Network Time Synchronization Using Decentralized Kalman Filtering

Maxime Cohen

Network Time Synchronization Using Decentralized Kalman Filtering

Research Thesis

Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science

Maxime Cohen

Submitted to the Senate of the
Technion - Israel Institute Technology

Cheshvan 5769 Haifa October 2009

This research thesis was done under the supervision of
Prof. Nahum Shimkin in the department of Electrical Engineering.

Acknowledgement

I would like to thank Prof. Nahum Shimkin for his guidance, for his advice and support, and for the great working atmosphere during all the stages of the research. I would also like to thank the reviewers for their precious and helpful insights on this thesis. Special gratitude to my parents for their love and for the education they gave me, with its enormous emphasis on academic excellence.

The generous financial help of the Technion is gratefully acknowledged

Contents

List of Figures

List of Tables

Abstract

Accurate clock synchronization is important in many distributed applications, both in wire line and wireless computer networks. Time synchronization between the nodes of a network was extensively treated in the literature, where several methods and algorithms were proposed to solve this problem efficiently. In the Internet for example, the "Network Time Protocol" (NTP) is the most widely accepted standard for clock synchronization.

In some recent work, improved algorithms that rely on Least-Squares estimation were introduced. The accuracy of clock synchronization was improved by imposing the global constraints for all the loops in the multihop network and the use of a distributed algorithm employing only local broadcasts. A central characteristic of these methods is their decentralized structure that requires only local communication with neighbors. In this research, we will extend the Least-Squares framework by developing algorithms that estimate the offset of the local clock at each network node, using a Kalman Filter framework. We will present a synchronous decentralized implementation of the filtering algorithm that employs only local broadcasts and we will prove that it converges to the optimal centralized solution. The Kalman Filter framework allows exploiting some a-priori knowledge and providing different weights to the measurements according to their accuracy. The next step is to consider the multiple measurement case and to present a recursive version of these algorithms. The recursive algorithm computes the optimal offsets and the corresponding variances after receiving each set of measurements in a decentralized manner. Finally, we will extend the results to the estimation of the clock skew (i.e., rate deviation) in addition to its offset. Then, we will consider different extensions of the basic algorithm. We will incorporate a discount factor in the objective function and treat the case where temporary communication failures are considered.

We also present simulation results over several network topologies for evaluating and comparing the accuracy of the proposed time synchronization schemes. We will provide several interesting comparisons and as expected, the Kalman Filter approach outperforms the existing algorithms.

Summary of Notations

A	Reduced incidence matrix
α_i	Skew (rate deviation) of node Λ_i
$\hat{\alpha}_{ji}$	Estimated skew ratio between nodes Λ_i and Λ_j
$\underline{b}$	Bias (constant random noise)
B	Bias covariance matrix
$Cov[\cdot,\cdot]$	Covariance function
γ	Forgetting factor
d_i	Deviation from the i-th data point
δ_{kl}	Kronecker's delta
Π	Objective function in the best fitting curve problem
e_{ij}	Bidirectional link between nodes Λ_i and Λ_j
$E[\cdot]$	Mathematical expectation
ε_{ij}	Additive noise in the measurement model between nodes Λ_i and Λ_j
J	Objective function
k	Iteration number
k_m	Probe packet number
L^T	Matrix transpose
m	Number of edges in the network
M	Iteration matrix
n	Discrete time index
N	Number of nodes in the network
$\|N_i\|$	Number of elements in the set N_i
$N(\mu,\Sigma)$	Gaussian density with mean μ and covariance matrix Σ
$\nabla_{\underline{x}}$	Gradient operator with respect to $\underline{x}$
$\hat{O}_{ij}$	Measurement between the pair Λ_i and Λ_j
$P\{\cdot\}$	Probability
$P\{\cdot \mid \cdot\}$	Conditional probability
$P(n \mid n)$	Error covariance matrix at time n given observations up to and including time n
$P(n+1 \mid n)$	Error covariance matrix at time $n+1$ given observations up to and including time n
$\rho(M)$	Spectral radius of M
P_0	Initial known covariance matrix
$(P_0^{-1})_{i*}$	Row number i of the matrix P_0^{-1}
r_{ji}	Variance of the measurement $\hat{O}_{ij}$
R^{-1}	Inverse covariance matrix of the measurement noise
Λ_i	Node number

