta^3(1-\vartheta^{-3})^2 + \vartheta^3 U/(\tau_2+\tau_3) + (\vartheta^3+1)\nu T$. Observe that, according to Lemma 9, we can interpret $\bar{\rho}(r)_v + \xi_v(r)$, $v \in C$, as the results of an approximate agreement step with error δ on inputs $\bar{\rho}(r)$. By Lemma 2, this implies that

$$|\hat{\mu}_v(r)h_v(\bar{t}_v) + \xi_v(r) - (\hat{\mu}_w(r)h_v(\bar{t}_w) + \xi_w(r))| \leq \frac{\|\bar{\rho}(r)\|}{2} + 2\delta\,.$$

By Corollary 1, $\max_{u \in C}|\{\xi_u(r)|\} \leq \|\bar{\rho}(r)\| + \delta$. Hence, we have for $u \in C$ that

$$|\hat{\mu}_u(r+1)h_u(\bar{t}_u) - (\hat{\mu}_u(r)h_u(\bar{t}_u) + \xi_u(r))| = \left|\frac{2h_u(\bar{t}_u)}{\vartheta + 1} - 1\right| \cdot |\xi_u(r)| \leq \frac{\vartheta - 1}{\vartheta + 1}(\|\bar{\rho}(r)\| + \delta)\,. \tag{8}$$

Using this bound for both v and w, we conclude that

$$\begin{aligned}|\hat{\mu}_v(r+1)h_v(\bar{t}_v) - \hat{\mu}_w(r+1)h_w(\bar{t}_w)| &\leq \frac{\|\bar{\rho}(r)\|}{2} + 2\delta + \frac{2(\vartheta-1)}{\vartheta+1}(\|\bar{\rho}(r)\|+\delta)\\ &< \frac{2\vartheta-1}{2}\|\bar{\rho}(r)\| + (\vartheta+1)\delta\\ &< \frac{2\vartheta-1}{2}\|\bar{\rho}(r)\| + \vartheta\varepsilon\,.\end{aligned}$$

For the second claim of the lemma, we apply Lemma 1. Together with (8), this shows for $v \in C$ that

$$\begin{aligned}\hat{\mu}_v(r+1)h_v(\bar{t}_v) &< \max_{u\in C}\left\{\mu_u(r)h_u(\bar{t}_u)\right\} + \delta + \frac{h_v(\bar{t}_v)-1}{2}(\|\bar{\rho}(r)\|+\delta)\\ \hat{\mu}_v(r+1)h_v(\bar{t}_v) &> \min_{u\in C}\left\{\mu_u(r)h_u(\bar{t}_u)\right\} - \left(\delta + \frac{h_v(\bar{t}_v)-1}{2}(\|\bar{\rho}(r)\|+\delta)\right),\end{aligned}$$

where we used that $2h_v(\bar{t}_v)/(\vartheta+1) - 1 \leq (h_v(\bar{t}_v)-1)/2$. By Condition 2 (and because $\|\bar{\rho}(r)\| \leq \vartheta^3 - 1$),

$$\frac{\varepsilon}{2}h_v(\bar{t}_v) \geq \left(\delta + \frac{(\vartheta-1)(\vartheta^3-1)}{2}\right)h_v(\bar{t}_v) > \delta + \frac{h_v(\bar{t}_v)-1}{2}(\|\bar{\rho}(r)\|+\delta)\,.$$

Combining this with the above inequalities completes the proof. □

Lemma 11 *For round $r \in \mathbb{N}$ and $v \in C$, abbreviate $\bar{t}_v := (p_v(r) + p_v(r+1))/2$, i.e., $\bar{\rho}(r)_v = \mu_v(r)h_v(\bar{t}_v)$. For all $v, w \in C$, we have that*

$$|\mu_v(r+1)h_v(\bar{t}_v) - \mu_w(r+1)h_w(\bar{t}_w)| \leq \max\left\{\frac{2\vartheta-1}{2}\|\bar{\rho}(r)\| + 3\vartheta\varepsilon,\, \|\bar{\rho}(r)\| - \frac{\varepsilon}{2}\right\}.$$

Proof Let $v \in C$ and $w \in C$ maximize and minimize $\mu_u(r+1)h_u(\bar{t}_u)$ over $u \in C$, respectively. By Lemma 10, we have that

$$|\hat{\mu}_v(r+1)h_v(\bar{t}_v) - \hat{\mu}_w(r+1)h_w(\bar{t}_w)| < \frac{2\vartheta-1}{2}\|\bar{\rho}(r)\| + \vartheta\varepsilon\,.$$

We make a case distinction.

Case 1: $\mu_v(r+1) - \hat{\mu}_v(r+1) \leq \varepsilon$ and $\hat{\mu}_w(r+1) - \mu_w(r+1) \leq \varepsilon$. Because we have that $\max\{h_v(\bar{t}_v), h_w(\bar{t}_w)\} \leq \vartheta$, we get

$$\begin{aligned}\mu_v(r+1)h_v(\bar{t}_v) - \mu_w(r+1)h_w(\bar{t}_w) &\leq (\mu_v(r+1) - \hat{\mu}_v(r+1))h_v(\bar{t}_v)\\ &\quad + \hat{\mu}_v(r+1)h_v(\bar{t}_v) - \hat{\mu}_w(r+1)h_w(\bar{t}_w)\\ &\quad + (\hat{\mu}_w(r+1) - \mu_w(r+1))h_w(\bar{t}_w)\\ &\leq \frac{2\vartheta-1}{2}\|\bar{\rho}(r)\| + 3\vartheta\varepsilon\,.\end{aligned}$$

Case 2: $\mu_v(r+1) - \hat{\mu}_v(r+1) > \varepsilon$. This implies that $\mu_v(r+1) = 1 \leq \mu_v(r)$.

a) $\hat{\mu}_w(r+1) \leq \vartheta$, i.e., we have that $\mu_w(r+1) \geq \hat{\mu}_w(r+1) + \varepsilon$. Using Lemma 10, we bound

$$\begin{aligned}\mu_v(r+1)h_v(\bar{t}_v) - \mu_w(r+1)h_w(\bar{t}_w) &\leq h_v(\bar{t}_v)\mu_v(r) \\ &\quad - \left(\min_{u\in C}\{\mu_u(r)h_u(\bar{t}_u)\} + \frac{\varepsilon}{2}\right) \\ &\leq \|\bar{\rho}(r)\| - \frac{\varepsilon}{2}\,.\end{aligned}$$

b) $\hat{\mu}_w(r+1) > \vartheta$, yielding that $\mu_w(r+1) \geq \vartheta - \varepsilon$. It follows that

$$\mu_v(r+1)h_v(\bar{t}_v) - \mu_w(r+1)h_w(\bar{t}_w) \leq h_v(\bar{t}_v) - (\vartheta - \varepsilon) \leq \varepsilon\,.$$

Case 3: $\hat{\mu}_w(r+1) - \mu_w(r+1) > \varepsilon$. This implies that $\mu_w(r+1) = \vartheta^2 \geq \mu_w(r)$.

