The Welch-Lynch
Clock Synchronization Algorithm

Bruno Dutertre

Technical Report 747

March 27, 1998

Department of Computer Science
Queen Mary and Westfield College, University of London
Mile End Road, London E1 4NS, UK
Abstract

This note describes the Welch-Lynch fault-tolerant algorithm for clock synchronization. The original proof given by Welch and Lynch shows that the clocks of non-faulty nodes are maintained in approximate agreement. The worst-case skew is bounded by a constant which depends on network and algorithm parameters. We give a simplified proof of correctness and obtain tight synchronization bounds.
1 Introduction

In [2], Welch and Lynch present a fault-tolerant algorithm for clock synchronization in distributed systems. The algorithm is intended for a fully connected network of \( n \) processes, less than a third of which are faulty. Byzantine failures are tolerated, that is, the behaviour of faulty processes is arbitrary.

The communication network is assumed to be reliable and the communication delays are bounded. The minimal and maximal transmission delay are specified using two constants \( \delta \) and \( \varepsilon \) such that \( 0 \leq \varepsilon < \delta \): the delay for any message is between \( \delta - \varepsilon \) and \( \delta + \varepsilon \).

Each process has a physical clock which can drift slowly from real time at a rate bounded by a small constant \( \rho \) such that \( 0 < \rho \ll 1 \). If a clock \( C \) does not fail during a real time interval \([t_1, t_2]\) then

\[
(1 - \rho)(t_2 - t_1) \leq C(t_2) - C(t_1) \leq (1 + \rho)(t_2 - t_1),
\]

where \( C(t_1) \) and \( C(t_2) \) denote the value of clock \( C \) at time \( t_1 \) and \( t_2 \), respectively. The elapsed clock time \( C(t_2) - C(t_1) \) is within \( \rho(t_2 - t_1) \) of the real time delay \( t_2 - t_1 \). During the same interval, the physical clocks of two processes can drift apart by as much as \( 2\rho(t_2 - t_1) \). Even for small values of \( \rho \), the error may become significant for large values of \( t_2 \). In order to ensure that all the processes have a consistent view of time, it is necessary to regularly resynchronize their clocks.

We assume that a process \( p \) has no control over its physical clock \( PC_p \). Instead, the local time for \( p \) is given by a virtual clock \( VC_p \) obtained by adding a correction to \( PC_p \). The correction is periodically computed by \( p \) and is stored in a local variable \( CORR_p \). The virtual clock of process \( p \) is then defined by

\[
VC_p(t) = PC_p(t) + CORR_p(t),
\]

where \( CORR_p(t) \) denotes the content of the correction variable at real time \( t \).

The algorithm runs in successive rounds during which processes exchange information about their clocks and perform a correction to their local clock. Initially,
the virtual clocks of non-faulty processes are approximately synchronized: all the non-faulty processes start the first round within a delay $\beta$ of each other. Under this assumption, the algorithm ensures the following properties:

- **Agreement:** The skew, that is, the difference between the virtual clocks of any two non-faulty processes at any real time is bounded. There is a constant $\gamma$ such that, for all real time $t$ and all non-faulty processes $p$ and $q$,

$$|VC_p(t) - VC_q(t)| \leq \gamma.$$

- **Validity:** The clocks of non-faulty processes are within a linear envelope of real-time.

The purpose of this note is to give a simplified proof of correctness of the Welch-Lynch algorithm and to provide tight synchronization bounds. The algorithm is described in section 2 and the proof of correctness is given in section 3.

## 2 Algorithm

The algorithm of Welch and Lynch is similar to the interactive convergence algorithm of Lamport and Melliar-Smith [1]. Every non-faulty process $p$ reads the clocks of all the other processes at regular intervals. From these readings, $p$ obtains an estimate of the drift between its virtual clock and the clocks of the other processes. A correction to $p$’s local clock is then computed by applying a fault-tolerant averaging function to the estimates. The two algorithms differ only in the methods of reading clocks and in the averaging functions used. Both assume that the clocks are synchronized initially.

The processes are numbered from 1 to $n$. We denote by $f$ the maximal number of faults the algorithm can tolerate; by assumption, we have $n \geq 3f + 1$. The averaging function used in the algorithm of Welch and Lynch is the fault-tolerant midpoint\(^1\) defined as follows. Given an array $A$ of $n$ real numbers, the fault-tolerant midpoint of $A$, denoted by $\text{cfn}(A)$, is obtained by discarding the $f$ largest and the $f$ smallest elements of $A$ and by taking the arithmetic mean of the maximum and minimum of the remaining elements. If $A[1] \leq A[2] \leq \ldots \leq A[n]$, we then have

$$\text{cfn}(A) = \frac{A[f+1] + A[n-f]}{2}.$$  

For an arbitrary array, $\text{cfn}(A)$ can be obtained by first sorting the elements in increasing order and then applying the formula above.

Figure 1 gives an informal description of the algorithm. A process $p$ executes a main program which consists of repeatedly broadcasting a message $\text{SYNC}$ and waiting for a delay $\Delta$ before computing a correction to the local clock. In parallel, every process stores the arrival time of any $\text{SYNC}$ message it receives in an array $\text{ARR}_p$. The arrival times and the delays are of course measured with respect to the local clock $VC_p$. Two local variables are used in addition to $\text{ARR}_p$ and $\text{CORR}_p$: $\text{ADJ}_p$ is the clock adjustment and $T$ indicates the time of the next broadcast.

The two parameters $T_0$ and $P$ determine when the broadcasts take place: the first broadcast is executed at local time $T_0$ and the subsequent ones at $T_0 + P$, $T_0 + 2P$, etc. The parameter $\Delta$ determines how long a process has to wait after a broadcast before performing the clock correction.

For simplicity, we assume that broadcasting a message, computing the adjustment, and storing arrival times are instantaneous operations. If two $\text{SYNC}$ messages

\(^1\) We use the terminology of [3].
\[ T := T_0 \]

repeat forever
  wait until \( VC_p = T \);
  broadcast \( SYNC \);
  wait for \( \Delta \) time units;
  \( ADJ_p := T + \delta - cfn(ARR_p) \);
  \( CORR_p := CORR_p + ADJ_p \);
  \( T := T + P \)
end of loop.

on reception of \( SYNC \) from \( q \) do \( ARR_p[q] := VC_p \).

Figure 1: Pseudo Code for Process \( p \)

are received from two processes simultaneously, the corresponding elements of \( ARR_p \) are then equal. Also, the clock adjustment operations are executed instantaneously when the delay \( \Delta \) has elapsed. The correction takes effect immediately after this delay, that is, at the end of the loop. We also assume that broadcast messages are received by every processes, including the sender.

For a correct execution of the algorithm, \( P \) and \( \Delta \) have to satisfy several conditions which depend on the network and clock parameters (i.e. \( \delta, \varepsilon, \) and \( \rho \)) and on the degree of synchronization required. These constraints are obtained by a formal analysis of the algorithm and will be specified precisely in the sequel.

Let \( T = T_0 + iP \) denote the starting time of an arbitrary round \( i \). All the non-faulty processes broadcast \( SYNC \) when their local clocks reads \( T \) and wait until \( T + \Delta \) to compute the adjustment to their clock.

Assume \( p \) and \( q \) are two non-faulty processes. Let \( u_p \) be the real-time when \( p \) adjust its clock, that is, \( VC_p(u_p) = T + \Delta \). Let \( x \) be the arrival time at \( p \) of the message sent by \( q \) at the start of the round. The constraints on \( \Delta \) ensure that \( VC_p(x) \leq T + \Delta \), or equivalently that \( x \leq u_p \). Other assumptions on \( P \) imply that the next message from \( q \) is received by \( p \) after \( u_p \). This means that at time \( u_p \), the element \( ARR_p[q] \) is equal to \( VC_p(x) \). The value \( ARR_p[q] \) is used by \( p \) to estimate the drift between its local clock and the clock of \( q \). The message was sent when \( VC_q \) was equal to \( T \) and took a delay between \( \delta - \varepsilon \) and \( \delta + \varepsilon \) to reach \( p \). During the interval, \( VC_q \) has progressed to a value which is around \( T + \delta \), that is, \( VC_q(x) \approx T + \delta \). Therefore \( p \) can estimate that the difference between \( VC_q(x) \) and \( VC_p(x) \) is approximately \( (T + \delta) - ARR_p[q] \). Since \( \rho \) is very small, the drift between \( VC_p \) and \( VC_q \) remains fairly constant until \( p \)'s clock is adjusted:

\[
VC_q(u_p) - VC_p(u_p) \approx (T + \delta) - ARR_p[q].
\]

The accuracy of this estimate depends on the imprecision \( \varepsilon \) on transmission delays and on the rate \( \rho \) of clock drift.