t	Real time (reference time)
T_b	Number of measurement sets in the skew estimation problem
$T_i(t)$	Local time (at node Λ_i)
T_S	Sampling interval
t_m	Transmission time of packet k_m
$\tilde{t}_m$	Received time of packet k_m
τ_i	Offset of node Λ_i
τ^*	Optimal centralized solution
$\underline{u}(n)$	External input at time n in the state space model
$\underline{v}(n+1)$	Measurement noise at time $n+1$ in the state space model
$\underline{w}(n)$	Process noise at time n in the state space model
W_{ii}	Weight number i in the WLS problem
$\overline{x}_0$	Initial known state vector
$\underline{x}(n)$	Sate vector at time n
$\underline{\hat{x}}(n\|n)$	State estimate at time n given observations up to and including time n
$\underline{\hat{x}}(n+1\|n)$	State estimate at time $n+1$ given observations up to and including time n
$x_{ij}(k_m)$	Propagation delay of packet k_m between nodes Λ_i and Λ_j
$X(n)$	Augmented state space vector at time n
y	Measurement vector
$\underline{z}(n+1)$	Measurement vector at time $n+1$ in the state space model

Summary of Abbreviations

BLUE	Best Linear Unbiased Estimator
CTP	Classless time Protocol
CKF	Centralized Kalman Filter
CLS	Centralized Least-Squares
DKF	Decentralized Kalman Filter
FTSP	Flooding Time Synchronization Protocol
GM	Gauss–Markov
GPS	Global Positioning System
IID	Independent Identically Distributed
KF	Kalman Filter
LQG	Linear-Quadratic-Gaussian
LQR	Linear-Quadratic Regulator
LS	Least-Squares
MAP	Maximum-A-Posteriori
ML	Maximum-Likelihood
MMSE	Minimum Mean Squared-Error
NTP	Network Time Protocol
OLS	Ordinary Least-Squares
PDF	Probability Density Function
PSD	Positive Semi-Definite
RBS	Reference-Broadcast Synchronization
RLS	Recursive Least-Squares
SNTP	Simple Network Time Protocol
SOA	Sub-Optimal Algorithm
s.t	Such That
UTC	Coordinated Universal Time
WSN	Wireless Sensor Network

1. Introduction

Accurate clock synchronization is required in many distributed applications in computer networks (e.g., sleep scheduling in the case of low duty cycle [24], and tracking in wireless sensor networks [33]). Moreover, network time synchronization is a critical component for commercial organizations that rely on several computers, all of which have clocks that are the source of time for the files or operations they handle. When clocks of the different components on such systems are not synchronized, data can be lost, processes can fail, the exposure increases and security is compromised. The task of synchronizing clocks in distributed systems is usually accomplished via the exchange of standard messages (probe packets) between the distributed entities in order to coordinate their time. We will assume for simplicity that the links are bi-directional, the network topology is time-invariant and that each node is capable of sending and receiving messages from its neighbors. There is a large literature on how to synchronize clocks in traditional networked systems; among these, the "Network Time Protocol" (NTP) is the most widely accepted standard for synchronizing clocks over the Internet [28-30].

More recently, a novel approach for time synchronization termed CTP – Classless Time Protocol [14] was proposed. This non-hierarchical approach exploits convex optimization theory in order to evaluate the impact of each clock offset on the overall objective function. It was shown that CTP substantially outperforms hierarchical schemes such as NTP in the sense of clock accuracy with respect to a universal clock, without increasing complexity. An alternative proposed approach is the well known Least-Squares Estimator in [41, 12]. The accuracy of clock synchronization was improved by exploiting global network-wide constraints (e.g., the relative offsets are summing up to zero over loops) and the use of a completely asynchronous, distributed algorithm employing only local broadcasts. The central characteristic of these methods relies in their decentralized structure that requires only local communication with neighbors.