a) $\hat{\mu}_v(r+1) > \vartheta$, i.e., we have that $\mu_v(r+1) \leq \hat{\mu}_v(r+1) - \varepsilon$. Using Lemma 10, we bound

$$\begin{aligned}\mu_v(r+1)h_v(\bar{t}_v) - \mu_w(r+1)h_w(\bar{t}_w) &\leq \left(\max_{u\in C}\{\mu_u(r)h_u(\bar{t}_u)\} - \frac{\varepsilon}{2}\right) \\ &\quad - h_w(\bar{t}_w)\mu_w(r) \\ &\leq \|\bar{\rho}(r)\| - \frac{\varepsilon}{2}\,.\end{aligned}$$

b) $\hat{\mu}_v(r+1) \leq \vartheta$, yielding that $\mu_v(r+1) \leq \vartheta + \varepsilon$. It follows that

$$\mu_v(r+1)h_v(\bar{t}_v) - \mu_w(r+1)h_w(\bar{t}_w) \leq (\vartheta+\varepsilon)h_v(\bar{t}_v) - \vartheta^2 \leq \vartheta\varepsilon\,.$$

In all cases, we get that

$$\begin{aligned}&\max_{u,u'\in C}\{|\mu_u(r+1)h_u(\bar{t}_u) - \mu_{u'}(r+1)h_{u'}(\bar{t}_{u'})\} \\ =\ &\mu_v(r+1)h_v(\bar{t}_v) - \mu_w(r+1)h_w(\bar{t}_w) \\ \leq\ &\max\left\{\frac{2\vartheta-1}{2}\,\|\bar{\rho}(r)\| + 3\vartheta\varepsilon,\, \|\bar{\rho}(r)\| - \frac{\varepsilon}{2}\right\}.\end{aligned}$$

□

It remains to take into account that hardware clock speeds change between rounds using Lemma 8.

Corollary 10 *For all $r \in \mathbb{N}$,*

$$\|\bar{\rho}(r+1)\| \leq \max\left\{\frac{2\vartheta-1}{2}\|\bar{\rho}(r)\| + 3\vartheta\varepsilon, \|\bar{\rho}(r)\| - \frac{\varepsilon}{2}\right\} + 2\nu(T+\tau_2)\,.$$

Proof By applying Lemma 11 and noting that for all $u \in C$, $|\bar{\rho}(r)_v - \bar{\rho}(r+1)_v| \leq \nu(T+\tau_2)$ by Lemma 8. □

We conclude that the steady state frequency error is in $O(\varepsilon)$.

Corollary 11 *Assume that $\beta := (2\vartheta-1)/2 < 1$. Then*

$$\lim_{r\to\infty}\sup_{r'\geq r}\{\|\vec{\rho}(r')\|\} \leq \frac{3\vartheta\varepsilon + 2\nu(T+\tau_2)}{1-\beta} + \nu(T+\tau_2) \in O(\varepsilon)\,.$$

Proof From iterative application of Corollary 10, we get that

$$\lim_{r\to\infty}\sup_{r'\geq r}\{\|\vec{\rho}(r')\|\} \leq \frac{3\vartheta\varepsilon + 2\nu(T+\tau_2)}{1-\beta}\,.$$

Lemma 8 shows that $\|\vec{\rho}(r')\| \leq \|\bar{\rho}(r')\| + \nu(T+\tau_2)$. Since Condition 2 holds, $1-\beta \in \Omega(1)$ and the overall error is bounded by $O(\varepsilon)$. □

5.2.3 Steady State Error with Frequency Correction

To make use of Corollary 11, we need to derive a variant of Corollary 4 that allows for better control of $\|\vec{p}(r+1)\|$ in case $\|\bar{\rho}(r)\|$ is small.

Lemma 12 *If round $r \in \mathbb{N}$ is executed correctly, then*

$$\|\vec{p}(r+1)\| \leq \frac{4\bar{\vartheta}^2 + 5\bar{\vartheta} - 7}{2(\bar{\vartheta}+1)}\|\vec{p}(r)\| + \left(4\bar{\vartheta}-2\right)U + \|\vec{\rho}(r)\|T\,.$$

Proof For $v, w \in C$, assume w.l.o.g. that $p_v(r+1) - p_w(r+1) \geq 0$ (the other case is symmetric). Denote by $\rho_v \in \vec{\rho}(r)_v$ the average (adjusted) clock rate of v during $[p_v(r), p_v(r+1)]$, i.e.,

$$T - \Delta_v(r) = \frac{H_v(p_v(r+1)) - H_v(p_v(r))}{\mu_v(r)} = \rho_v(p_v(r+1) - p_v(r))\,;$$

ρ_w is defined analogously for w. Recall that $1 \leq \rho_u \leq \bar{\vartheta}$ for $u \in \{v, w\}$. Using this and Corollary 3 (with ϑ replaced by $\bar{\vartheta} = \vartheta^3$), we conclude that

$$\begin{aligned}
&p_v(r+1) - p_w(r+1) \\
&= p_v(r) - p_w(r) + \frac{T - \Delta_v(r)}{\rho_v} - \frac{T - \Delta_w(r)}{\rho_w} \\
&\leq p_v(r) - \Delta_v(r) - (p_w(r) - \Delta_w(r)) + \frac{\rho_w - \rho_v}{\rho_v \rho_w} T \\
&\quad + \left(1 - \frac{1}{\rho_v}\right) |\Delta_v(r)| + \left(1 - \frac{1}{\rho_w}\right) |\Delta_w(r)| \\
&\leq \frac{5\bar{\vartheta} - 3}{2(\bar{\vartheta} + 1)} \|\vec{p}(r)\| + 2\bar{\vartheta} U + \|\vec{\rho}(r)\| T + 2(\bar{\vartheta} - 1)(\|\vec{p}(r)\| + U) \\
&= \frac{4\bar{\vartheta}^2 + 5\bar{\vartheta} - 7}{2(\bar{\vartheta} + 1)} \|\vec{p}(r)\| + \left(4\bar{\vartheta} - 2\right) U + \|\vec{\rho}(r)\| T \,.
\end{aligned}$$

□

Plugging this into our machinery we arrive at the main result of this section.

Theorem 3 *Suppose that Condition 2 is satisfied and that*

$$\bar{\alpha} := \frac{2\bar{\vartheta}^2 + 5\bar{\vartheta} - 5}{2(\bar{\vartheta} + 1)} + (4\bar{\vartheta} + 3)(\bar{\vartheta} - 1) < 1$$

(which is the case for $\vartheta \leq 1.01$). Then, with $\alpha := (4\bar{\vartheta}^2 + 5\bar{\vartheta} - 7)/(2(\bar{\vartheta} + 1)) < 1$ and $\beta := (2\vartheta - 1)/2 < 1$, Algorithm 3 has steady state error

$$E \leq \frac{(4\bar{\vartheta} - 2)U + \nu(T + \tau_2)T}{1 - \alpha} + \frac{(3\vartheta\varepsilon + 2\nu(T + \tau_2))T}{(1 - \alpha)(1 - \beta)} \,.$$

Proof As the preconditions of Theorem 2 are satisfied, all rounds are executed correctly. By Corollary 11, this implies that

$$\lim_{r \to \infty} \sup_{r' \geq r} \{\|\vec{\rho}(r')\|\} \leq \frac{3\vartheta\varepsilon + 2\nu(T + \tau_2)}{1 - \beta} + \nu(T + \tau_2) \,.$$

We plug this into the bound from Lemma 12, which we apply inductively to show that

$$\begin{aligned}
E = \lim_{r \to \infty} \sup_{r' \geq r} \{\|\vec{p}(r')\|\} &\leq \frac{(4\bar{\vartheta} - 2)U + \lim_{r \to \infty} \sup_{r' \geq r} \{\|\vec{\rho}(r)\| T\}}{1 - \alpha} \\
&\leq \frac{(4\bar{\vartheta} - 2)U + \nu(T + \tau_2)T}{1 - \alpha} + \frac{(3\vartheta\varepsilon + 2\nu(T + \tau_2))T}{(1 - \alpha)(1 - \beta)} \,.
\end{aligned}$$

□

Under reasonable assumptions we can obtain a more readable error bound. Intuitively, we require that (i) ϑ is not too large, so that $\alpha \approx 1/2$, (ii) rounds are long enough to allow for a sufficiently accurate frequency measurement, which is the case

if $T \gg \max\{F, U\}$, i.e., rounds are long compared to both the precision F of the initialization and the uncertainty U, and (iii) rounds remain short enough to not let the drifting clocks dominate the error. The third condition amounts to two further constraints: we need that $\nu T^2 \ll U$, since the rate of change of the *speed* of clocks enters the skew bound quadratically in T, and we also need that $(\vartheta - 1)^2 T \ll U$, because inaccurate frequency measurements prevent us from synchronizing frequencies better than up to a factor of $\Theta((\vartheta - 1)^2)$.

