4OR (2008) 6:393–401
DOI 10.1007/s10288-007-0060-6

INDUSTRY

A linear programming approach to highly precise clock synchronization over a packet network

Renaud Sirdey · François Maurice

Received: 31 August 2007 / Revised: 27 September 2007 / Published online: 30 October 2007

Abstract In this paper, we propose a linear programming-based method suitable for *precise* and *reliable* estimation of the skew of a slave clock respective to a master clock using timing information carried over an asynchronous packet network. Solving this problem is key to the viability of deploying low-cost IP-based transport technology in existing GSM networks. The paper is concluded by empirical evidence suggesting that the proposed method indeed has the potential to meet the stringent GSM precision requirements.

Keywords Linear programming · Clock synchronization · OR in telecommunications

MSC classification (2000) 90C05 · 90C90 · 68M10

1 Introduction

With its still rapidly expanding two billion user base over more than 200 countries, GSM is to remain the de facto wireless telephony standard for the years to come. However, in the context of such a mass market, there is a strong pressure to optimize both the capital and operational expenditures of GSM networks. Regarding the latter, a key

R. Sirdey (✉)
Service d'architecture BSC, Nortel GSM Access R&D,
Parc d'activités de Magny-Châteaufort, 78928 Yvelines Cedex 09, France
e-mail: renauds@nortel.com

F. Maurice
Service d'architecture BTS, Nortel GSM Access R&D,
Parc d'activités de Magny-Châteaufort, 78928 Yvelines Cedex 09, France
e-mail: maurice@nortel.com

requirement is to adapt GSM traditional high lease-cost Time Division Multiplexing (TDM) transmission to low-cost IP-based transport technology. This is especially critical between the Base Transceiver Stations (BTS), the "antennas", and the Base Station Controllers (BSC), the switches in charge of the first level of traffic concentration in a GSM network[1], as most of the TDM transmission opex are imputable to that subset of the network.

This task is not without challenges.

In particular, one price to pay for GSM spectral efficiency is that a GSM network can work only if the BTS can maintain a highly stable and controlled frequency over the radio interface so as to keep the mobiles synchronized with the network. At the BTS level, the required accuracy is 50 Parts Per Billion[2] (PPB) meaning that when, say, a cesium clock derived frequency ticks one billion times, the clock of the BTS must tick in between one billion ± 50 times (3GPP 2001). Note that as far as GSM is concerned, phase is of little relevance.

To date, common practice has been to install a very high precision reference clock[3] at the BSC and to keep the BTS local oscillator calibrated using the frequency carried over the TDM backhaul signals. In such a setting, a BTS with a relatively low-cost quartz oscillator can hold the 50 PPB requirement virtually indefinitely. Now, however, as the backhaul transitions to IP-based technology, BTS become isolated from their source of synchronization. There are three main ways to address this issue: install an external GPS clock to time the BTS, embed a high precision (e.g., rubidium-based) oscillator inside the BTS or somehow tunnel timing signals from the BSC to the BTS through the IP network. For obvious reasons, the last option is the most cost-effective especially for widely deployed systems such as GSM.

In this paper, we present a *simple* and *robust* method to estimate the skew of the BTS clock relative to the BSC one, when real-time streams of timestamped[4] non-fixed size packets—e.g., the packetized voice calls—flow *both* from the BSC to the BTS (*downlink*) as well as from the BTS to the BSC (*uplink*). Over a relatively short period of time, the BTS collects pairs of sending and receiving dates, respectively with reference to the BSC and BTS clock for the downlink stream and respectively with reference to the BTS and BSC clock for the uplink stream[5], and use these data to obtain an estimate for the local clock skew by solving a linear program in $\mathbb{R}^3$. This estimate is then used to apply a correction to the BTS oscillator.

In the sequel, the BSC and BTS clocks are respectively referred to as the *master* and *slave* clocks.

[1] We refer the reader unfamiliar with the GSM network architecture to the seminal treatise of Mouly and Pautet (1992).

[2] This requirement is relaxed to 100 PPB for a certain class of small BTS, the so called pico-class BTS.