In estimation theory, for a linear dynamic system under the Gaussian assumption the Kalman Filter (KF) is the optimal MMSE (Minimum Mean Squared-Error) state estimator. If the Gaussian assumption is relaxed, we will obtain the linear optimal MMSE state estimator. The implementation of the KF in a decentralized manner was extensively treated in the literature, as we will see in Section 3. Our objective is to develop efficient decentralized estimation algorithms in order to synchronize the different clocks over the network with respect to the reference time. Without loss of generality, we can assume that Node 1 is synchronized with the universal clock, and we thus have to synchronize the other clocks with respect to it. Firstly, we will consider the case where all the clocks run exactly at the same rate (i.e., there is no clock skew). In this case, our objective reduces to estimate the clock offsets at each network node relative to the clock reference.

We will extend the Least-Squares framework by developing algorithms that estimate the offset of the local clock at each network node, using a Kalman Filter framework. The first step is to formulate the model in the state space form where the state is the vector of biases of the clocks in the network. Then, we will show that a single measurement vector update can be done using a distributed iterative scheme that converges to the optimal centralized estimator. The Kalman Filter framework allows exploiting a-priori knowledge about the estimated quantity and providing different weights to the measurements according to their accuracy and quality. We will make the natural assumption that the initial state covariance matrix is diagonal, however we will observe that after the first measurement update of the KF, the state covariance matrix does not remain diagonal. Hence, from this step the standard KF equations cannot be decentralized and each node has to communicate with every other node in the network. This is not a desirable situation since it is prohibitively expensive in terms of communication time. We will solve this issue by proposing a decentralized recursive algorithm that relies on manipulating the standard equations. We rely on the theorem that claims the equivalence between the KF solution and the minimizing vector of a deterministic constrained LS problem. In this way, we will be able to obtain the existing LS solution as a special case. We find the optimal solution by a coordinate differentiation and then we will implement the optimal equation by a synchronous iterative algorithm that employs only local broadcasts. Then, we will prove that it converges to the optimal centralized solution. The next step is to consider the multiple measurement case and to present a recursive version of these algorithms. The recursive algorithm computes the optimal offsets and the corresponding variances in a decentralized manner after receiving each set of measurements. We also consider a simple sub-optimal algorithm that neglects the off-diagonal terms of the inverse covariance matrix. This method reduces significantly the complexity, but looses its optimal property. We will see in the simulation results section that this algorithm leads to poor results.

In the extensions section we will incorporate a discount factor in the objective quadratic function to compensate for the time-invariant offsets assumption. Then, we modify our algorithm slightly to make it robust to temporary communication failures. We briefly consider the extension of our results to the estimation of both the offsets and the clock skew (i.e., rate deviation). We will show that the clock skew estimation problem reduces to the same mathematical setup as the offset estimation problem under the appropriate substitutions. For the clock skew estimation problem, we propose different approaches. In the first, the clock skew estimation is performed separately from that of the clock offset. In the second, we propose an optimal combined estimation of both the clock offset and the clock skew.

Finally, we present simulation results over several network topologies for evaluating and comparing the accuracy of the proposed time synchronization schemes. We provide several interesting comparisons, where the Kalman Filter approach outperforms the existing algorithms.

It is interesting to note that the time synchronization problem is mathematically equivalent to any related distributed estimation problem stemming from relative additive measurements in sensor networks [3]. For example, one can apply the same algorithms to the sensor localization problem. We will briefly elaborate on this point in Section 7.

This thesis is organized as follows. In sections 2 and 3, we review the required scientific background and the related work respectively. In Section 4, we describe the model and formulate the problem. Then, in Section 5, we present the different algorithms for the case of single measurement update (both centralized and decentralized versions). Section 6 is devoted to show the convergence of the most general decentralized algorithm to the optimal centralized solution. In Section 7, we provide the recursive version of our algorithm for multiple measurements and an additional non-recursive algorithm. Sections 8 and 9 treat several extensions, like the incorporation of a discount factor and the estimation of the clock skew. Numerical results are presented in Section 10. Finally, the conclusions and some notes on future directions are reported in Section 11.