When \( p \) has received a \( SYNC \) message from all the non-faulty processes, it can compute the correction to its clock. The adjustment \( ADJ_p \) is the fault-tolerant average of the estimated drifts:

\[
ADJ_p = T + \delta - cfn(ARR_p).
\]

If \( cfn(ARR_p) \) is larger than \( T + \delta \) then \( p \)'s clock is currently ahead of the average. Conversely, if \( cfn(ARR_p) \) is smaller than \( T + \delta \) then \( VC_p \) is behind the average.
The variable $CORR_p$ is then updated to cancel the average drift:

$$CORR_p := CORR_p + ADJ_p.$$ 

The virtual clock $VC_p$ is set back or forth by the amount $|ADJ_p|$. The function $efn$ is essential to the correctness of the algorithm. It ensures that the clock adjustment $ADJ_p$ is fairly insensitive to the presence of faulty elements in the array $ARR_p$. Furthermore, $efn$ has an averaging effect which implies that after the adjustments, the clocks $VC_p$ and $VC_q$ of two non-faulty processes are better synchronized than they were at the start of the round.

Between two successive resynchronization, the virtual clocks can drift apart from one another but adjusting the clocks sufficiently often ensure that the skew is bounded. The algorithm assumes that all the non-faulty processes start the first round within a real-time delay $\beta$ of one another. The values of $\Delta$ and $P$ are determined from $\beta$ in order to ensure that the non-faulty processes also start the other rounds within $\beta$ of one another.

From the latter invariant, a bound on the worst case skew can be derived. The clock adjustment computed during each round is small in comparison with the length of each round and this ensures that the virtual clocks are within a linear envelope of real-time.

## 3 Formal Analysis

### 3.1 Overview

We represent both real time and clock time by the reals. By convention, lowercase letters are used to denote real time quantities and uppercase letters to denote clock times. Clocks are defined as follows:

**Definition 1** A clock $C$ is a mapping from the reals to the reals such that, for all $t_1$ and $t_2$, if $t_1 \leq t_2$ then

$$(1 - \rho)(t_2 - t_1) \leq C(t_2) - C(t_1) \leq (1 + \rho)(t_2 - t_1).$$

We assume that every process $p$ has a clock $PC_p$ which satisfies the above constraint; the rate of drift of $PC_p$ is no more than $\rho$ over the interval $[t_1, t_2]$, whatever $t_1$ and $t_2$. This means that the physical clocks are assumed to be reliable; only processes can fail. These is no loss of generality because the behavior of faulty processes is arbitrary and because a process cannot access another process’s clock directly.

The crucial part of the analysis is to examine the effect of a single resynchronization round on the virtual clocks. Assume a non-faulty process $p$ starts a round at real-time $t_p$ such that $VC_p(t_p) = T$, performs the correction at real-time $u_p$ such that $VC_p(u_p) = T + \Delta$, and starts the subsequent round at $t'_p$ such that $VC_p(t'_p) = T + P$. Let $corr_p$ and $corr'_p$ denote the value of the variable $CORR_p$ at time $t_p$ and $t'_p$, respectively. We have assumed that the clock adjustment takes effect immediately after $u_p$, so

$$VC_p(t) = \begin{cases} PC_p(t) + corr_p & \text{if } t_p \leq t \leq u_p, \\ PC_p(t) + corr'_p & \text{if } u_p < t \leq t'_p. \end{cases}$$

The round can then be split in two parts. From $t_p$ to $u_p$, $p$’s local time is given by the clock $C_p$ such that:

$$C_p(t) = PC_p(t) + corr_p,$$
and, from \( u_p \) to \( t'_p \), \( p \)'s local time is given by \( C'_p \) defined by:

\[
C'_p(t) = PC_p(t) + \text{corr}'_p.
\]

The two successive clocks are related by the equation

\[
C'_p(t) = C_p(t) + T + \delta - \text{cfn}(ARR_p),
\]

where the value of the array \( ARR_p \) is taken at time \( u_p \).

Two non-faulty processes \( p \) and \( q \) switch then from old clocks \( C_p \) and \( C_q \) to new clocks \( C'_p \) and \( C'_q \) during the round. Since \( C_p(t_p) = C_q(t_q) = T \), the distance \(|t_p - t_q|\) gives a measure of the degree of synchronization between \( C_p \) and \( C_q \). Similarly, we can evaluate the degree of synchronization of \( C'_p \) and \( C'_q \) by measuring the distance \(|v_p - v_q|\) for two points \( v_p \) and \( v_q \) such that \( C'_p(v_p) = C'_q(v_q) \). Since the clock corrections are based on estimates for clock times which are close to \( T + \delta \), it is natural to choose \( v_p \) and \( v_q \) such that

\[
C_p(v_p) = C_q(v_q) = T + \delta.
\]

In the first part of the proof, we establish the following fundamental result. If for a given \( \beta \) and for all non-faulty processes \( p \) and \( q \), we have

\[
|t_p - t_q| \leq \beta
\]

then we also have

\[
|v_p - v_q| \leq (1 + \rho)\beta^2 + 2\varepsilon,
\]

for all non-faulty \( p \) and \( q \). This essential property shows that the new clocks are more closely synchronized with each other than the old ones.

The following section lists various lemmas about clocks which are used in the sequel. The essential synchronization property is proved in section 3.3. In section 3.4, we derive constraints on the parameters \( \Delta \) and \( P \) for the algorithm to execute properly and achieve a given synchronization bound \( \beta \). If the conditions are satisfied then the algorithm guarantees that all the non-faulty processes start each round within a real-time delay \( \beta \) of one another. The constraints on \( \Delta \) and \( P \) have a solution provided \( \beta \) is larger than an optimal \( \beta_{\text{min}} \), which is equal to approximately \( 4\varepsilon \). The worst-case skew is determined in Sect. 3.5, using the assumptions on \( \Delta \) and \( P \). Section 3.6 shows that the virtual clocks of non-faulty processes are within two linear functions of real-time.

### 3.2 Clock Properties

It is easy to see that a clock is strictly increasing, continuous, and not bounded. For any real \( T \), there is a unique \( t \) such that \( C(t) = T \). The lemma below is another easy consequence of the definition:

**Lemma 1** For a clock \( C \) and two reals \( t_1, t_2 \) such that \( t_1 \leq t_2 \),

\[
\frac{C(t_2) - C(t_1)}{1 + \rho} \leq t_2 - t_1 \leq \frac{C(t_2) - C(t_1)}{1 - \rho}.
\]

The following lemma is important for proving the resynchronization property. It shows that, if \( C(v) \) is the mean of \( C(t) \) and \( C(u) \) then \( v \) is very close to \((t + u)/2\).
Lemma 2 If \( t \leq u \) and \( C(v) = \frac{1}{2}(C(t) + C(u)) \) then
\[
\frac{t + u - \rho}{2} \leq v \leq \frac{t + u}{2} + \frac{u - t}{2}.
\]

**Proof:** Let \( X = (C(u) - C(t))/2 \) so that \( C(v) - C(t) = C(u) - C(v) = X \). Since \( C \) is increasing, we have \( t \leq v \leq u \) and Lemma 1 applied twice gives
\[
t + X/(1 + \rho) \leq v \leq t + X/(1 - \rho),
\]
\[
u - X/(1 - \rho) \leq v \leq u - X/(1 + \rho).
\]

Two cases can be distinguished:

1. If \( X \leq (1 - \rho^2)(u - t)/2 \), we use
\[
u - X/(1 - \rho) \leq v \leq t + X/(1 - \rho).
\]

Since \( X/(1 - \rho) \leq (1 + \rho)(u - t)/2 \), we get
\[
u - (1 + \rho)(u - t)/2 \leq v \leq t + (1 + \rho)(u - t)/2
\]

which simplifies to
\[
(u + t)/2 - \rho(u - t)/2 \leq v \leq (u + t)/2 + \rho(u - t)/2.
\]

2. If \( X \geq (1 - \rho^2)(u - t)/2 \), we use
\[
t + X/(1 + \rho) \leq v \leq u - X/(1 + \rho)
\]

and \( X/(1 + \rho) \geq (1 - \rho)(u - t)/2 \) to obtain
\[
t + (1 - \rho)(u - t)/2 \leq v \leq u - (1 - \rho)(u - t)/2.
\]

By an elementary calculation, this gives the same relation as previously. The expected bound holds for \( v \) in both cases. \( \square \)

The bound is tight. \( X \) is comprised between \((1 - \rho)(u - t)/2 \) and \((1 + \rho)(u - t)/2 \) and it is possible to have \( X = (1 - \rho^2)(u - t)/2 \). In such a case, the distance between \( \nu \) and \((t + u)/2 \) can be equal to \( \rho(u - t)/2 \).