Corollary 12 *Assume that the prerequisites of Theorem 3 are satisfied (including (1)). Moreover, suppose that*

- $\alpha \approx 1/2$,
- ε *is chosen minimally such that it satisfies Condition 2,*
- $T \approx \tau_3 \gg \tau_2$, *which is feasible whenever* $T \gg \bar{\vartheta}(e(1) + d)$, *and*
- $\max\{(\bar{\vartheta} - 1)^2 T, \nu T^2\} \ll U$.

Then the steady state error of Algorithm 3 is bounded by roughly $28U$.

Proof Note that $\alpha \approx 1/2$ implies that $\beta \approx 1/2$ and that $\bar{\vartheta} \approx 1$. Plugging ε into the bound from Theorem 3, the steady state error is approximately bounded by

$$\begin{aligned} & 4U + 10\nu(T + \tau_2)T + 12\varepsilon T \\ \approx\ & 4U + 10\nu(T + \tau_2)T + 12\left(6(\bar{\vartheta} - 1)^2 + \frac{2U}{\tau_2 + \tau_3} + 4\nu T\right) T \\ \approx\ & \left(4 + \frac{24T}{\tau_2 + \tau_3}\right) U + 72(\bar{\vartheta} - 1)^2 T + 58\nu T^2 \\ \approx\ & 28U\,. \end{aligned}$$

□

A few remarks:

- Note that that $\vartheta \leq 1.01$ implies that $\beta < \alpha < 0.55$, $\bar{\vartheta} < 1.031$ and $e(1) \leq \max\{1.031F, 0.07T + 4.65U\}$. Thus the requirements of the corollary are met if $\max\{F, U\} \ll T$ and $\max\{(\bar{\vartheta} - 1)^2 T, \nu T^2\} \ll U$ for the minimal choice of ε, yielding the claim stated in the introduction.
- Corollary 12 basically states that increasing T is fine, as long as $\max\{(\bar{\vartheta} - 1)^2 T, \nu T^2\} \ll U$. This improves over Algorithm 2, where it is required that $(\vartheta - 1)T \ll U$, as it permits transmitting pulses at significantly smaller frequencies.
- While the error bound of roughly $28U$ is about factor 7 larger than the about $4U$ Algorithm 2 provides, this is likely to be overly conservative. The source of this difference is that we assume that in a frequency measurement, the full uncertainty U may skew the observation of the relative clock speed. However, this measurement is based on sending two signals in the same direction over the same communication link in fairly short order. In most settings, the difference in delays will be much smaller than between messages on *different* communication links. Accordingly, the relative contribution of the frequency measurement to the error is likely to be much smaller in practice.

– If this is not the case, one may extend the time span for a frequency measurement over multiple rounds to decrease the effect of the uncertainty. This requires that the accumulated phase corrections do not become so large as to prevent a clear distinction of the frequency-related pulse (whose sending time must not be altered due to phase corrections) from phase-related pulses.[12] To not further complicate the analysis, we refrained from presenting this option; it is used in [16, 17].

6 Self-Stabilization

In this section, we propose a generic mechanism that can be used to transform Algorithm 2 and Algorithm 3 into *self-stabilizing* solutions and give the corresponding main results in Theorem 4 and Theorem 5. An algorithm is self-stabilizing, if it (re)establishes correct operation from arbitrary states in bounded time. If there is an upper bound on the time this takes in the worst case, we refer to it as the stabilization time. We stress that, while self-stabilizing solutions to the problem are known, all of them have skew $\Omega(d)$; augmenting the Lynch-Welch approach with self-stabilization capabilities thus enables us to achieve an optimal skew bound of $O((\vartheta - 1)T + U)$ in a Byzantine self-stabilizing manner for the first time.

Our approach can be summarized as follows. Nodes locally count their pulses modulo some $M \in \mathbb{N}$. We use a low-frequency, imprecise, but self-stabilizing synchronization algorithm (called FATAL) from earlier work [4, 5] to generate a "heartbeat." On each such beat, nodes will locally check whether the next pulse with number 1 modulo M will occur within an expected time (local) window whose size is determined by the precision the algorithm would exhibit after M correctly executed pulses (in the non-stabilizing case). If this is not the case, the node is "reset" such that pulse 1 will occur within this time window.

This simple strategy ensures that a beat forces all nodes to generate a pulse with number 1 modulo M within a bounded time window. Assuming a value of F corresponding to its length in Algorithm 2 or Algorithm 3 hence ensures that the respective algorithm will run as intended—at least up to the point when the next beat occurs. Inconveniently, if the beat is not synchronized with the next occurrence of a pulse 1 mod M, some or all nodes may be reset, breaking the guarantees established by the perpetual application of approximate agreement steps. This issue is resolved by leveraging a feedback mechanism provided by FATAL: FATAL offers a (configurable) time window during which a NEXT signal externally provided to each node may trigger the next beat. If this signal arrives at each correct node at roughly the same time, we can be sure that the corresponding beat is generated shortly thereafter. This allows for sufficient control on when the next beat occurs to prevent any node from ever being reset after the first (correct) beat. Since FATAL stabilizes regardless of how the externally provided signals behave, this suffices to achieve stabilization of the resulting compound algorithm.

[12] This issue can be circumvented by having a second, dedicated communication link between each pair of nodes.

6.1 FATAL

Algorithm 4 Interface algorithm, actions for node $v \in C$ in response to a local event at time t. Runs in parallel to local instances of FATAL and either Algorithm 2 or Algorithm 3. In case Algorithm 2 is used, we assume that $\tau_1(r)$, $\tau_2(r)$, and $T(r)$ do not depend on $r \in \mathbb{N}$ and omit r from the notation

```
1  // algorithm maintains local variable i ∈ {0, ..., M − 1}
2  if v generates a pulse at time t then
3      i := i + 1 mod M;
4      if i = 0 then
5          wait for local time H_v(t) + ϑe(M);
6          trigger NEXT signal;
7  if v generates a beat at time t then
8      if i ≠ 0 then
9          // beats should align with every M^th pulse, hence reset
10         reset(R^+);
11     else if next pulse would be sent before local time H_v(t) + R^− then
12         // reset to avoid early pulse
13         reset(R^+ − (H_v(t′) − H_v(t))), where t′ is the current time;
14     else if next round has not started yet at local time H_v(t) + R^+ then
15         // reset to avoid late pulse and start listening for other nodes' pulses on
           time
16         reset(0);
17 Function reset(τ)
18     halt local instance of clock synchronization algorithm;
19     wait for τ local time;
20     i := 0;
21     H_v(t_v(0)) := H_v(t′), where t′ is current time (i.e., t_v(0) := t′);
22     restart loop of clock synchronization algorithm (in round r = 1);
```

We summarize the properties of FATAL in the following corollary, where each node has the ability to trigger a local NEXT signal perceived by the local instance of FATAL at any time.