2. Scientific Background

2.1 Least-Squares Fit

Let us first consider the Least-Squares (LS) method in a deterministic context and then explain its statistic interpretation. The method of Least-Squares or Ordinary Least-Squares (OLS) is used to solve over-determined systems and can be interpreted as a method of fitting data. This algorithm is often applied in statistical contexts, particularly in regression analysis. The best Least-Squares fit is that instance of the model for which the sum of squared residuals has its lowest value, a residual being the difference between an observed value and the value given by the model. In other words, the method of Least-Squares assumes that the best-fit curve of a given type is the curve that has the minimal sum of deviations squared (least square error) for a given set of data. For example, we can fit the data to a polynomial function, as presented in the following figure:

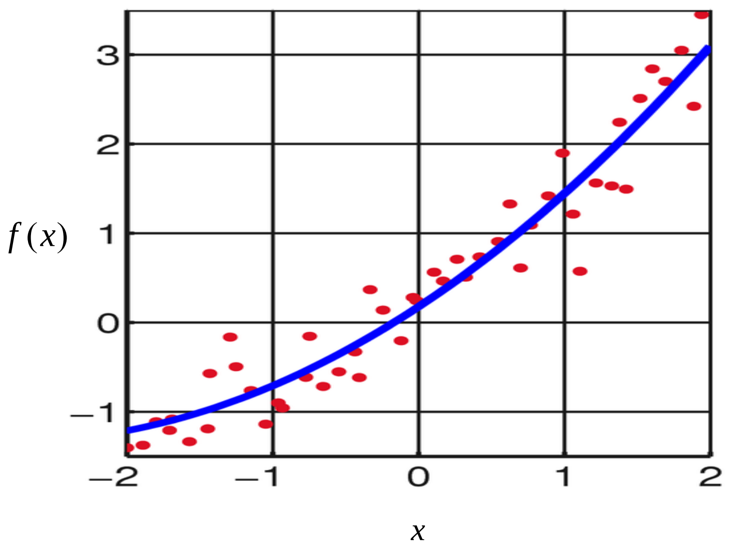


Figure 2.1. Fitting a set of data points using a quadratic function.

Suppose that the data points are:

$$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$$

Here y_i are the measured values (data) and x_i are the independent variables (unknown). The fitting curve $f(x)$ has the deviation d_i from each data point:

$$d_i = y_i - f(x_i) \quad i = 1, 2, ..n.$$

According to the LS method, the best fitting curve has the property that the following expression is minimal:

$$\Pi = \sum_{i=1}^{n} (d_i)^2 = \|\underline{d}\|^2 = \sum_{i=1}^{n} (y_i - f(x_i))^2 \rightarrow \min \qquad (2.1.1)$$

The above minimum in (2.1.1) can be found by setting the gradient to zero. Since the model contains n parameters, we will obtain n gradient equations.

As a special case, the linear LS problem with the following over-determined system (M linear equations in N unknown variables, with M>N) is considered:

$$\sum_{j=1}^{N} a_{ij} x_j = y_i \qquad i = 1,2,..,M$$

In a matrix notation:

$$Ax = y$$

In order to find the optimal LS solution, we have to minimize the following quadratic objective function:

$$\Pi = \|y - Ax\|^2 \to \min$$

A unique optimal solution is obtained (when $A^T A > 0$) by solving the normal equations:

$$\left(A^T A\right) x = A^T y$$

The above equation can be obtained by differentiating the objective function with respect to the vector x and setting the result to zero.

Now, we will consider several approaches to iteratively solve the normal equations. Let us define:

$$\begin{cases} M = I - A^T A \\ \overline{y} = A^T y \end{cases}$$

We note that the matrices M and $I - M$ are the projection matrices.
Using these notations in the normal equations, we will obtain:

$$\left(I - M\right) x = \overline{y}$$

This implies:

$$x = Mx + \overline{y}$$

We can implement the above equation through the use of an iterative (synchronous or asynchronous) algorithm and the convergence depends on the structural properties of the matrix M. For example, the synchronous algorithm (all the entries are updated simultaneously) is given by:

$$x^{(k+1)} = Mx^{(k)} + \overline{y} \qquad (2.1.2)$$

Here, $k \geq 0$ is the iteration number. The initial conditions can be randomly chosen and does not affect the convergence of the algorithm in (2.1.2).