[3] Either corresponding to the core network reference clock (which is either a cesium or a GPS-locked rubidium clock) carried over TDM or based on GPS and G.812 holdover (ITU-T 2004).

[4] In practice, in order to avoid the noise induced by the software layers a packet has to traverse, timestamping is performed at the hardware level just before (respectively after) physically sending (respectively receiving) it. A consequence of this is that the sending date of a given packet cannot be conveyed in the packet itself but rather in a *follow up* packet (IEEE 2007) e.g., the next packet in the real time stream.

[5] The uplink timestamps thus have to somehow find their way down to the BTS.

This paper is organized as follows. Section 2 provides some terminology as well as a brief literature survey on clock skew estimation techniques, Sect. 3 depicts our approach and Sect. 4 provides empirical evidence that this approach allows to meet the stringent GSM precision requirements on simulated data derived from measurements obtained in network settings typical to our application.

2 Preliminaries on clocks and clock skew estimation

2.1 Clock terminology

This section recaps some basic definitions (Moon et al. 1999; Bi et al. 2006).

A *clock* is a piecewise continuous function $C : \mathbb{R} \longrightarrow \mathbb{R}$ that is twice differentiable except on a finite set of points. Let C_A and C_B be two clocks. The *offset* of C_A relative to C_B at time t is defined as $C_A(t) - C_B(t)$. The *frequency*, i.e. the rate at which the clock progresses, of C_A at time t is $C'_A(t)$. The *skew* of C_A relative to C_B at time t is defined as $C'_A(t) - C'_B(t)$. Also, the *clock ratio* of C_A relative to C_B at time t is given by $C'_A(t)/C'_B(t)$. Lastly, the *drift* of C_A relative to C_B at time t is $C''_A(t) - C''_B(t)$.

Over a "short" period of time (typically a few minutes), the drift is generally neglected and, thus, the frequency, skew and clock ratio are assumed to be constant.

Skew and clock ratio are used interchangeably in this paper (in fact we mostly work on the clock ratio) and it is easy to convert from one another as $\delta = C'_A - C'_B = C'_A - C'_A/\alpha = (1 - 1/\alpha)\, C'_A$ where $\alpha = C'_A/C'_B$.

2.2 Literature survey

Perhaps the simplest approach to clock skew estimation consists, using one-way measurements, in plotting the interarrival times of subsequent packets (measured using the slave clock) in ordinate against the associated interdeparture times (measured using the master clock) in abscissa and in performing a linear regression. This approach, however, is known not to work well (Paxson 1998) and is not exempt of assumptions on the well-behavedness of the packet delay probability distribution.

Duda et al. (1987) propose several algorithms motivated by considering another geometric interpretation of the problem: by plotting the slave clock times (receiving dates for the downlink and sending dates for the uplink) in ordinate against the master clock times (sending dates for the downlink and receiving dates for the uplink) in abscissa, one obtains a small empty corridor containing the line which relates the master time to the slave time (see Sect. 3 as well as Fig. 2 for more details). However, the size of the corridor, on which the quality of the estimation relies, depends on the minimum packet delay which is often quite substantial. In essence, the method which we propose in Sect. 3 can be considered a refinement of that approach.

Moon et al. (1999) have been the first to apply linear programming to the problem of clock skew estimation. Their approach consists in fitting a line below (and as close as possible to) the cloud of points obtained by plotting the delay measurements in ordinate against the sending dates in abscissa. The slope of the line then provides the estimated skew. Finding that line requires solving a linear program in $\mathbb{R}^2$. This

approach was further refined by Zhang et al. (2002) who proposed additional objective functions as well as to exploit the convex hull of the cloud of points leading to a method able to reliably deal with master clock resets (an event which cannot happen in the context of our application[6]).

Other approaches include Aweya et al. (2006) (fixed size packets without carrying explicit in-band timing information), Khlifi and Grégroire (2006) (which purposely address low performance systems) as well as Bi et al. (2006) and Wang et al. (2004).