In the following lemma, two clocks \( C \) and \( C' \) are considered together with two reals \( t \) and \( t' \) such that \( C(t) = C'(t') = X \). The lemma gives a bound on the delay \( |u - u'| \) for \( u \) and \( u' \) such that \( C(u) = C(u') = X + Y \).

**Lemma 3** If \( C(t) = C'(t') = X \) and \( C(u) = C'(u') = X + Y \) where \( Y \geq 0 \) then
\[
|u - u'| \leq |t - t'| + \frac{2\rho}{1 - \rho^2} Y.
\]

The following lemma will be used to evaluate the skew between two clocks \( C \) and \( C' \).

**Lemma 4** Let \( X \) and \( Y \) be arbitrary reals. Given \( t \) and \( t' \) such that \( C(t) = C'(t') = X \) then, for any \( x \) such that \( X \leq C(x) \leq X + Y \) and \( X \leq C'(x) \leq X + Y \), we have
\[
|C(x) - C'(x)| \leq \frac{2\rho}{1 + \rho} Y + (1 - \rho) |t - t'|.
\]

This bound can be reached provided \( Y \geq (1 + \rho)|t - t'| \). The situation is illustrated in Fig. 2. The two points of coordinates \((x, C(x))\) and \((x, C'(x))\) are contained within the area delimited by the two oblique lines of slope \((1 + \rho)\) and \((1 - \rho)\). The bound can be attained at the point \( u \) such that \( u - t = Y/(1 + \rho) \). If \( Y < (1 - \rho)|t - t'| \), no \( x \) can satisfy the assumptions of the lemma and if \( Y \) is between \((1 - \rho)|t - t'| \) and \((1 + \rho)|t - t'| \), the skew is no more than \( Y \).
3.3 Resynchronization Property

3.3.1 Assumptions

We assume that an arbitrary real \( T \) and a set \( G \) of \( m \) processes are fixed, where \( m \geq n - f \). With every \( p \) of \( G \) are associated a clock \( C_p \) and a real \( t_p \) which satisfy

\[
C_p(t_p) = T. \tag{1}
\]

We also assume that two arrays \( arr_p \) and \( ARR_p \) are given for every \( p \) of \( G \). \( ARR_p \) and \( arr_p \) are two arrays of \( n \) reals and satisfy the two constraints below:

\[
\forall q \in G : t_q + \delta - \varepsilon \leq arr_p[q] \leq t_q + \delta + \varepsilon, \tag{2}
\]

\[
\forall q \in G : ARR_p[q] = C_p(arr_p[q]). \tag{3}
\]

We denote by \( C'_p \) the clock defined by

\[
C'_p(t) = C_p(t) + ADJ_p, \tag{4}
\]

where

\[
ADJ_p = T + \delta - cfn(ARR_p). \tag{5}
\]

Finally, we assume that a constant \( \beta \) gives a bound on the delay between \( t_p \) and \( t_q \) for \( p \) and \( q \) in \( G \):

\[
\forall p, q \in G : |t_p - t_q| \leq \beta. \tag{6}
\]

The intention is, of course, that these assumptions are satisfied if \( T \) is a clock time corresponding to the start of a round, \( G \) the set of processes which do not fail during that round, and \( C_p \) and \( C'_p \) are the virtual clocks of a process \( p \) at the beginning and at the end of the round, respectively. The array \( arr_p \) stores the arrival time of \( SYNC \) messages received by \( p \). If \( q \) is not faulty, the message sent by \( q \) to \( p \) at time \( t_q \) is received at time \( arr_p[q] \) and the corresponding clock time is given by \( ARR_p[q] \). For a faulty process \( r \), \( arr_p[r] \) and \( ARR_p[r] \) are arbitrary.

For every \( p \) of \( G \), we denote by \( v_p \) the time when \( C'_p \) reaches \( T + \delta \). The main objective of this section is to estimate the distance \( |v_p - v_q| \) where \( p \) and \( q \) are arbitrary processes of \( G \). We also bound the clock adjustment \( ADJ_p \).
3.3.2 Bounding $v_p$

Let $p$ be an arbitrary element of $G$. We have $C'_p(v_p) = T + \delta$ so, using equations (4) and (5),

$$C_p(v_p) = cfn(ARR_p).$$

Let $A = (A_1, \ldots, A_n)$ be the $n$-tuple formed by sorting the elements of $ARR_p$ in increasing order. By definition of $cfn$, we have

$$C_p(v_p) = \frac{A_{f+1} + A_{n-f}}{2}. \quad (7)$$

Similarly, let $a = (a_1, \ldots, a_m)$ be the tuple obtained by sorting in increasing order the $m$ elements of $arr_p$ whose index belongs to $G$. We then have $a_1 \leq a_2 \leq \ldots \leq a_m$, and each $a_i$ is equal to $arr_p[q]$ for some element $q$ of $G$.

Proposition 5

$$C_p(a_1) \leq A_{f+1} \leq C_p(a_{f+1})$$

$$C_p(a_{n-f}) \leq A_{n-f} \leq C_p(a_m).$$

Proof: Since $C_p$ is increasing, we have $C_p(a_1) \leq C(a_2) \leq \ldots \leq C_p(a_m)$. By construction, $C_p(a_1), \ldots, C_p(a_m)$ is then a subsequence of $A_1, \ldots, A_n$, obtained by removing fewer than $f$ elements.

$C_p(a_i)$ is equal to $A_i$ for some index $i$. There are at least $m$ elements among $A_1, \ldots, A_n$ which are larger than or equal to $A_i$ so $i$ must be smaller than or equal to $n + 1 - m$. By the assumptions on $n$ and $m$, this implies that $i \leq f + 1$ and then

$$C_p(a_1) = A_1 \leq A_{f+1}.$$  

Similarly, there are at least $f + 1$ elements among $A_1, \ldots, A_n$ which are smaller than or equal to $C_p(a_{f+1})$ so

$$A_{f+1} \leq C_p(a_{f+1}).$$

A symmetric reasoning proves the other part of the proposition. □

Now, let $k$ be any index between $f + 1$ and $m - f$; since $m \geq 2f + 1$, such a $k$ does exist. As a consequence of the previous proposition, we get

$$C_p(a_1) \leq A_{f+1} \leq C_p(a_k) \leq A_{n-f} \leq C_p(a_m)$$

because $a_{f+1} \leq a_k \leq a_{n-f}$. Using (7), we can then bound $C_p(v_p)$ as follows:

$$\frac{C_p(a_1) + C_p(a_k)}{2} \leq C_p(v_p) \leq \frac{C_p(a_k) + C_p(a_m)}{2}. \quad (8)$$

From these two bounds and Lemma 2, we obtain:

Proposition 6

$$\frac{a_1 + a_k}{2} - \rho \frac{a_k - a_1}{2} \leq v_p \leq \frac{a_k + a_m}{2} + \rho \frac{a_m - a_k}{2}.$$  

This proposition and relation (8) explain why the algorithm is fault-tolerant. The midpoint $cfn(ARR_p)$ is equal to $C_p(v_p)$ and is fairly insensitive to possibly wide variation in $f$ of the array elements. At worst, $cfn(ARR_p)$ can be shifted towards
the lower or the upper ends of the interval given by relation (8). The two extremities of the interval only depend on the values of $ARR_p$ for non-faulty processes.