The above method is known as the Jacobi algorithm and is very common in linear algebra. In this thesis, we will employ this method in a synchronous way to solve the linear Least-Squares problem.

We also mention the relaxed form of the Jacobi algorithm. This is an alternative approach that leads to similar equations based on the gradient algorithm. Given the similar linear equations:

$$Ax = y$$

The gradient descent method is given by:

$$x^{(k+1)} = x^{(k)} - \eta \cdot \nabla_x \Pi = x^{(k)} - \eta \cdot \left[\left(I - M \right) x^{(k)} - \bar{y} \right]$$
$$x^{(k+1)} = \left[I - \eta \left(I - M \right) \right] x^{(k)} + \eta \bar{y}$$
$$x^{(k+1)} = \left(1 - \eta \right) \cdot I \, x^{(k)} + \eta \left(M x^{(k)} + \bar{y} \right)$$

Here, η is the step size of the algorithm. One can note that if $\eta = 1$, the gradient algorithm reduces to the Jacobi method. Hence, the gradient algorithm is more general and can be viewed as a relaxed Jacobi method. In the case where the Jacobi algorithm does not converge, we can try to use the gradient algorithm and reduce the step size to achieve convergence.

Two interesting extensions to the basic LS case are considered: the Weighted Least-Squares (WLS) and the Recursive Least-Squares (RLS). In the WLS method, each data is multiplied by a weighting factor. In other words, the objective function to be minimized is a weighted sum of the form:

$$\Pi = \sum_{i=1}^{n} W_{ii} \left(d_i \right)^2 \to \min$$

The RLS method is the recursive version of the basic LS algorithm where data arrives progressively. In this particular case, the minimization process is repeated for each set of measurements. Moreover, the most useful form is RLS with exponential data weighting (incorporation of a forgetting factor). In the latter, we consider the scenario where the most recent data is assumed to be more informative than past data and hence we exponentially discard old data.

The Least-Squares method also has a statistical interpretation in estimation theory. In a linear model in which the errors have a zero expectation conditional on the independent variables, are uncorrelated and have equal variances (IID), the Best Linear Unbiased Estimator (BLUE) of any linear combination of the observations is its Least-Squares estimator. This result is known as the Gauss-Markov (GM) theorem. "Best" means that the Least-Squares parameter estimators have minimum variance. The assumption of equal variance is valid when the errors all belong to the same distribution. Moreover, in a linear model, if the errors belong to a Normal distribution, the Least-Squares estimators are also the Maximum-Likelihood estimators (as we will show in Appendix B).

Aitken [1] showed that when a weighted sum of squared residuals is minimized, the solution is the BLUE if each weight is equal to the reciprocal of the variance-covariance matrix of the observations. This method is known as the Weighted-Least-Squares (WLS) method. In the linear non-deterministic case, there exists a closed form solution to the RLS algorithm that can be implemented through an iterative procedure. For more details on RLS, see [13, 40].

It can be found in Appendix A that the Kalman-Filter algorithm can be viewed as a deterministic LS optimization problem. Next, we present the basic background on Kalman Filtering.

2.2 Kalman Filtering

The Kalman Filter (KF) is an efficient recursive filter that estimates the state of a linear dynamic system from a series of noisy measurements. It is used in a wide range of engineering applications from radar to computer vision, and is an important topic in control theory and control systems engineering. Together with the Linear-Quadratic Regulator (LQR), the Kalman Filter solves the Linear-Quadratic-Gaussian control problem (LQG) [27, 13]. As seen in Figure 2.2, the KF is fed measurements from the system of interest and produces an estimate of the system state. The system is modeled either as a set of differential equations in the continuous-time case or as a set of difference equations in the case of a discrete-time system. The system model is used to propagate the estimate of the system state forward in time until a new measurement is received. At this point, the system state attained from the measurement is compared to the estimate of the system state and combined in an optimal (MMSE) manner.