Apart from the method of Duda et al. (1987), all the above methods rely only on one-way measurements. Using two-way measurements, however, provides additional robustness at reasonably low additional cost (uplink measurements can be downed to the slave node in—possibly compressed—periodic batches). Also, to the best of the authors' knowledge, none of the aforementioned techniques have been shown to meet the kind of precision required by the GSM standard. As already mentioned, our approach may be considered a refinement of that of Duda et al. (1987) that takes advantage of the extra robustness of using two-way measurements even under substantial minimum packet delay. Furthermore, the method presented in this paper is both *simple* and *intuitive* as well as amenable to linear time implementation.

3 A linear programming approach

3.1 Principle

Let n and m, respectively, denote the number of downstream and upstream packets. Let $t_i^{(\mathrm{M})}$ and $t_i^{(\mathrm{S})}$ denote the sending and receiving date of the ith downstream packet measured using the master and slave clock, respectively. Furthermore, let $\tau_j^{(\mathrm{S})}$ and $\tau_j^{(\mathrm{M})}$ denote the sending and receiving date of the jth upstream packet measured using the slave and master clock, respectively. Also, let α and β, respectively, denote the skew and offset of the slave clock relative to the master clock, i.e. when the master clock reads t_0 the slave clock reads $\alpha t_0 + \beta$. Lastly, $D_i^{(d)} > 0$ and $D_j^{(u)} > 0$ are random variables which respectively, correspond to the downstream and upstream packet delay.

As illustrated on Fig. 1a, for the ith downstream packet, we have

$$t_i^{(\mathrm{S})} = \alpha t_i^{(\mathrm{M})} + \beta + \alpha D_i^{(d)}. \tag{1}$$

Hence,

$$t_i^{(\mathrm{S})} \geq \alpha t_i^{(\mathrm{M})} + \beta. \tag{2}$$

Also, as illustrated on Fig. 1b, for the jth upstream packet, we have

$$\tau_j^{(\mathrm{S})} = \alpha \tau_j^{(\mathrm{M})} + \beta - \alpha D_j^{(u)}. \tag{3}$$

[6] The BSC reference clock is never reset and in the event of a BSC reset, all the BTS under its responsibility restart.

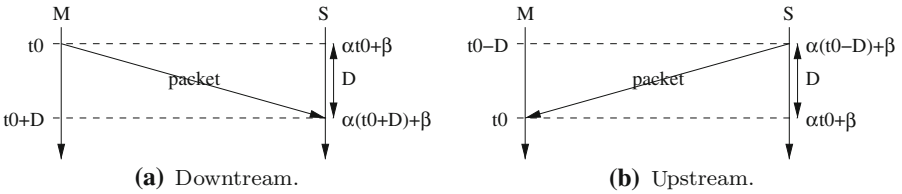


Fig. 1 Illustration of the timestamping principle

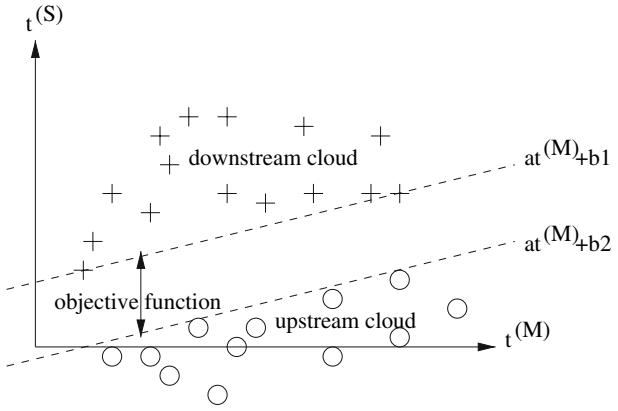


Fig. 2 Illustration of the method principle

Hence,

$$\tau_j^{(S)} \leq \alpha \tau_j^{(M)} + \beta. \tag{4}$$

Thus, the line $t^{(S)} = \alpha t^{(M)} + \beta$ is stuck between the downstream cloud of points obtained by plotting $t_i^{(S)}$ in ordinate against $t_i^{(M)}$ in abscissa (which lies above) and the upstream cloud of points obtained by plotting $\tau_i^{(S)}$ in ordinate against $\tau_i^{(M)}$ in abscissa (which lies below). However, an approach which would estimate a single line, either by some kind of linear regression or by arbitrarily separating the two clouds of points, as suggested by Duda et al. (1987), can be expected not to perform very well in case of a non zero minimum delay, a situation which, in practice, is more the rule than the exception. Thus, rather than estimating a single line, our approach consists in fitting an as wide as possible "corridor" to the data, i.e. two parallel lines lying in between the downstream and upstream clouds of points but as far as apart as possible (see Fig. 2).