If follows immediately from the fact that $C_p$ is increasing and from relation (8) that $a_1 \leq v_p \leq a_m$. The two reals $a_1$ and $a_m$ are the smallest and largest of the elements $arr_p[r]$ for $r \in G$. Let $t_{\min}$ and $t_{\max}$ be the smallest and largest of the times $t_r$ for $r \in G$. By (2),

$$
a_1 \geq t_{\min} + \delta - \varepsilon
$$

$$
a_m \leq t_{\max} + \delta + \varepsilon,
$$

and then

$$
t_{\min} + \delta - \varepsilon \leq v_p \leq t_{\max} + \delta + \varepsilon. \quad (9)
$$

By (6), we also obtain for any non-faulty process $q$,

$$
t_q - \beta + \varepsilon \leq v_p \leq t_q + \beta + \varepsilon. \quad (10)
$$

This relation holds for arbitrary $q$, in particular, in the case $q = p$. It will be used to bound the clock adjustment and determine a lower bound on $\Delta$.

### 3.3.3 Bounding $|v_p - v_q|$

Assume $p$, $k$, and $a = (a_1, \ldots, a_m)$ are defined as in the previous section. Let $q$ be another element of $G$ and let $b = (b_1, \ldots, b_m)$ be formed by sorting in increasing order the elements $arr_q[r]$ for $r \in G$. The tuple $b$ is then obtained from $arr_q$ in the same way as $a$ is obtained from $arr_p$. Proposition 6 gives the following bounds for $v_p$ and $v_q$:

$$
\frac{a_1 + a_k - \rho a_k - a_1}{2} \leq v_p \leq \frac{a_k + a_m - a_k}{2} + \rho \frac{a_m - a_k}{2},
$$

$$
\frac{b_1 + b_k - \rho b_k - b_1}{2} \leq v_q \leq \frac{b_k + b_m - b_k}{2} + \rho \frac{b_m - b_k}{2}.
$$

These bounds imply that

$$
v_p - v_q \leq (1 + \rho) \frac{a_m - b_1}{2} + (1 - \rho) \frac{a_k - b_k}{2} \quad (11)
$$

and, symmetrically,

$$
v_q - v_p \leq (1 + \rho) \frac{b_m - a_1}{2} + (1 - \rho) \frac{b_k - a_k}{2} \quad (12)
$$

In order to evaluate the difference $v_p - v_q$ we have to compare $a_k$ and $b_k$. We need the following lemma.

**Lemma 7** Let $d_1, \ldots, d_l$ and $e_1, \ldots, e_l$ be two finite sequences of reals, such that $d_1 \leq d_2 \leq \ldots \leq d_l$ and $e_1 \leq e_2 \leq \ldots \leq e_l$. If there is a number $x$ and a bijection $h$ from $\{1, \ldots, l\}$ to $\{1, \ldots, l\}$ such that

$$
|d_i - e_{h(i)}| \leq x \quad \text{for } i = 1, \ldots, l,
$$

then we also have

$$
|d_i - e_i| \leq x \quad \text{for } i = 1, \ldots, l.
$$


\textbf{Proof:} We reason by induction on \(l\). For the base case, \(l = 0\), the property is vacuously true. For the inductive case, assume \(d_1, \ldots, d_{l+1}\) and \(e_1, \ldots, e_{l+1}\) are two ordered sequences and \(h\) and \(x\) satisfy the assumption. Let \(r = h(l + 1)\) and \(s\) be such that \(h(s) = l + 1\); we have
\[
|d_{l+1} - e_r| \leq x, \quad |d_s - e_{l+1}| \leq x, \quad d_s \leq d_{l+1}, \quad \text{and} \quad e_r \leq e_{l+1}.
\]
From these four inequalities, it is easy to see that
\[
|d_s - e_r| \leq x \quad \text{and} \quad |d_{l+1} - e_{l+1}| \leq x.
\]
Consider the mapping \(h'\) defined for \(i = 1, \ldots, l\) by
\[
h'(i) = \begin{cases} r & \text{if } i = s \\ h(i) & \text{otherwise.} \end{cases}
\]
It is clear that \(h'\) is a bijection from \(\{1, \ldots, l\}\) to \(\{1, \ldots, l\}\); in particular, if \(r = l + 1\), \(h'\) is the restriction of \(h\) to \(\{1, \ldots, l\}\). We also have
\[
|d_i - e_{h'(i)}| \leq x \quad \text{for } i = 1, \ldots, l,
\]
so we can apply the induction hypothesis. This gives \(|d_i - e_i| \leq x\) for \(i = 1, \ldots, l\) and the inequality also holds for \(i = l + 1\) as shown above. \(\square\)

As a consequence, we obtain the following property.

\textbf{Proposition 8} For all \(i\) such that \(1 \leq i \leq m\), \(|a_i - b_i| \leq 2\varepsilon\).

\textbf{Proof:} Since \((a_1, \ldots, a_m)\) is a permutation of the elements \(\text{arr}_p[r]\) for \(r \in G\), there is a bijection \(g\) from \(\{1, \ldots, m\}\) to \(G\), such that
\[
a_i = \text{arr}_p[g(i)] \quad \text{for } i = 1, \ldots, m.
\]
Similarly, there is a bijection \(h\) from \(\{1, \ldots, m\}\) to \(G\) such that
\[
b_i = \text{arr}_q[h(i)] \quad \text{for } i = 1, \ldots, m.
\]
The composite \(g' = h^{-1} \circ g\) is a bijection from \(\{1, \ldots, m\}\) to \(\{1, \ldots, m\}\) and
\[
|a_i - b_{g'(i)}| = |\text{arr}_p[g(i)] - \text{arr}_q[g(i)]| \quad \text{for } i = 1, \ldots, m.
\]
By assumption (2), \(|\text{arr}_p[r] - \text{arr}_q[r]| \leq 2\varepsilon\) for any \(r \in G\). It follows that
\[
|a_i - b_{g'(i)}| \leq 2\varepsilon \quad \text{for } i = 1, \ldots, m
\]
and Lemma 7 gives
\[
|a_i - b_i| \leq 2\varepsilon \quad \text{for } i = 1, \ldots, m. \quad \square
\]

We can now bound the difference \(|v_p - v_q|\) as follows.

\textbf{Theorem 9}

\[
|v_p - v_q| \leq (1 + \rho) \frac{\beta}{2} + 2\varepsilon.
\]
**Proof:** Proposition 8 implies that \( (a_k - b_k) \leq 2\varepsilon \). By relation (11), we then have

\[
v_p - v_q \leq (1 + \rho) \frac{a_m - b_1}{2} + (1 - \rho)\varepsilon.
\]

Now for any two elements \( r \) and \( s \) of \( G \), the two assumptions (2) and (6) mean that

\[
arr_p[r] - arr_p[s] \leq t_r - t_s + 2\varepsilon \leq \beta + 2\varepsilon.
\]

This holds for arbitrary \( r \) and \( s \) so \( a_m - b_1 \leq \beta + 2\varepsilon \) and then

\[
v_p - v_q \leq (1 + \rho) \frac{\beta + 2\varepsilon}{2} + (1 - \rho)\varepsilon
\]

\[
\leq (1 + \rho) \frac{\beta}{2} + 2\varepsilon.
\]

By symmetry, we can derive from relation (12) that

\[
v_q - v_p \leq (1 + \rho) \frac{\beta}{2} + 2\varepsilon. \quad \Box
\]

### 3.3.4 Bounding the Clock Adjustment

For a process \( p \), the clock adjustment \( ADJ_p \) is the difference \( C_p^r(t) - C_p(t) \), which is constant for any \( t \). For \( t = v_p \), we have \( C_p(v_p) = T + \delta \). As noted previously,

\[
t_p - \beta + \delta - \varepsilon \leq v_p \leq t_p + \beta + \delta + \varepsilon,
\]

then

\[
C_p(t_p - \beta + \delta - \varepsilon) \leq C_p(v_p) \leq C_p(t_p + \beta + \delta + \varepsilon).
\]

The adjustment is then between the two limits below:

\[
T + \delta - C_p(t_p + \beta + \delta + \varepsilon) \leq ADJ_p \leq T + \delta - C_p(t_p - \beta + \delta - \varepsilon).
\]

For the lower bound, we get

\[
C_p(t_p + \beta + \delta + \varepsilon) \leq C_p(t_p) + (1 + \rho)(\beta + \delta + \varepsilon),
\]

then, since \( C_p(t_p) = T \),

\[
-(1 + \rho)(\beta + \varepsilon) - \rho\delta \leq ADJ_p.
\]

The upper bound depends on whether \( \beta \) is smaller or larger than \( \delta - \varepsilon \). If \( \beta \leq \delta - \varepsilon \), we obtain

\[
C_p(t_p - \beta + \delta - \varepsilon) - C_p(t_p) \geq (1 - \rho)(-\beta + \delta - \varepsilon),
\]

then

\[
ADJ_p \leq (1 - \rho)(\beta + \varepsilon) + \rho\delta.
\]

In the other case,

\[
C_p(t_p) - C_p(t_p - \beta + \delta - \varepsilon) \leq (1 + \rho)(\beta - \delta + \varepsilon),
\]

and

\[
ADJ_p \leq (1 + \rho)(\beta + \varepsilon) - \rho\delta.
\]

In both cases, we have

\[
ADJ_p \leq (\beta + \varepsilon) + \rho |\beta - \delta + \varepsilon|,
\]

and the clock adjustment is between the two bounds given by the following property.