Corollary 13 (of [5]) *For suitable parameters $P, B_1, B_2, B_3, D \in \mathbb{R}^+$, FATAL stabilizes within $O((B_1+B_2+B_3)n)$ time with probability $1-2^{-\Omega(n)}$. Once stabilized, nodes $v \in C$ generate beats $b_v(k)$, $k \in \mathbb{N}$, such that the following properties hold for all $k \in \mathbb{N}$.*

1. *For all $v, w \in C$, we have that $|b_v(k) - b_w(k)| \leq P$.*
2. *If no $v \in C$ triggers its NEXT signal during $[\min_{w\in C}\{b_w(k)\} + B_1, t]$ for some $t \leq \min_{w\in C}\{b_w(k)\} + B_1 + B_2 + B_3$, then $\min_{w\in C}\{b_w(k+1)\} \geq t$.*
3. *If all $v \in C$ trigger their NEXT signals during $[\min_{w\in C}\{b_w(k)\} + B_1 + B_2, t]$ for some $t \leq \min_{w\in C}\{b_w(k)\} + B_1 + B_2 + B_3$, then $\max_{w\in C}\{b_w(k+1)\} \leq t + P$.*

Denoting by d_F the maximum end-to-end delay (sum of maximum message and computational delay) of FATAL, for any $\phi \geq 1$ and any constant C we can ensure that

$$\begin{aligned} P &\in O(d_F) \\ B_1 &\geq P + d \\ B_1 + B_2 + B_3 &\in \Theta(\phi \cdot (d_F + d)) \\ B_3 &\geq C(B_1 + B_2)\,. \end{aligned}$$

Proof For $\phi = 1$, all statements follow directly from Lemma 3.4 and Corollary 4.16 in [5], noting that nodes will switch from state ready to propose (in the main state machine) in response to a NEXT signal if their timeout T_3 is expired. Once all correct nodes switched to propose, this results in all nodes switching to accept and generating a beat within d_F time. For $\phi > 1$, one simply needs to observe that multiplying each timeout for choices satisfying Condition 3.3 in [5] by ϕ results in another valid choice; the bound on the stabilization time given in Corollary 4.16 scales accordingly. □

6.2 Algorithm

Our self-stabilizing solution utilizes both FATAL and the clock synchronization algorithm with very limited interaction. We already stressed that FATAL will stabilize regardless of the NEXT signals and note that it is not influenced by Algorithm 4 in any other way. Concerning the clock synchronization algorithm (either Algorithm 2 or Algorithm 3), we assume that a "careful" implementation is used that does not maintain state variables for a long time. Concretely, Algorithm 2 will clear memory between loop iterations, and Algorithm 3 will memorize the new multiplier value $\mu_v(r+1)$ only, which is explicitly assigned during round r. If this is satisfied, no further consistency checks of variables are required, and it will be straightforward to re-use the analyses from Sections 4.3 and 5.2.

Having said this, let us turn to Algorithm 4, which is basically an ongoing consistency check based on the beats that resets the clock synchronization algorithm if necessary. The feedback triggering the next beat in a timely fashion is implemented by simply triggering the NEXT signal on each M^{th} beat, with a small delay ensuring that all nodes arrive in the same round and have their counter variable i reading 0. The consistency checks then ask for $i = 0$ and the next pulse being triggered within a certain local time window; if either does not apply, the reset function is called, ensuring that both conditions are met (Fig. 3).

Condition 3 lists the constraints on R^- (the minimum local time between a beat and local pulse 1 mod M), R^+ (the respective maximum local time), and M (the number of pulses between beats) – the parameters of Algorithm 4 – need to satisfy so that we can show that the algorithm is guaranteed to stabilize.

Condition 3 *We require that*

$$P + R^+ + \tau_1 - \frac{R^-}{\vartheta} \leq e(1) \tag{9}$$

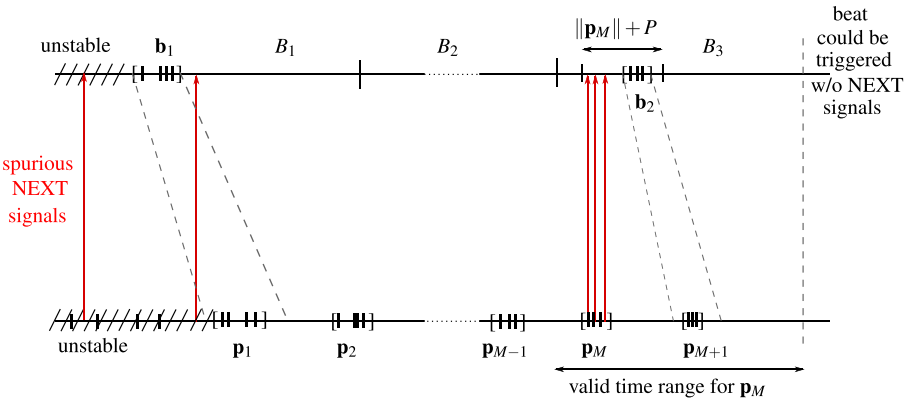

Fig. 3 Interaction of the beat generation and clock synchronization algorithms in the stabilization process, controlled by Algorithm 4. Beat $\vec{b}_1$ forces pulse $\vec{p}_1$ to be roughly synchronized. The approximate agreement steps then result in tightly synchronized pulses. By the time the nodes trigger beat $\vec{b}_2$ by providing NEXT signals based on $\vec{p}_M$, synchronization is tight enough to guarantee that the beat results in no resets

$$P + R^+ \leq \frac{R^-}{\vartheta} \tag{10}$$

$$P + R^+ + \tau_1 + d \leq \frac{R^- + \tau_2}{\vartheta} \tag{11}$$

$$P + d \leq \frac{R^- - \tau_1}{\vartheta} \tag{12}$$

$$P + R^+ + T + \vartheta(e(1) + U) \leq B_1 + B_2 \tag{13}$$

$$P + \vartheta e(M) \leq B_1 \tag{14}$$

$$B_1 + B_2 \leq e(M) + (M-1)\left(\frac{T}{\vartheta} - \tau_1\right) + \frac{R^-}{\vartheta} \tag{15}$$

$$\vartheta e(M) + (M-1)(T + \vartheta\tau_1) + P + R^+ + \tau_1 \leq B_1 + B_2 + B_3 \tag{16}$$

$$R^- \leq \frac{T}{\vartheta} - ((\vartheta + 2)e(M) + U + P) \tag{17}$$

$$T + \vartheta(e(M) + U) - \tau_1 \leq R^+ . \tag{18}$$

Intuitively, these constraints ensure the following:

- Equation (9) says that resets on a beat enforce the skew to become bounded by $e(1)$.
- Equations (10) and (11) ensure that correct nodes receive the first pulses from all other correct nodes after a beat.
- Equation (12) guarantees that these are actually the "round-1" pulses also for nodes that have been reset, i.e., there are no spurious pulses from before such a reset that are received during the respective time window.

- Equations (13) and (14) make sure that FATAL will ignore any NEXT signals that may still be active when a beat occurs and that there is sufficient time for the first round after the beat to complete.
- Equations (15) and (16) enforce that the (now correctly executing) algorithm will trigger the NEXT signals and thus the next beat is well-aligned with the time reference it provides.
- Finally, (17) and (18) imply that such a beat will result in no resets.

We need to show that these constraints can be satisfied in conjunction with the ones required by the employed synchronization algorithm.