3.2 Linear programming formulation

Clearly, inequalities (2) and (4) can be interpreted as the constraints of a linear program.

Let $t^{(S)} = \alpha t^{(M)} + \beta_1$ and $t^{(S)} = \alpha t^{(M)} + \beta_2$, respectively, denote the upper and lower frontiers of the corridor. Then an estimate for α is obtained by solving the

following linear program:

$$\begin{cases} \text{Maximize } \beta_1 - \beta_2 \\ \text{s. t.} \\ \alpha t_i^{(\mathrm{M})} + \beta_1 \leq t_i^{(\mathrm{S})}, \quad i \in \{1, \ldots, n\}, \\ \alpha \tau_j^{(\mathrm{M})} + \beta_2 \geq \tau_j^{(\mathrm{S})}, \quad j \in \{1, \ldots, m\}. \end{cases}$$

Fortunately enough, only three variables are involved and it is well known that linear programming in both $\mathbb{R}^2$ and $\mathbb{R}^3$ can be dealt with in time linear in the number of constraints (Megiddo 1983; Dyers 1983), here $n+m$. Needless to say, of course, that a more readily available implementation of the simplex algorithm can also potentially be used, when appropriate.

Also, note that $(\beta_1 + \beta_2)/2$ provides a rough estimate of the slave clock offset relative to the master clock. However, the validity of this estimate depends on the degree of symmetry which the network behavior exhibits.

3.3 System-level view

From a system perspective, the overall method requires the slave node to

1. Collect the downlink and uplink timing data.
2. Solve the linear program of the previous section.
3. Use the estimated skew to apply a *bounded* correction (e.g., limited to 10 PPB per period) to its local oscillator.

This needs to be done continuously with a data collection period short enough so that the constant skew assumption remains reasonable (typically a few minutes).

Also, as clock drift is in essence a slow phenomenon, it would be good engineering practice to dismiss a skew estimation as an outlier when it differs too significantly from the previous one. Indeed, although the method can be expected to exhibit a certain degree of robustness to network events such as reroutings, since two-way measurements are exploited, the fact that certain transient network conditions may from time to time lead to aberrant results cannot be ruled out.

Additionally, it should be emphasized that the resolution of the linear program of Sect. 3.2 is not real-time critical (typically a budget of a few seconds can be allocated to that task). Hence, the choice between a general purpose off-the-shelf implementation of the simplex algorithm and a custom implementation of a specialized linear-time algorithm limited to solving linear programs in $\mathbb{R}^3$ is driven by software engineering considerations such as the amount of software it is reasonable to embed in the system (general purpose solvers tend to be huge), development effort (off-the-shelf versus custom), trustworthiness (external versus internal), etc.

4 Computational experiments

This section provides empirical evidence that our LP approach indeed allows to meet the GSM precision requirements in two realistic and typical network settings: a BTS

connected to its BSC over a private Wide Area Network (WAN) and a pico-class BTS connected to its BSC over the public Internet and an ADSL connection.

4.1 Private WAN setting

Initially, in order to get an idea of the kind of conditions under which the method would operate, we started by performing around 10,000 pings of a workstation residing near London from another computer residing near Paris, both computers being part of the same private WAN. See Fig. 3a. During this experiment, the average Round Trip Delay (RTD) was 27.70 ms with a standard deviation of 11.19 ms as well as a minimum and maximum RTD of 26 and 196 ms, respectively. Approximately 0.01% of the packets were lost.

Since only single trip delays were of interest to us, we assumed that the network behavior was symmetric during the above experiment and divided the ping data by 2. Then, in order to perform realistic simulations, a Weibull distribution[7] was fitted to these data leading to position, shape and scale parameters respectively equal to 13 ms, 0.30 and 0.11.