**Proposition 10**

\[
-(\beta + \varepsilon) - \rho (\beta + \delta + \varepsilon) \leq ADJ_p \leq (\beta + \varepsilon) + \rho |\beta - \delta + \varepsilon|.
\]
Assumptions:

- $|G| = m$ and $m \geq n - f$.
- $C_p(t_p) = T$.
- $\forall p, q \in G : |t_p - t_q| \leq \beta$.
- $\forall p, q \in G : t_q + \delta - \varepsilon \leq arr_p[q] \leq t_q + \delta + \varepsilon$.
- $\forall p, q \in G : ARR_p[q] = C_p(arr_p[q])$.
- $ADJ_p = T + \delta - \text{cfn}(ARR_p)$.
- $\forall t : C'_p(t) = C_p(t) + ADJ_p$.
- $C'_p(v_p) = T + \delta$.

Results:

- For any $p \in G$ and $q \in G$,
  \[|v_p - v_q| \leq (1 + \rho) \frac{\beta}{2} + 2\varepsilon.
  \]
  \[t_q - \beta + \delta - \varepsilon \leq v_p \leq t_q + \beta + \delta + \varepsilon.
  \]
- For any $p \in G$,
  \[-(\beta + \varepsilon) - \rho(\beta + \delta + \varepsilon) \leq ADJ_p \leq (\beta + \varepsilon) + \rho|\beta - \delta + \varepsilon|.

Figure 3: Results of Section 3.3

3.3.5 Summary

The main results obtained in this section are summarized in Fig. 3.

3.4 Algorithm Parameters

3.4.1 Constraints on $\Delta$ and $P$

From now on, a constant $\beta$ is fixed which specifies the degree of synchronization to maintain. We assume that non-faulty processes start a round within a real-time delay $\beta$ of one another and we want to ensure that the non-faulty processes also start the next round within $\beta$ of one another. At the start of the first round, the local time of each process is equal to $T$ and the next round starts at local time $T + P$.

Using the same notations as previously, a process $p$ can compute $\text{cfn}(ARR_p)$ as soon as it knows $A_{f+1}$ and $A_{n-f}$. In the worst case, $p$ has to wait for the last message coming from a process in $G$, that is, until time $a_m$. Process $p$ executes the clock adjustment procedure a delay $\Delta$ after the start of a round, at real-time $u_p$ such that $C_p(u_p) = T + \Delta$. For the adjustment to be correctly computed, we must make sure that $a_m \leq u_p$. As shown previously, $a_m \leq t_p + \beta + \delta + \varepsilon$, and
\[ C_p(t_p + \beta + \delta + \varepsilon) \leq T + (1 + \rho)(\beta + \delta + \varepsilon). \] It is sufficient to take \( \Delta \) such that

\[ \Delta \geq (1 + \rho)(\beta + \delta + \varepsilon). \tag{13} \]

This ensures that \( u_p \geq t_p + \beta + \delta + \varepsilon \) which also implies \( u_p \geq v_p \) and \( u_p \geq t_q + \delta + \varepsilon \) for any non-faulty process \( q \).

At time \( u_p \), \( p \)'s local clock is equal to \( T + \Delta \) and the adjustment \( ADJ_p \) can be at most \((\beta + \varepsilon) + \rho |\beta - \delta + \varepsilon|\). For \( p \) not to miss the next round, \( T + P \) must be larger than the new clock at the time of the correction. A lower bound for \( P \) is then

\[ P \geq \Delta + (\beta + \varepsilon) + \rho |\beta - \delta + \varepsilon|. \tag{14} \]

Assuming this condition is satisfied, \( p \) starts the next round at time \( t'_p \) such that \( C'_p(t'_p) = T + P \). Let \( q \) be another non-faulty process and \( u_q \) be such that \( C_q(u_q) = T + \Delta \). The clock correction computed by \( q \) at time \( u_q \) assumes that \( ARR_q[p] \) is the arrival time of the message broadcast by \( p \) at time \( t_p \). The message sent by \( p \) at time \( t'_q \) must not arrive at \( q \) before time \( u_q \). \( P \) must then be large enough to ensure

\[ t'_p + \delta - \varepsilon > u_q. \]

Since \( C_q(u_q) - C_q(t_q) = \Delta \), Lemma 1 yields:

\[ u_q \leq t_q + \frac{\Delta}{1 - \rho}. \]

We also have \( C'_p(t'_p) - C'_q(v_p) = P - \delta \) which is positive by (14). Using Lemma 1 again, we obtain

\[ t'_p \geq v_p + \frac{P - \delta}{1 + \rho}. \]

By (10), \( v_p \geq t_q - \beta + \delta - \varepsilon \), therefore

\[ t'_p + \delta - \varepsilon \geq t_q - \beta + 2\delta - 2\varepsilon + \frac{P - \delta}{1 + \rho}. \]

A sufficient condition to ensure \( t'_p + \delta - \varepsilon > u_q \) is then

\[ \frac{P - \delta}{1 + \rho} - \beta + 2\delta - 2\varepsilon > \frac{\Delta}{1 - \rho}, \]

or, equivalently,

\[ P > (1 + \rho)(\beta + 2\varepsilon) - (1 + 2\rho)\delta + \frac{1 + \rho}{1 - \rho} \Delta. \tag{15} \]

Depending on the value of \( \Delta, \beta \) and the network parameters, this bound may be larger or smaller than the bound given by (14).

If two non-faulty processes \( p \) and \( q \) start a round at local time \( T \) as measured by their respective clocks \( C_p \) and \( C_q \) then the next round will start at real-time \( t'_p \) and \( t'_q \) such that

\[ C'_p(v_p') = T + P \quad \text{and} \quad C'_q(v_q') = T + P. \]

By relations (13) and (14), \( P \) is larger than \( \delta \). We also have that \( C'_p(v_p) = C'_q(v_q) = T + \delta \) then using Lemma 3 and Theorem 9, we obtain

\[ |t'_p - t'_q| \leq (1 + \rho) \frac{\beta}{2} + 2\varepsilon + (P - \delta) \frac{2\rho}{1 - \rho}. \]
\[ \Delta \geq (1 + \rho)(\beta + \delta + \varepsilon). \]
\[ P \geq \Delta + (\beta + \varepsilon) + \rho|\beta - \delta + \varepsilon|. \]
\[ P > (1 + \rho)(\beta + 2\varepsilon) - (1 + 2\rho)\delta + \frac{1 + \rho}{1 - \rho} \Delta. \]
\[ P - \delta \leq \frac{1 - \rho^2}{\rho} \left[ (1 - \rho)\frac{\beta}{4} - \varepsilon \right]. \]

Figure 4: Constraints on the Parameters.

For the invariant to be maintained, the delay \(|t'_p - t'_q|\) must be smaller than \(\beta\). This requires the following condition to be satisfied
\[ (P - \delta) - \frac{2\rho}{1 - \rho^2} \leq (1 - \rho)\frac{\beta}{2} - 2\varepsilon. \] (16)

The upper bound for \(P\) is then given by
\[ P - \delta \leq \frac{1 - \rho^2}{\rho} \left[ (1 - \rho)\frac{\beta}{4} - \varepsilon \right]. \] (17)

In summary, the lower bounds for \(\Delta\) and \(P\) ensure that for all \(p\) of \(G\), the value of \(ARR_p\) at time \(u_p\) satisfies assumptions (2) and (3). The message broadcast by \(q\) at time \(t_q\) is received by \(p\) before \(u_p\) and no other message from \(q\) is received by \(p\) until after \(u_p\). The upper bound on \(P\) ensures that the clock resynchronizations are performed sufficiently often.