Lemma 13 *Conditions 1 and 3 can be simultaneously satisfied such that* $\tau_1(r) = \tau_1$, $\tau_2(r) = \tau_2$ *and* $T(r) = T$ *for all* $r \in \mathbb{N}$, *and* $\lim_{r\to\infty} e(r) < \infty$ *if*

$$\alpha = \frac{2\vartheta^2 + \vartheta}{2 - \vartheta} \cdot \left(1 - \frac{1}{\vartheta^2} + \frac{4(\vartheta - 1)}{1 - \beta}\right) < 1\,,$$

where $\beta = (2\vartheta^2 + 5\vartheta - 5)/(2(\vartheta + 1))$. *In this case,*

$$\lim_{r\to\infty} e(r) = \frac{(1 - 1/\vartheta)T + (3\vartheta - 1)U}{1 - \beta}\,.$$

Here, we may choose any $T \geq T_0 \in O((d_F + d)/(1 - \alpha))$ *and* B_1, B_2, *and* B_3 *such that FATAL stabilizes in time* $O(n(d_F + d))$ *with probability* $1 - 2^{-\Omega(n)}$.

Proof We choose R^- and R^+ such that (17) and (18) are satisfied with equality. Thus, any choice of

$$F \geq \left(1 - \frac{1}{\vartheta^2}\right) T + 2P + 4\vartheta e(M) + 2\vartheta U$$

satisfies (9), and for (10)–(12) to hold it is sufficient that

$$\begin{aligned} F \;\leq\; \tau_1 &\leq \frac{T}{\vartheta} - 3\vartheta e(M) - \vartheta d - (\vartheta - 1)P \\ \vartheta F \;&\leq\; \tau_2\,. \end{aligned}$$

These lower bounds on τ_1 and τ_2 are weaker than those imposed by Condition 1, which demands that $\min\{\tau_1, \tau_2\} \geq \vartheta e(1) > F$. Setting $\tau_1 := \vartheta e(1)$, $\tau_2 := \vartheta(e(1) + d)$, and requiring $T \geq \vartheta(\tau_1 + \tau_2 + e(1) + U)$ thus guarantees that the above lower bounds on τ_1 and τ_2 hold. We get that

$$\frac{T}{\vartheta} > \tau_1 + F + \vartheta d > \tau_1 + 3\vartheta e(M) + \vartheta d + (\vartheta - 1)P\,,$$

and the inequalities of Condition 1 are satisfied for $r = 1$. Moreover, with $x := (3\vartheta - 1)U + (1 - 1/\vartheta)T$, we have for $r \in \mathbb{N}$ that

$$e(r) = \beta^{r-1} e(1) + \frac{1 - \beta^{r-1}}{1 - \beta}\, x\,,$$

i.e., $e(r)$ is a convex combination of $e(1)$ and $x/(1-\beta)$. We require that $e(1) \geq x/(1-\beta)$, i.e.,

$$\frac{F}{2-\vartheta} = e(1) \geq \frac{(3\vartheta-1)U + (1-1/\vartheta)T}{1-\beta};$$

here, we used that $2-\vartheta > 0$, because $\alpha < 1$. Thus, $e(r) \leq e(1)$, and we conclude that Condition 1 holds for

$$F := \max\left\{\left(1-\frac{1}{\vartheta^2}\right)T + 2P + 4\vartheta e(M) + 2\vartheta U, \frac{(2-\vartheta)((3\vartheta-1)U+(1-1/\vartheta)T)}{1-\beta}\right\}$$

under the constraint that

$$T \geq \vartheta(\tau_1 + \tau_2 + e(1) + U) = \vartheta\left(\frac{(2\vartheta+1)F}{2-\vartheta} + \vartheta d + U\right).$$

For any $c > 1$, sufficiently large M ensures that

$$e(M) \leq c \lim_{r\to\infty} e(r) = \frac{cx}{1-\beta} = \frac{c((3\vartheta-1)U + (1-1/\vartheta)T)}{1-\beta},$$

where the last step uses that $1-\beta \in \Omega(1)$ because $\alpha < 1$.

Assuming sufficiently large M, the above lower bound on T can hence be met iff

$$\frac{2\vartheta^2+\vartheta}{2-\vartheta} \cdot \max\left\{1 - \frac{1}{\vartheta^2} + \frac{4(\vartheta-1)}{1-\beta}, \frac{(2-\vartheta)(1-1/\vartheta)}{1-\beta}\right\} = \alpha < 1.$$

In this case, for sufficiently large M the constraint on T is satisfied if

$$(1-\alpha)T \geq (1-\alpha)T_0 \in O\left(\max\left\{P + \frac{U}{1-\beta} + U, \frac{U}{1-\beta}\right\} + d + U\right) = O(P+d),$$

where we used that ϑ and thus $1-\alpha$ and $1-\beta$ are constants.

To complete the proof, it remains to show that, for any such choice of T and a given lower bound on M, we can satisfy Inequalities (13)–(16) such that FATAL has the claimed guarantees on the stabilization time. Given that all parameters except for M, B_1, B_2, and B_3 are already fixed independently of these values, it suffices if we can solve the system

$$\begin{aligned} K &\leq B_1 \\ B_1 + B_2 &\leq (M-1)K \\ \vartheta M K &\leq B_1 + B_2 + B_3 \end{aligned}$$

for an arbitrary $K \in \mathbb{R}^+$ such that M is sufficiently large. By Corollary 13, we may choose B_1, B_2, and B_3 such that, e.g., $B_3 \geq B_1 + B_2$. Picking $\phi \geq 1$ in the corollary sufficiently large, we get that $\phi B_1 \geq K$ and $M := \lfloor 2(B_1+B_2)/(\vartheta K)\rfloor$ is sufficiently large and satisfies the second and third inequality (where again we use that $2-\vartheta \in \Omega(1)$).

Finally, note that $P \in O(d_F)$ and all factors occurring in this proof are constants depending on ϑ only, implying that ϕ and M are constants as well. The bound on the stabilization time thus readily follows from Corollary 13 as well. □

In the remainder of the section, we assume (i) that the beat generation algorithm has already stabilized, i.e., the guarantees stated in Corollary 13 hold, (ii) that the executed clock synchronization algorithm is Algorithm 2, and (iii) that Condition 1 holds. The analysis for Algorithm 3 is analogous, where $\bar{\vartheta} = \vartheta^3$ takes the role of ϑ and Condition 2 takes the role of Condition 1; this is formalized by the following corollary and Theorem 5 at the end of this section.

Corollary 14 *Conditions 2 and 3 can be simultaneously satisfied with* $\lim_{r\to\infty} e(r) < \infty$ *if*

$$\bar{\alpha} = \frac{4\bar{\vartheta}^2 + 5\bar{\vartheta}}{2 - \bar{\vartheta}} \cdot \left(1 - \frac{1}{\bar{\vartheta}^2} + \frac{4(\bar{\vartheta} - 1)}{1 - \bar{\beta}}\right) < 1\,,$$

where $\bar{\vartheta} = \vartheta^3$ *and* $\bar{\beta} = (2\bar{\vartheta}^2 + 5\bar{\vartheta} - 5)/(2(\bar{\vartheta} + 1))$. *In this case,*

$$\lim_{r\to\infty} e(r) = \frac{(1 - 1/\bar{\vartheta})T + (3\bar{\vartheta} - 1)U}{1 - \beta}\,.$$

Here, we may choose any $T \geq T_0 \in O((d_F + d + U)/(1 - \alpha))$ *and* B_1, B_2, *and* B_3 *such that FATAL stabilizes in time* $O(n(d_F + d))$ *with probability* $1 - 2^{-\Omega(n)}$.