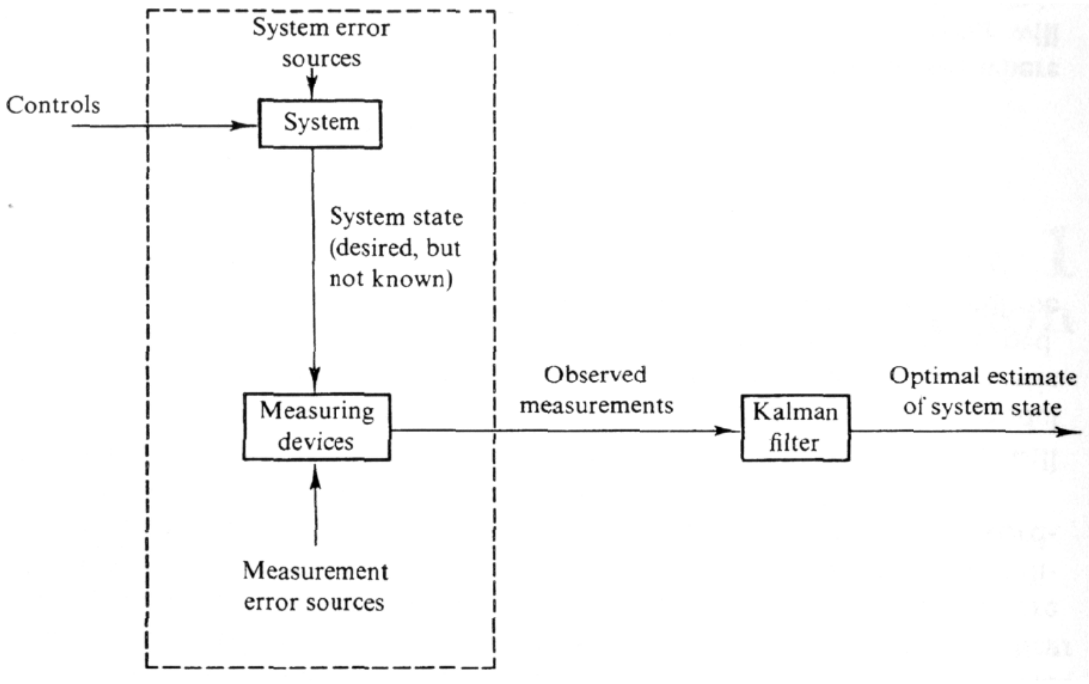

System error sources
Controls
System
System state (desired, but not known)
Observed measurements
Optimal estimate of system state
Measuring devices
Kalman filter
Measurement error sources

Figure 2.2. Typical Kalman Filter application, from Maybeck [27].

In order to use the Kalman Filter to estimate the internal state of a process given only a sequence of noisy observations, one must model the process in accordance with the framework of the KF, i.e., in a state space model notation. This means specifying the matrices Φ, Γ, H for each time-step n as described below. In other words, the KF model assumes the true state at time $n+1$ is evolved from the state at n according to:

$$\begin{cases} \underline{x}(n+1) = \Phi(n)\underline{x}(n) + \Gamma(n)\underline{w}(n) \\ \underline{y}(n) = H(n)\underline{x}(n) + \underline{v}(n) \end{cases}$$

The Kalman Filter is a recursive estimator. This means that only the estimated state from the previous time step and the current measurement are needed to compute the estimate of the current state. In contrast to batch estimation techniques, no history of observations and/or estimates is required. The state of the filter is represented by two variables:

- $\underline{\hat{x}}(n|n)$, the state estimate at time n given observations up to and including time n.
- $P(n|n)$, the error covariance matrix (a measure of the estimated accuracy) at time n given observations up to and including time n.

The Kalman Filter has two distinct phases: prediction and update. The prediction phase uses the state estimate from the previous step to produce an estimate of the state at the current step. In the update phase, measurement information at the current time is used to refine this prediction to arrive at a new, (hopefully) more accurate state estimate.

We will next present the equations of the Kalman Filter algorithm. We consider both the standard form and the information form.