The lower bounds for \(\Delta\) and \(P\) also guarantee that \(u'_p > u_q\) for all non-faulty \(p\) and \(q\) because \(u'_p > t'_p + \delta + \beta + \varepsilon\) and \(u_q < t'_q + \delta - \varepsilon\). By Proposition 10 and the constraint on \(\Delta\), we have \(C_p(u_p) \geq T + \delta\). The clock correction can set \(p\)'s virtual clock to a value smaller than \(T + \Delta\) but more than \(T + \delta\).

### 3.4.2 Optimal Synchronization

Figure 4 shows the necessary conditions on \(\Delta\) and \(P\) to ensure that a synchronization bound \(\beta\) is maintained. The constraints can be satisfied if the two lower bounds for \(P\) obtained from relations (14) and (15) by setting \(\Delta = (1 + \rho)(\beta + \delta + \varepsilon)\) are smaller than the upper bound given by (17). This requirement is equivalent to the three following constraints
\[ (1 - 11\rho + 3\rho^2 - \rho^3)\frac{\beta}{4} \geq (1 + \rho)\varepsilon - \rho(1 - 3\rho)\delta \] (18)
\[ (1 - 9\rho - \rho^2 + \rho^3)\frac{\beta}{4} \geq (1 + 2\rho - \rho^2)\varepsilon + 2\rho^2\delta \quad \text{if } \beta \leq \delta - \varepsilon \] (19)
\[ (1 - 10\rho + \rho^2)\frac{\beta}{4} \geq (1 + \rho)\varepsilon \quad \text{if } \beta \geq \delta - \varepsilon. \] (20)

For small values of \(\rho\) these constraints are satisfied in the following three cases:
\[ (5 - 6\rho + \rho^2)\varepsilon \leq (1 - 10\rho + \rho^2)\delta \]
\[ \beta \geq \frac{4(1 + 2\rho - \rho^2)\varepsilon + 8\rho^2\delta}{1 - 9\rho - \rho^2 + \rho^3}; \]
• If \((5 - 6\rho + \rho^2) \varepsilon \leq (1 - 10\rho + \rho^2) \delta\) then
\[
\beta_{\text{min}} = \frac{4(1 + 2\rho - \rho^2)\varepsilon + 8\rho^2\delta}{1 - 9\rho - \rho^2 + \rho^3}.
\]

• If \((1 + \rho)(1 - \rho)^2 \varepsilon \geq (1 - 3\rho)(1 - 10\rho + \rho^2)\delta\) then
\[
\beta_{\text{min}} = \frac{4(1 + \rho)\varepsilon - 4\rho(1 - 3\rho)\delta}{1 - 11\rho + 3\rho^2 - \rho^3}.
\]

• Otherwise
\[
\beta_{\text{min}} = \frac{4(1 + \rho)\varepsilon}{1 - 10\rho + \rho^2}.
\]

Figure 5: Optimal Synchronization Bound.

\[(5 - 6\rho + \rho^2) \varepsilon \geq (1 - 10\rho + \rho^2) \delta\]
\[(1 + \rho)(1 - \rho)^2 \varepsilon < (1 - 3\rho)(1 - 10\rho + \rho^2)\delta\]

\[\beta \geq \frac{4(1 + \rho)\varepsilon}{1 - 10\rho + \rho^2};\]

\[(1 + \rho)(1 - \rho)^2 \varepsilon \geq (1 - 3\rho)(1 - 10\rho + \rho^2)\delta\]

\[\beta > \frac{4(1 + \rho)\varepsilon - 4\rho(1 - 3\rho)\delta}{1 - 11\rho + 3\rho^2 - \rho^3}.
\]

These results are obtained by a routine but lengthy calculation and hold provided \(\rho\) is small enough\(^2\). For a fixed \(\rho\), the ratio \(\delta/\varepsilon\) determines which of the three above cases apply. Neglecting factors of degree 2 or more, the first case corresponds to \(\delta/\varepsilon \geq 5 + 44\rho\), the second to \(1 + 12\rho < \delta/\varepsilon < 5 + 44\rho\) and the last to \(1 \leq \delta/\varepsilon \leq 1 + 12\rho\). The parameter \(\beta\) can be smaller than \(\delta - \varepsilon\) only in the first case.

The smallest synchronization bound \(\beta_{\text{min}}\) which can be maintained by the algorithm is defined in Fig. 5. In practice, the drift rate of hardware clocks is very small; \(\rho\) is typically less than \(10^{-5}\) and factors such as \(\rho\varepsilon, \rho^2, \rho^2\delta, \text{etc.}\) are negligible. The optimal synchronization bound \(\beta_{\text{min}}\) is approximately \(4\varepsilon\) in all three cases and the corresponding resynchronization period \(P\) is approximately \(10\varepsilon + \delta\).

Conversely, for a fixed resynchronization period \(P\) larger than \(10\varepsilon + \delta\), the synchronization bound \(\beta\) given by (15) is very close to \(4\varepsilon + 4\rho(P - \delta + \varepsilon)\). If \(P\) is large and \(\rho(\delta - \varepsilon)\) is negligible in comparison to \(\rho P\), the bound is approximately \(4\varepsilon + 4\rho P\).

\(^2\text{Requiring } \rho \leq 10^{-2} \text{ is sufficient.}\)
3.5 Agreement

In this section, we assume that the parameters satisfy the constraints given in Fig. 4 and we examine the difference $|VC_p(t) - VC_q(t)|$ where $p$ and $q$ are non-faulty. Figure 6 illustrates how $VC_p$ and $VC_q$ can evolve from the start of a round until the clock correction of the subsequent round. Process $p$ and $q$ start the round at real times $t_p$ and $t_q$ and adjust their respective clock at $u_p$ and $u_q$. The next clock adjustments are performed at times $u_p'$ and $u_q'$.

In the interval $[t_p, u_p]$ the clock $VC_p$ is equal to $C_p$ and in $(u_p, u_p')$ it is equal to $C_p'$. Similarly, the clock $VC_q$ is equal to $C_q$ in $[t_q, u_q]$ and to $C_q'$ in $(u_q, u_q')$. Lemma 4 yields the following result.

Proposition 11

1. For $t \in [t_p, u_p] \cap [t_q, u_q]$,

\[ |C_p(t) - C_q(t)| \leq \frac{2\rho}{1 + \rho} \Delta + (1 - \rho) \beta. \]

2. For $t \in (u_p, u_p') \cap (u_q, u_q')$,

\[ |C_p'(t) - C_q'(t)| \leq \frac{2\rho}{1 + \rho} \Delta + (1 - \rho) \beta. \]

Proof. For $t \in [t_p, u_p] \cap [t_q, u_q]$, both $C_p(t)$ and $C_q(t)$ are in the interval $[T, T + \Delta]$. Lemma 4 together with the fact that $|t_p - t_q| \leq \beta$ gives the first part.

For the second part, we know that $u_p \geq v_p$ and $u_q \geq v_q$, therefore $t$ is larger than both $v_p$ and $v_q$. This means that $T + \delta \leq C_p'(t) \leq T + \Delta + P$ and $T + \delta \leq C_q'(t) \leq T + \Delta + P$. Lemma 4 can then be applied and we get

\[ |C_p'(t) - C_q'(t)| \leq \frac{2\rho}{1 + \rho} (P + \Delta - \delta) + (1 - \rho) \left( (1 + \rho) \frac{\beta}{2} + 2\epsilon \right). \]
By assumption, \( P \) satisfies inequality (16) so
\[
\frac{2\rho}{1 + \rho} (P - \delta) \leq \left( 1 - \rho \right) \left[ (1 - \rho) \frac{\beta}{2} - 2\varepsilon \right].
\]
The result follows from the last two relations. \( \square \)

The first part of this lemma is only used to estimate the worst case skew at the start of the algorithm, that is, for \( T = T_0 \). For all other values of \( T \), the interval \([t_p, u_p]\) is included in \([u_{p-1}, u_p]\). Using (14) and (13), it is readily verified that
\[
(P + \Delta - \delta) \geq (1 + \rho) \left[ (1 + \rho) \frac{\beta}{2} + 2\varepsilon \right].
\]
This means that the skew between \( C'_p \) and \( C'_q \) can be as large as the bound given in case 2 above.