Proof Analogous to the proof of Lemma 13, but replacing the constraint $T \geq \vartheta(\tau_1 + \tau_2 + e(1) + U)$ by $T \geq \tau_1 + \tau_2 + \tau_3 + \tau_4 + \bar{\vartheta}(e(1) + U) > \bar{\vartheta}(\tau_1 + \tau_2 + e(1) + U)$ and setting $\tau_3 := \bar{\vartheta}(e(1) + (1 - 1/\bar{\vartheta})(\tau_1 + \tau_2))$ and $\tau_4 := \bar{\vartheta}(e(1) + d + (1 - 1/\bar{\vartheta})(\tau_1 + \tau_2))$ in accordance with Condition 2. This results in the requirement that

$$T \geq \frac{(4\bar{\vartheta}^2 + 5\bar{\vartheta})F}{2 - \vartheta} + \bar{\vartheta}d + U\,,$$

which in turn leads to the value for $\bar{\alpha}$. □

6.3 Analysis

Our analysis starts with the first correct beat produced by FATAL, which is perceived at node $v \in C$ at time $b_v(1)$. Subsequent beats at v occur at times $b_v(2)$, $b_v(3)$, etc. We first establish that the first beat guarantees to "initialize" the synchronization algorithm such that it will run correctly from this point on (neglecting for the moment the possible intervention by further beats). We use this do define the "first" pulse times $p_v(1)$, $v \in C$, as well; we enumerate consecutive pulses accordingly.

Lemma 14 *Let* $b := \min_{v\in C}\{b_v(1)\}$. *We have that*

1. *Each* $v \in C$ *generates a pulse at time* $p_v(1) \in [b + R^-/\vartheta, b + P + R^+ + \tau_1]$.
2. $\|\vec{p}(1)\| \leq e(1)$.
3. *At time* $p_v(1)$, $v \in C$ *sets* $i := 1$.
4. $w \in C$ *receives the pulse sent by* $v \in C$ *at a local time from the range* $[H_w(p_w(1)) - \tau_1, H_w(p_w(1)) + \tau_2]$.
5. *This is the only pulse* w *receives from* v *at a local time from the range* $[H_w(p_w(1)) - \tau_1, H_w(p_w(1)) + \tau_2]$.

6. *Denoting by round 1 the execution of the for-loop in Algorithm 2 during which each $v \in C$ sends the pulse at time $p_v(1)$, this round is executed correctly.*

Proof Assume for the moment that $\min_{v\in C}\{b_v(2)\}$ is sufficiently large, i.e., no second beat will occur at any correct node for the times relevant to the proof of the lemma; we will verify this at the end of the proof.

From the pseudocode given in Algorithms 2 and 4, it is straightforward to verify that $v \in C$ generates a pulse at a local time from $[H_v(b_v(1)) + R^-, H_v(b_v(1)) + R^+ + \tau_1]$. Since $b_v(1) \in [b, b + P]$ by Corollary 13, this shows the first claim. The second follows immediately, since

$$\|\vec{p}(1)\| \leq P + R^+ + \tau_1 - \frac{R^-}{\vartheta} \overset{(9)}{\leq} e(1)\,.$$

Note that, until we show the last claim, it is not clear that $p_v(1)$ is unique for each $v \in C$. For the moment, let $p_v(1)$ be the first pulse $v \in C$ sends during the local time interval $[H_v(b_v(1)) + R^-, H_v(b_v(1)) + R^+ + \tau_1]$. With this convention, the third claim is shown as follows. Observe that any $v \in C$ that executes the reset function in response to the beat sets $i := 0$ when doing so. Hence, it will set $i := 1$ at time $p_v(1)$. Thus, consider $v \in C$ that does not execute the reset function. This entails that $i = 0$ at time $b_v(1)$ and v generates no pulse during local times from $[H_v(b_v(1), H_v(b_v(1)) + R^-)$. Consequently, v will increase i to 1 at time $p_v(1)$.

For the fourth claim, we bound

$$p_v(1) \geq b + \frac{R^-}{\vartheta} \geq b_w(1) + \frac{R^-}{\vartheta} - P \overset{(10)}{\geq} b_w(1) + R^+\,.$$

Thus, either the next round has already started at node w by time $p_v(1)$ or w calls reset with argument 0, i.e., starts a new round. Either way, we have that w receives the pulse from v no earlier than local time $H_w(p_w(1)) - \tau_1$. To see that the pulse arrives on time, we bound

$$p_v(1) + d \leq p_w(1) + P + R^+ + \tau_1 + d - \frac{R^-}{\vartheta} \overset{(11)}{\leq} p_w(1) + \frac{\tau_2}{\vartheta}\,.$$

As $H_w(p_w(1) + \tau_2/\vartheta) \leq H_w(p_w(1)) + \tau_2$, the fourth claim follows.

Concerning the fifth claim, observe that $v \in C$ sends exactly one pulse during the local time interval $[H_v(b_v(1)), H_v(p_v(1))]$. As for $w \in C$ we have that

$$b_v(1) + d \leq b_w(1) + P + d \leq p_w(1) - \frac{R^-}{\vartheta} + P + d \overset{(12)}{\leq} p_w(1) - \frac{\tau_1}{\vartheta}\,,$$

no pulse v sent at an earlier local time is received by w at or after local time $H_w(p_w(1)) - \tau_1$. In particular, the first pulse w receives from v at a local time from $[H_w(p_w(1)) - \tau_1, H_w(p_w(1)) + \tau_2]$ arrives at w at a time $t_{vw} \in [p_v(1) + d - U, p_v(1) + d]$. Since we also showed that $\|\vec{p}(1)\| \leq e(1)$, we conclude that the analysis of Section 4.3 can be applied to show that any subsequent pulse arrives after the round is complete at all nodes. Furthermore, we conclude that round 1 is executed correctly.

Recall that in the above reasoning, we assumed that $\min_{v\in C}\{b_v(2)\}$ is sufficiently large. Clearly, this is the case if round 1 ends at all nodes before this time. Accordingly, we bound for $v \in C$

$$\begin{aligned} p_v(1) + T - \Delta_v(1) - \tau_1 &\leq b_v(1) + R^+ + T - \Delta_v(1) \\ &\leq b + P + R^+ + T + \vartheta(e(1) + U) \\ &\overset{(13)}{\leq} b + B_1 + B_2\,, \end{aligned}$$

where the second last step makes use of Corollary 3. Because no node $v \in C$ generates a pulse with $i = M$ during times $[b_v(1) + \vartheta e(M),\, p_v(2)]$, no such node triggers a NEXT signal during this time interval (cf. Algorithm 4). We have that

$$b_v(1) + \vartheta e(M) \leq b + P + \vartheta e(M) \overset{(14)}{\leq} B_1\,,$$

implying by Corollary 13 that $\min_{v\in C}\{b_v(2)\} \geq b + B_1 + B_2$. □

Lemma 14 serves as induction anchor for the argument showing that all rounds of the algorithm are executed correctly. However, due to possible interference of future beats, for the moment we can merely conclude that this is the case until the next beat; we obtain the following corollary.