Thus, for the private WAN setting, our experiment consisted in simulating the sending of one packet every 5 ms in both downlink and uplink and to draw both $D_i^{(d)}$ and $D_i^{(u)}$ (Eqs. (1), (3), respectively) from the aforementioned Weibull distribution. We then tried to estimate a skew of +20 PPB that is, $\alpha = 1.000000020$.

Table 1 summarizes our results. The "Duration" column indicates the duration of the data collection period, the "# packets" column provides the number of pairs of sending/receiving dates which were collected and the "Mean", "St. dev.", "Min" and "Max" column give basic statistics (in PPB units) for the absolute error, i.e. $|\hat{\alpha} - \alpha|$, where $\hat{\alpha}$ denotes the estimation obtained by solving the linear programs of Sect. 3.2. The linear program were solved using COIN-OR[8] simplex solver which, performance-wise, is able to handle programs with around 250 000 constraints in less than a second.

The results in Table 1 do indeed suggest that, in a private WAN setting, the method has the potential to estimate the skew with sub-PPB accuracy, with an observation period of reasonable duration. These results are further discussed in Sect. 4.3.

4.2 Public Internet setting

As for the private WAN setting, we started by performing around 10,000 pings of a workstation residing in a private WAN from another computer having access to the

[7] Recall that the Weibull distribution is given by

$$f(x;\tau,a,b) = ab^{-a}(x-\tau)^{a-1}e^{-\left(\frac{x-\tau}{b}\right)^a},$$

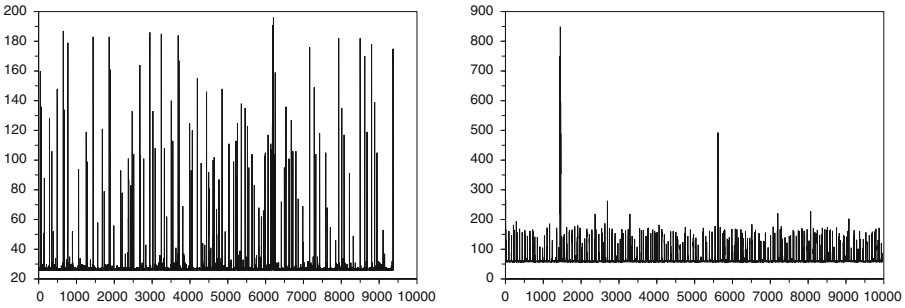
where τ, a and b, respectively, denote the position, shape and scale parameters (Saporta 1990). The Weibull distribution is commonly used to model the packet delay in connectionless networks (Norros 1995; Papagiannaki et al. 2003).

[8] Computational Infrastructure for Operations Research (www.coin-or.org).

Table 1 Statistics of $|\hat{\alpha} - \alpha|$ in function of the duration, for the private WAN setting

Duration	# Packets	Mean	St. dev.	Min	Max
10 s	2,000	0.06743	0.13902	0.00001	0.74213
1 min	12,000	0.02789	0.04952	2.7×10^{-6}	0.25863
10 min	120,000	0.00451	0.01065	2×10^{-7}	0.08190

The figures are in PPB units and were obtained over 100 simulations



(a) Private WAN. **(b)** Public Internet.

Fig. 3 Ping data (time in seconds in abscissa, RTD in milliseconds in ordinate)

Table 2 Statistics of $|\hat{\alpha} - \alpha|$ in function of the duration, for the public Internet setting

Duration	# Packets	Mean	St. dev.	Min	Max
10 s	500	0.75066	1.13316	0.00029	6.72219
1 min	3,000	0.04291	0.05569	0.00002	0.27037
10 mins	30,000	0.01065	0.01640	3.8×10^{-6}	0.08255

The figures are in PPB units and were obtained over 100 simulations

public Internet via an ADSL connection. See Fig. 3b. During this experiment, the average Round Trip Delay was 64.06 ms with a standard deviation of 28.04 ms as well as a minimum and maximum RTD of 55 and 849 ms, respectively. Approximately 0.33% of the packets were lost.