We now consider the transient phase which occurs during a round when some processes have updated their clocks and others have not. For this intermediate phase, the worst case skew is given by the following proposition.

**Proposition 12** If \( u_p \leq t \leq u_q \) then
\[
|C'_p(t) - C'_q(t)| \leq \frac{2\rho}{1 - \rho} \Delta + (1 + \rho)(\beta + \varepsilon) - \rho \delta.
\]

**Proof:** The lower bound on \( \Delta \) ensures that \( u_p \geq v_p \) and \( u_p \geq t_q + \delta + \varepsilon \). We then have
\[
(1 - \rho)(t - v_p) \leq C'_p(t) - C'_p(v_p) \leq (1 + \rho)(t - v_p),
\]
\[
(1 - \rho)(t - t_q) \leq C'_q(t) - C'_q(t_q) \leq (1 + \rho)(t - t_q).
\]
Since \( C'_p(v_p) = T + \delta \) and \( C'_q(t_q) = T \), it follows that
\[
C'_p(t) - C'_q(t) \geq \delta + (1 - \rho)(t - v_p) - (1 + \rho)(t - t_q) \geq \delta - 2\rho(t - t_q) + (1 - \rho)(t_q - v_p) \geq \delta - 2\rho(t_q - v_p),
\]
and
\[
C'_q(t) - C'_p(t) \leq \delta + (1 + \rho)(t - v_p) - (1 - \rho)(t - t_q) \leq \delta + 2\rho(t - t_q) + (1 + \rho)(t_q - v_p) \leq \delta + 2\rho(u_q - t_q) + (1 + \rho)(t_q - v_p).
\]
Since \( C'_p(u_p) = T + \Delta \), we have
\[
u_q - t_q \leq \frac{\Delta}{1 - \rho},
\]
and by (10), we also get
\[-(\beta + \delta + \varepsilon) \leq t_q - v_p \leq \beta - \delta + \varepsilon.
\]
As a consequence, the difference \( C'_p(t) - C'_q(t) \) satisfies the two inequalities below
\[
C'_p(t) - C'_q(t) \geq -\frac{2\rho}{1 - \rho} \Delta - (1 - \rho)(\beta + \varepsilon) + \rho \delta,
\]
\[
C'_p(t) - C'_q(t) \leq \frac{2\rho}{1 - \rho} \Delta + (1 + \rho)(\beta + \varepsilon) - \rho \delta.
\]
• If $u_p < t \leq u'_p$ and $u_q < t \leq u'_q$ then

$$|C'_p(t) - C'_q(t)| \leq \frac{2\rho}{1 - \rho} \Delta + (1 - \rho) \beta.$$

• If $u_p \leq t \leq u_q$ then

$$|C'_p(t) - C_q(t)| \leq \frac{2\rho}{1 - \rho} \Delta + (1 + \rho)(\beta + \varepsilon) - \rho \delta.$$

Figure 7: Worst Case Skew.

The result follows by taking the absolute values of these bounds. □

Proposition 11 and 12 cover the two possible cases. The parameter constraints imply that $u_p < u'_p$ and $u_q < u'_q$. When $VC_p = C'_q$, i.e. between $u_p$ and $u'_q$, the virtual clock of $q$ is equal to either $C_q$ or $C'_q$. In the worst case, the skew between the clocks $VC_p$ and $VC_q$ is then bounded as shown in Proposition 12:

$$|VC_p(t) - VC_q(t)| \leq \frac{2\rho}{1 - \rho} \Delta + (1 + \rho)(\beta + \varepsilon) - \rho \delta.$$

The maximal skew can be attained if $t_q = t_p + \beta$ and the physical clock of $p$ and $q$ run at rate $(1 + \rho)$ and $(1 - \rho)$, respectively. In such a case, the skew at time $u_p$ is

$$VC_p(u_p) - VC_q(u_p) = C_p(u_p) - C_q(u_p)$$

$$= \frac{2\rho}{1 - \rho} \Delta + (1 - \rho) \beta.$$

This is the maximal skew given by Proposition 11. Even though the virtual clocks of non-faulty processes cannot be ahead of $VC_p$, it is possible for $p$ to further advance its local clock at time $u_p$. This may happen, for example, in the following circumstances:

• $f$ processes are faulty and send messages which arrive at $p$ before $t_p + \delta - \varepsilon$.

• A majority of the non-faulty processes start the round at exactly the same time as $p$ (i.e. at time $t_p$) and the messages from these non-faulty processes all arrive at $p$ at time $t_p + \delta - \varepsilon$.

As a result, $cfn(ARR_p) = T + (1 + \rho)(\delta - \varepsilon)$ and $ADJ_p = (1 + \rho)\varepsilon - \rho \delta$. For realistic values of $\rho$ and $\delta$, the correction is positive. Process $p$ advances its local clock and the difference between $VC_p$ and $VC_q$ increases. The value of $v_p$ in this case is $t_p + \delta - \varepsilon$. For the remainder of the interval $[u_p, u_q]$ the two clocks continue to drift apart and it can be shown that at time $u_q$,

$$VC_p(u_q) - VC_q(u_q) = C'_p(u_q) - C_q(u_q)$$

$$= \frac{2\rho}{1 - \rho} \Delta + (1 + \rho)(\beta + \varepsilon) - \rho \delta.$$
3.6 Validity

Assume $p$ is not faulty and $X$ is a clock time such that $X \geq T_0$ and let $t$ be such that $VC_p(t) = X$. The previous sections have shown that the local clocks of other processes at time $t$ are close to $X$. In this section, we examine how close the virtual clocks are from real-time.

The non-faulty processes are assumed to be initially synchronized within a delay $\beta$ of one another. There are then two reals $x_0$ and $y_0$ such that $y_0 \leq x_0 + \beta$ and all the non-faulty processes start the first round within the real-time interval $[x_0, y_0]$. From $x_0$ and $y_0$, we construct four sequences $(x_i)_{i\in\mathbb{N}}$, $(y_i)_{i\in\mathbb{N}}$, $(z_i)_{i\in\mathbb{N}}$ and $(w_i)_{i\in\mathbb{N}}$ as follows.

\[
\begin{align*}
  z_i &= x_i + \delta - \varepsilon \\
  w_i &= y_i + \delta + \varepsilon \\
  x_{i+1} &= z_i + \frac{1}{1 + \rho} (P - \delta) \\
  y_{i+1} &= w_i + \frac{1}{1 - \rho} (P - \delta).
\end{align*}
\]

The first round starts at clock time $T_0$ and for $i \geq 1$, we denote by $T_i = T_0 + iP$, the clock time corresponding to the start of round $i$. In order to simplify the analysis, we assume

\[
\rho \leq \frac{\varepsilon}{\delta + \varepsilon}.
\]

This condition holds in practice and it implies $\rho \leq \varepsilon/(\delta - \varepsilon)$. As a result, the two
following inequalities are satisfied:

\[ 1 - \frac{\varepsilon}{\delta} \leq \frac{1}{1 + \rho} \quad \text{and} \quad \frac{1}{1 - \rho} \leq 1 + \frac{\varepsilon}{\delta}. \]

Under these assumptions, the virtual clocks of all the non-faulty processes are within the area delimited by the two curves \( c_1 \) and \( c_2 \) shown in Fig. 8. The first curve, \( c_1 \), is the union of segments joining the points of coordinates \((x_i, T_i)\) and \((z_i, T_i + \delta)\) and the second curve, \( c_2 \), joins the points of coordinates \((y_i, T_i)\) and \((w_i, T_i + \delta)\).