Corollary 15 *Denote by N the infimum over all times $t \geq b + B_1$ at which some $v \in C$ triggers a NEXT signal. If $\min_{v\in C}\{p_v(M) + e(M)\} \leq \min\{N, b + B_1 + B_2 + B_3\}$, then all rounds $r \in \{1, \dots, M\}$ are executed correctly and $\|\vec{p}(r)\| \leq e(r)$.*

Proof Lemma 14 shows that the first beat "initializes" the system such that $\|\vec{p}(1)\| \leq e(1)$ and the first round is executed correctly. By Corollary 13, $\min_{v\in C}\{b_v(2)\} \geq \min\{N, b + B_1 + B_2 + B_3\}$. Hence, after round 1 Algorithm 2 will be executed without interference from Algorithm 4 until (at least) time $\min_{v\in C}\{p_v(M) + e(M)\}$. For $r \in \{2, \dots, M\}$, the claim thus follows as in Section 4.3. □

Next, we leverage this insight to prove that the progress of the synchronization algorithm – which will operate correctly at least until the next beat – together with the constraints of Condition 3 ensures the following: the first time when node $v \in C$ triggers its NEXT signal after time $b + B_1$ falls within the window of opportunity for triggering the next beat provided by FATAL.

Lemma 15 *For $v \in C$, denote by $N_v(1)$ the infimum of times $t \geq b + B_1$ when it triggers its NEXT signal. We have that $H_v(N_v(1)) = p_v(M) + \vartheta e(M)$ and that*

$$b + B_1 + B_2 \leq N_v(1) \leq b + B_1 + B_2 + B_3\,.$$

Proof At time $b_v(1)$, $v \in C$ sets $i := 0$ (unless it already holds that $i = 0$). Thus, v will not trigger the NEXT signal until it sent at least M pulses and waited for $\vartheta e(M)$

local time, i.e., $N_v(1) \geq p_v(M) + e(M)$. As observed in the proof of Lemma 14, we have that $b_v(1) \geq b + B_1$. Thus, we can apply Corollary 15, where

$$N := \min_{v \in C}\{N_v(1)\} \geq \min_{v \in C}\{p_v(M) + e(M)\}\,,$$

to conclude that one of the following must hold true: (i) all rounds $r \in \{1, \ldots, M\}$ are executed correctly or (ii) $\min_{v \in C}\{p_v(M) + e(M)\} > b + B_1 + B_2 + B_3$.

In the first case, we have that

$$H_v(N_v(1)) = H_v(p_v(1)) + \vartheta e(M) + \sum_{r=1}^{M-1} T - \Delta_v(r)\,,$$

where

$$\sum_{r=1}^{M-1} |\Delta_v(r)| \leq \sum_{r=1}^{M_1} e(r) \leq \vartheta(M-1)\tau_1\,.$$

We conclude that

$$p_v(1) + e(M) + (M-1)\left(\frac{T}{\vartheta} - \tau_1\right) \leq N_v(1) \leq p_v(1) + \vartheta e(M) + (M-1)(T + \vartheta\tau_1).$$

Applying the first statement of Lemma 14, this yields that

$$\begin{aligned} & b + e(M) + (M-1)\left(\frac{T}{\vartheta} - \tau_1\right) + \frac{R^-}{\vartheta} \\ \leq\ & N_v(1) \\ \leq\ & b + \vartheta e(M) + (M-1)(T + \vartheta\tau_1) + P + R^+ + \tau_1\,. \end{aligned}$$

The claim now follows from (15) and (16).

With respect to the second case, observe that since no NEXT signal is triggered at any $v \in C$ after time $b + B_1$ until time $b + B_1 + B_2 + B_3$, $\min_{v \in C}\{b_v(2)\} \geq b + B_1 + B_2 + B_3$ by Corollary 13. Thus, Algorithm 2 runs without interference up to this time. Using this, we can establish the same bounds as for the first case. □

This immediately implies that the second beat occurs in response to the NEXT signals, which itself are aligned with pulse M.

Corollary 16 *For all $v \in C$, $b_v(2) \in [p_v(M),\ p_v(M) + (\vartheta + 1)e(M) + P]$.*

Proof By Lemma 15, $N_v(1) \in [b + B_1 + B_2,\ b + B_1 + B_2 + B_3]$ for all $v \in C$. Thus, by Corollary 15, $\|\vec{p}(M)\| \leq e(M)$. As $v \in C$ triggers its NEXT signal at local time $H_v(p_v(M)) + \vartheta e(M)$, it follows that

$$p_v(M) \leq \min_{w \in C}\{p_w(M) + e(M)\} \leq \min_{w \in C}\{N_w(1)\}$$

and that

$$\max_{w \in C}\{N_w(1)\} \leq \max_{w \in C}\{p_w(M) + \vartheta e(M)\} \leq p_v(M) + (\vartheta + 1)e(M)\,.$$

The claim now follows from the second and third statements of Corollary 13. □

Having established this timing relation between $\vec{b}(2)$ and $\vec{p}(M)$, we can conclude that no correct node is reset due to the second beat.

Lemma 16 *Node $v \in C$ does not call the reset function of Algorithm 4 in response to beat $b_v(2)$.*

Proof By Corollary 16, $b_v(2) \in [p_v(M), p_v(M) + (\vartheta + 1)e(M) + P]$. By Corollary 15, Algorithm 2 has been executed without interruption by beat after time $b_v(1)$ up to this time. Hence, v sets $i := M \bmod M = 0$ at time $p_v(M) \leq b_v(2)$. As also round M is executed correctly, the earliest time when v could generate pulse $M + 1$ without a reset is bounded by

$$\begin{aligned} p_v(M) + \frac{T - \Delta_v(M)}{\vartheta} &\geq p_v(M) - (e(M) + U) + \frac{T}{\vartheta} \\ &\geq b_v(2) - ((\vartheta + 2)e(M) + P + U) + \frac{T}{\vartheta} \\ &\overset{(17)}{\geq} b_v(2) + R^- , \end{aligned}$$

where in the first step we applied Corollary 3. This implies that node v's variable i equals 0 at time $b_v(2)$ and v does not generate a pulse at a local time from $[H_v(b_v(2)), H_v(b_v(2)) + R^-]$. It remains to show that v enters round $M + 1$ at the latest at local time $H_v(b_v(2)) + R^+$. To show this, we bound

$$\begin{aligned} H_v(p_v(M)) + T - \tau_1 - \Delta_v(M) &\leq H_v(p_v(M)) + T - \tau_1 + \vartheta(e(M) + U) \\ &\leq H_v(b_v(2)) + T - \tau_1 + \vartheta(e(M) + U) \\ &\overset{(18)}{\leq} b_v(2) + R^+ . \end{aligned}$$

□

Repeating the above reasoning for all pairs of beats $\vec{b}(k)$, $\vec{b}(k + 1)$, $k \in \mathbb{N}$, it follows that no correct node is reset by any beat other than the first. Thus, the clock synchronization algorithm is indeed (re-)initialized by the first beat to run without any further meddling from Algorithm 4. This implies the same bounds on the steady state error as for the original synchronization algorithm.

Theorem 4 *Suppose that Algorithm 4 is executed with Algorithm 2 as synchronization algorithm. If*

$$\alpha = \frac{2\vartheta^2 + \vartheta}{2 - \vartheta} \cdot \left(1 - \frac{1}{\vartheta^2} + \frac{4(\vartheta - 1)}{1 - \beta}\right) < 1$$

(which holds for $\vartheta \leq 1.03$), where $\beta = (2\vartheta^2 + 5\vartheta - 5)/(2(\vartheta + 1))$, then all parameters can be chosen such that the compound algorithm is self-stabilizing and has steady state error

$$E \leq \frac{(\vartheta - 1)T + (3\vartheta - 1)U}{1 - \beta} .$$

Here, any nominal round length $T \geq T_0 \in O(d_F + d)$ is possible.

Proof Lemma 13 that Conditions 1 and 3 can be satisfied such that $\lim_{r\to\infty} e(r) = ((\vartheta-1)T+(3\vartheta-1)U)/\beta$ and $T_0 \in O(d_F+d)$. Hence, we may apply the statements derived in this section.