Again, as in the previous section, the ping data were divided by 2 and a Weibull distribution was fitted to them leading to position, shape and scale parameters respectively equal to 27.5 ms, 0.40 and 1.35.

Thus, for the public Internet setting, our experiment consisted in simulating the sending of one packet every 20 ms in both downlink and uplink and to draw both $D_i^{(d)}$ and $D_i^{(u)}$ (Eqs. (1) and (3), respectively) from the aforementioned Weibull distribution. We then tried to estimate a skew of $+40$ PPB that is, $\alpha = 1.000000040$. Recall that the ± 50 PPB requirement is relaxed to ± 100 PPB for the pico-class BTS (3GPP 2001).

Table 2 summarizes our results (please refer to the previous section for a detailed description of each of the columns).

4.3 Discussion

The results in the previous two sections suggest that our approach does indeed provide a satisfactory level of precision as long as the timing data collection period is greater than 1 min (in summary, a worst case error of approximately 0.3 PPB was observed for both the private WAN—for a 20 PPB skew—and public Internet settings—for a 40 PBB skew). Thus, a data collection period in between 1 and 10 min appears reasonable both in terms of precision as well as in term of reasonableness of the constant skew assumption.

Acknowledgments The authors wish to thank Gil Botet and Jean-Louis Meneghetti for several suggestions that led to improvements in the paper.

References

3GPP (2001) Digital cellular telecommunications system (Phase 2+)—Radio subsystem synchronization. Technical Report ETSI TS 145 010 V4.0.0 (2001–2004), European Telecommunications Standards Institute

Aweya J, Montuno DY, Ouellette M, Felske K (2006) Clock recovery based on packet inter-arrival time averaging. Comput Commun 29:1696–1709

Bi J, Wu Q, Li Z (2006) On estimating clock skew for one-way measurements. Comput Commun 29:1213–1225

Duda A, Harrus G, Haddad Y, Bernard G (1987) Estimating global time in dsitributed systems. In: Proceedings of the 7th IEEE international conference on distributed computing systems, pp 299–306

Dyers ME (1983) Linear time algorithms for two- and three-variable linear programs. SIAM J Comput 13:31–45

IEEE (2007) Draft standard for a precision clock synchronization protocol for networked measurement and control systems. Technical Report IEEE P1588/D1-I 2007-04-15, Institute of Electrical and Electronics Engineer

ITU-T (2004) Timing requirements of slave clocks suitable for use as node clocks in synchronization networks. Technical Report ITU-T Recommendation G.812 (06/2004), International Telecommunication Union

Khlifi H, Grégroire J-C (2006) Low complexity offline and online clock skew estimation and removal. Comput Netw 50:1872–1884

Megiddo N (1983) Linear-time algorithms for linear programs in $\mathbb{R}^3$ and related problems. SIAM J Comput 12:759–776

Moon SB, Skelly P, Towsley D (1999) Estimation and removal of clock skew from network delay measurements. In: Proceedings of IEEE INFOCOM, pp 227–234

Mouly M, Pautet M-B (1992) The GSM system for mobile commnunications—a comprehensive overview of the European Digital Cellular Systems. Telecom Publishing

Norros I (1995) On the use of fractional brownian motion in the theory of connectionless networks. IEEE J Sel Areas Commun 13:953–962

Papagiannaki K, Moon S, Fraleigh C, Thiran P, Diot C (2003) Measurement and analysis of single-hop delay on an IP backbone network. IEEE J Sel Areas Commun 21:908–921

Paxson V (1998) On calibrating measurements of packet transit times. In: Joint international conference on measurement and modeling of computer systems, pp 11–21

Saporta G (1990) Probabilités, analyse des données et statistiques. Éditions Technip

Wang J, Zhou M, Zhou H (2004) Clock synchronization for internet measurements: a clustering algorithm. Comput Netw 45:731–741

Zhang L, Liu Z, Xia CH (2002) Clock synchronization algorithms for network measurements. In: Proceedings of IEEE INFOCOM, pp 160–169