More precisely, \( c_1 \) is the set of points of coordinates \( (c_1(X), X) \) where \( X \geq T_0 \) and \( c_1 \) is the mapping from clock time to real time defined as follows:

\[
\begin{align*}
  c_1(X) &= x_i + \left(1 - \frac{\varepsilon}{\delta}\right) [X - T_i] & \text{if } T_i \leq X < T_i + \delta \\
  c_1(X) &= z_i + \frac{1}{1 + \rho} [X - (T_i + \delta)] & \text{if } T_i + \delta \leq X < T_{i+1}.
\end{align*}
\]

Similarly, \( c_2 \) is the set of points \( (c_2(X), X) \) where \( X \geq T_0 \) and

\[
\begin{align*}
  c_2(X) &= y_i + \left(1 + \frac{\varepsilon}{\delta}\right) [X - T_i] & \text{if } T_i \leq X < T_i + \delta \\
  c_2(X) &= w_i + \frac{1}{1 - \rho} [X - (T_i + \delta)] & \text{if } T_i + \delta \leq X < T_{i+1}.
\end{align*}
\]

If \( p \) is a non-faulty process and \( VC_p(t) = X \) where \( X \geq T_0 \) then we show that \( t \) is within the interval \([c_1(X), c_2(X)]\).

Let \( T = T_i = T_0 + iP \) for an arbitrary round \( i \). For all non-faulty process \( q \) the clocks \( C_q \) and \( C_q' \) and the reals \( t_q, u_q \) and \( v_q \) are defined as previously. The following lemma shows that the property holds for \( X \) such that \( T \leq X \leq T + P \) provided the non-faulty processes start round \( i \) between \( x_i \) and \( y_i \).

**Lemma 13** Assuming \( x_i \leq t_q \leq y_i \) for all non-faulty process \( q \), then

- if \( T \leq X \leq T + \Delta \) and \( C_p(t) = X \) then \( c_1(X) \leq t \leq c_2(X) \),
- if \( T + \delta \leq X \leq T + P \) and \( C_q'(t) = X \) then \( c_1(X) \leq t \leq c_2(X) \).

**Proof:** For the first part, Lemma 1 yields:

\[
t_p + \frac{1}{1 + \rho} (X - T) \leq t \leq t_p + \frac{1}{1 - \rho} (X - T).
\]

By assumption, \( t_p \) is in the interval \([x_i, y_i]\). It is easy to check that the left hand side is smaller than \( c_1(X) \) and the right hand side larger than \( c_2(X) \).

For any non-faulty process \( q \), the algorithm ensures that \(|t_p - t_q| \leq \beta\). The minimal and maximal elements among the instants \( t_q \) are between \( x_i \) and \( y_i \). Using relation (9), it follows that

\[
x_i + \delta - \varepsilon \leq v_p \leq y_i + \delta + \varepsilon,
\]

that is, \( z_i \leq v_p \leq w_i \). As previously, Lemma 1 gives

\[
v_p + \frac{1}{1 + \rho} [X - (T + \delta)] \leq t \leq v_p + \frac{1}{1 - \rho} [X - (T + \delta)],
\]

for \( X \geq T + \delta \) and this implies \( c_1(X) \leq t \leq c_2(X) \). \( \square \)

Within the round starting at time \( T \), the clock \( VC_p \) is either equal to \( C_p \) or to \( C_p' \). Let \( t \) be between \( t_p \) and \( t'_p \) and let \( X = VC_p(t) \). We have either \( t_p \leq t \leq u_p \)
and $VC_p(t) = C_p(t)$ or $u_p < t \leq t'_p$ and $VC_p(t) = C'_p(t)$. In the first case, $X$ must be between $T$ and $T + \Delta$. In the other case, $C'_p(u_p) < X \leq T + P$. We noted in section 3.4 that $C'_p(u_p) \geq T + \delta$, therefore $T + \delta < X \leq T + P$. The previous lemma implies then that

$$c_1(X) \leq t \leq c_2(X)$$

for all $t$ such that $t_p \leq t \leq t'_p$ and $X = VC_p(t)$. Taking $t = t'_p$ we have $VC_p(t'_p) = T_{i+1}$ and

$$x_{i+1} \leq t'_p \leq y_{i+1}$$

by definition of $c_1$ and $c_2$. As a consequence, the assumption of Lemma 13 is satisfied for round $i + 1$: the non-faulty processes start round $i + 1$ between $x_{i+1}$ and $y_{i+1}$. Since the assumption also holds for the first round, we obtain by induction the following property.

**Proposition 14** For all $X \geq T_0$ and all $t$ such that $VC_p(t) = X$,

$$c_1(X) \leq t \leq c_2(X).$$

The curve $C_1$ can be approximated by the straight line passing through the points of coordinates $(z_i, T_i + \delta)$. These points are circled in Fig. 8. The slope of this line is given by

$$\alpha_1 = \frac{P}{z_{i+1} - z_i} = \frac{(1 + \rho)}{P - \varepsilon + \rho(\delta + \varepsilon)}.$$

Similarly the curve $C_2$ can be approximated by the line which passes through the points of coordinates $(y_i, T_i + \delta)$ and the slope of this line is

$$\alpha_2 = \frac{P}{w_{i+1} - w_i} = \frac{(1 - \rho)}{P + \varepsilon - \rho(\delta + \varepsilon)}.$$

It follows that the clock $VC_p$ of a non-faulty process $p$ is within a linear envelope of real time.

**Proposition 15** For all $t \geq y_0$,

$$T_0 + \alpha_2(t - y_0) \leq VC_p(t) \leq T_0 + \alpha_1(t - x_0).$$

From the assumption $\rho \leq \varepsilon/(\delta + \varepsilon)$, it is easy to see that the coefficients $\alpha_1$ and $\alpha_2$ satisfy the two following inequalities

$$\alpha_2 \leq 1 - \rho \quad \text{and} \quad 1 + \rho \leq \alpha_1.$$

In the worst case, the virtual clocks of non-faulty processes can then drift more from real-time than their physical clocks. The extra drift is caused by the imprecision $\varepsilon$ on communication. During each resynchronization, the virtual clocks can be shifted from real-time by $\pm \varepsilon$ depending on the actual transmission delays experienced during the round.

Unlike the worst case skew, the rate of virtual drift increases as $P$ diminishes. Smaller values of $P$ ensures that the virtual clocks are better synchronized with one another but may result in a faster drift from real-time.
4 Conclusion

The main properties of the Welch-Lynch clock synchronization algorithms are given by Propositions 11 and 12, and by Propositions 14 and 15. For parameters $\beta$, $\Delta$, and $P$ which satisfy the conditions of Fig 4, the algorithm maintains the clocks of two non-faulty processes $p$ and $q$ in approximate agreement and the virtual clocks are limited by two linear functions of real-time as shown in Proposition 15.

In order to minimize the skew, $\Delta$ must be as small as possible. The minimal value of $\Delta$ is $(1 + \rho)(\beta + \delta + \varepsilon)$ and for this value, Proposition 11 and 12 give
\[
|V C_p(t) - V C_q(t)| \leq \frac{(1 + \rho)^2}{1 - \rho} (\beta + \varepsilon) + \frac{\rho(1 + 3\rho)}{1 - \rho} \delta,
\]
for all $t \geq x_0$. As shown at the end of Sect. 3.5, this bound can be effectively attained. Neglecting factors of degree at least two in $\rho$, the above relation gives the worst skew:
\[
\gamma = (1 + 3\rho)(\beta + \varepsilon) + \rho \delta.
\]
This improves slightly over the bound below obtained in the original analysis of the algorithm [2]:
\[
\gamma = (1 + 7\rho)(\beta + \varepsilon) + 3\rho \delta.
\]
This result is based on a slightly different model of clocks than ours but the two are equivalent if factors of degree two or more in $\rho$ are negligible.

To maintain a chosen synchronization level $\beta$, it is reasonable to take $P$ as large as possible. According to relation (17), the maximal value of $P$ is approximately:
\[
P = \frac{\beta}{4\rho} - \frac{\varepsilon}{\rho} - \frac{\beta}{\rho} + \delta.
\]
As previously, this improves a little over the original bound given in [2]:
\[
P = \frac{\beta}{4\rho} - \frac{\varepsilon}{\rho} - 2\beta - \delta - 2\varepsilon.
\]

Despite these slight improvements, the results obtained in the preceding sections are essentially the same as given in [2]. The difference between the two estimates for $\gamma$ is not significant in practice unless $\rho \delta$ is large. Similarly, the difference between the two upper bounds for $P$ is fairly small, except for large values of $\delta$.

The new elements of the proof of correctness are developed in Sect. 3.3. In particular Theorem 9 gives an accurate estimate of the effect of the clock resynchronization procedure. As a result, the bounds on clock skew given in Sect. 3.5 are tight.

References