By Corollary 13, the beat generation mechanism will eventually stabilize. Afterwards, we can apply Lemma 16 to show that the second (correct) beat results in no calls to the reset function in Algorithm 4. In fact, this extends to any beat except for the first: letting beat $k \in \mathbb{N}$ take the role of beat 1, our reasoning shows that beat $k+1$ does not result in a reset at any node. Moreover, applying the same reasoning to Corollary 15, we conclude that all rounds $r \in \mathbb{N}$ are executed correctly, and that $\|\vec{p}(r)\| \leq e(r)$. The bound on E follows. □

Observe that, in comparison to Theorem 1, the expression obtained for the steady state error replaces d by $O(d_F+d)$, which is essentially the skew upon initialization by the first beat. In Algorithm 2, we circumvented any dependence on F by varying round lengths over time. For the self-stabilizing solution, this is not possible, since counting rounds locally is not guaranteed to ensure a consistent opinion across all nodes concerning the nominal length of the current round; we are restricted to counting rounds $\bmod M \in \mathbb{N}$, so any long round length will reoccur regularly.

It remains to draw the analogous conclusions for using Algorithm 4 with Algorithm 3 as synchronization algorithm.

Theorem 5 *Suppose that Algorithm 4 is executed with Algorithm 3 as synchronization algorithm (where (1) holds). If*

$$\bar{\alpha} = \frac{4\bar{\vartheta}^2+5\bar{\vartheta}}{2-\bar{\vartheta}} \cdot \left(1-\frac{1}{\bar{\vartheta}^2}+\frac{4(\bar{\vartheta}-1)}{1-\bar{\beta}}\right) < 1$$

(which holds for $\vartheta \leq 1.004$), where $\bar{\vartheta} = \vartheta^3$ and $\bar{\beta} = (2\bar{\vartheta}^2+5\bar{\vartheta}-5)/(2(\bar{\vartheta}+1))$, then all parameters can be chosen such that the compound algorithm self-stabilizes in $O(n)$ time and has steady state error

$$E \leq \frac{(4\bar{\vartheta}-2)U+\nu(T+\tau_2)T}{1-\alpha} + \frac{(3\vartheta\varepsilon+2\nu(T+\tau_2))T}{(1-\alpha)(1-\beta)},$$

where $\alpha := (4\bar{\vartheta}^2+5\bar{\vartheta}-7)/(2(\bar{\vartheta}+1)) < 1$ and $\beta := (2\vartheta-1)/2 < 1$. Here, any value of $T \geq T_0 \in O(d_F+d)$ is possible.

Proof As for Theorem 4, with Corollary 14 taking the place of Lemma 13 and noting that the convergence argument for the frequencies relies on rounds being executed correctly only (i.e., no assumptions on $\mu_v(1)$, $v \in C$, are required). □

We remark that despite the stringent requirements on ϑ for the recovery argument to work (i.e., $\bar{\alpha} < 1$), the actual bound on the precision involves α and β. If $\vartheta \leq 1.004$, we have $\alpha \leq 0.512$ and $\beta \leq 0.502$. Concerning stabilization, we remark that it takes $O(n)$ time with probability $1-2^{-\Omega(n)}$, which is directly inherited from FATAL. The subsequent convergence to small skews is not affected by n, and will be much faster for realistic parameters, so we refrain from a more detailed statement.

7 Conclusions

The results derived in this paper demonstrate that the Lynch-Welch synchronization principle is a promising candidate for reliable clock generation, not only in software, but also in hardware. Apart from accurate bounds on the synchronization error depending on the quality of clocks, we present a generic coupling scheme enabling to add self-stabilization properties.

We believe these results to be of practical merit. Concretely, first results from a prototype Field-Programmable Gate Array (FPGA) implementation of Algorithm 2 show a skew of 182 ps [12]. Given the appealing simplicity of the presented algorithms and this excellent performance, we consider the approach a viable candidate for reliable clock generation in fault-tolerant low-level hardware and other areas.

Acknowledgements Open access funding provided by Max Planck Society. We thank Matthias Függer and Attila Kinali for fruitful discussions, and the anonymous reviewers of an earlier version for valuable comments.

References

1. Overview of Silicon Oscillators by Linear Technology (retrieved May 2016). http://cds.linear.com/docs/en/product-selector-card/2PB_osccalcfb.pdf
2. Daliot, A., Dolev, D.: Self-stabilizing byzantine pulse synchronization computing research repository. arXiv:0608092 (2006)
3. Distributed Algorithms for Robust Tick-Synchronization (2005–2008). Research project [retrieved: 05, 2014]. http://ti.tuwien.ac.at/ecs/research/projects/darts
4. Dolev, D., Függer, M., Lenzen, C., Posch, M., Schmid, U., Steininger, A.: Rigorously modeling self-stabilizing fault-tolerant circuits: An ultra-robust clocking scheme for systems-on-chip. J. Comput. Syst. Sci. **80**(4), 860–900 (2014)
5. Dolev, D., Függer, M., Lenzen, C., Schmid, U.: Fault-tolerant algorithms for tick-generation in asynchronous logic: Robust pulse generation. J. ACM **61**(5), 30:1–30:74 (2014)
6. Dolev, D., Halpern, J.Y., Strong, H.R.: On the possibility and impossibility of achieving clock synchronization. J. Comput. Syst. Sci. **32**(2), 230–250 (1986)
7. Dolev, D., Lynch, N.A., Pinter, S.S., Stark, E.W., Weihl, W.E.: Reaching approximate agreement in the presence of faults. J. ACM **33**, 499–516 (1986)
8. Dolev, S., Welch, J.L.: Self-stabilizing clock synchronization in the presence of byzantine faults. J. ACM **51**(5), 780–799 (2004)
9. FlexRay Consortium, et al.: FlexRay communications system-protocol specification. Version 2.1 (2005)
10. Függer, M., Armengaud, E., Steininger, A.: Safely stimulating the clock synchronization algorithm in time-triggered systems - a combined formal & experimental approach. IEEE Trans. Indus. Inf. **5**(2), 132–146 (2009)
11. Függer, M., Schmid, U.: Reconciling fault-tolerant distributed computing and systems-on-chip. Distrib. Comput. **24**(6), 323–355 (2012)
12. Huemer, F., Kinali, A., Lenzen, C.: Fault-tolerant clock synchronization with high precision. In: IEEE Symposium on VLSI (ISVLSI), pp. 490–495 (2016)

13. Kopetz, H., Bauer, G.: The time-triggered architecture. Proc. IEEE **91**(1), 112–126 (2003)
14. Lenzen, C., Rybicki, J.: Self-stabilising Byzantine clock synchronisation is almost as easy as consensus. In: 31st Symposium on Distributed Computing (DISC). To appear (2017)
15. Lundelius, J., Lynch, N.: An upper and lower bound for clock synchronization. Inf. Control. **62**(2–3), 190–204 (1984)
16. Schossmaier, K.: Interval-based Clock State and Rate Synchronization. Technical University of Vienna, Ph.D. thesis (1998)
17. Schossmaier, K., Weiss, B.: An algorithm for fault-tolerant clock state and rate synchronization. In: 18th Symposium on Reliable Distributed Systems (SRDS), pp. 36–47 (1999)
18. Srikanth, T.K., Toueg, S.: Optimal clock synchronization. J. ACM **34**(3), 626–645 (1987)
19. Welch, J.L., Lynch, N.A.: A new fault-tolerant algorithm for clock synchronization. Inf. Comput. **77**(1), 1–36 (1988)