Consider the following state space model:

$$\begin{cases} \underline{x}(n+1) = \Phi(n)\underline{x}(n) + \Gamma(n)\underline{w}(n) \\ \underline{y}(n) = H(n)\underline{x}(n) + \underline{v}(n) \end{cases}$$

$\underline{x}(0)$ is the initial state of the system with the following first and second order statistics:

$$E\left[\underline{x}(0)\right] = \underline{m}_x(0) \quad \operatorname{cov}\left[\underline{x}(0)\right] = E\left[\left(\underline{x}(0)-\underline{m}_x(0)\right)\left(\underline{x}(0)-\underline{m}_x(0)\right)^T\right] = P_x(0) = P_0 .$$

$\{\underline{w}(n)\}$ is the process noise modeled as a white Gaussian noise with zero mean and covariance $Q(n) \geq 0$.
$\{\underline{v}(n)\}$ is the measurement noise modeled as a white Gaussian noise with zero mean and covariance $R(n) > 0$.
$\{\underline{w}(n)\}, \{\underline{v}(n)\}, \underline{x}(0)$ are uncorrelated, namely:

$$E\left[\underline{v}(n)\underline{w}^T(n)\right] = E\left[\underline{x}(0)\underline{v}^T(m)\right] = E\left[\underline{x}(0)\underline{w}^T(m)\right] = 0 \ \ \forall m,n$$

The state estimation cycle is divided into two steps (n is the discrete time index):

- Time update (prediction):

$$\begin{cases} \underline{\hat{x}}(n+1|n) = \Phi(n)\underline{\hat{x}}(n|n) \\ P(n+1|n) = \Phi(n)P(n|n)\Phi^T(n) + \Gamma(n)Q(n)\Gamma^T(n) \end{cases}$$

- Measurement update:

$$\begin{cases} \underline{\hat{x}}(n+1|n+1) = \underline{\hat{x}}(n+1|n) + K(n+1)\left[\underline{y}(n+1) - H(n+1)\underline{\hat{x}}(n+1|n)\right] \\ K(n+1) = P(n+1|k)H^T(n+1)\left[H(n+1)P(n+1|k)H^T(n+1) + R(n+1)\right]^{-1} \\ P(n+1|n+1) = \left[I - K(n+1)H(n+1)\right]P(n+1|n) \end{cases}$$

The initialization is as follows:

$$\underline{\hat{x}}(0|0) = \underline{m}_x(0) \ ; \ P(0|0) = P_x(0)$$

As we will see later, in our case, we have:

$$\Gamma(n) = \Phi(n) = I$$
$$Q(n) = Q \; ; \; R(n) = R$$

Next, we will review an additional form of the Kalman Filter, called the information form.

2.3 Kalman Filter - Information Form

The information form of the Kalman Filter differs in the fact that the covariance prediction and update equations are different. Since the prediction covariance equation is quite complex, another option is to use the regular form of the KF with the modification in the update covariance equation only. Under the same assumptions as those stated previously, the Kalman Filter equations are given by the same equations as before except for the following update inverse covariance equation:

$$P^{-1}(n+1\,|\,n+1) = P^{-1}(n+1\,|\,n) + H^T(n+1)R^{-1}(n+1)H(n+1) \qquad (2.3.1)$$

There exists an additional equation for the Kalman gain $K(n+1)$ (see for example in [36]):

$$K(n+1) = P(n+1\,|\,n+1)H^T(n+1)R^{-1}(n+1) \qquad (2.3.2)$$

For more details on the Kalman Filter, one can refer to the original article of R. E. Kalman [18], or to any book on optimal filtering (e.g., [27, 13]).

2.4 Facts from Graph Theory

A common method of obtaining estimates of clock offsets between directly communicating pairs of nodes is based on the exchange of time-stamped packets. Viewing the network as a graph, this corresponds to finding estimates of clock offsets across the edges of the graph. These quantities must then be processed by the network to obtain estimates of the clock offsets at each node with respect to the reference clock.

Hence it is worthwhile to model the network as a directed graph $G = (V, \varepsilon)$ with $|V| = \tilde{\mathrm{N}}$ nodes $\{\Lambda_1, \Lambda_2, ..., \Lambda_{\tilde{N}}\}$ and $|\varepsilon| = m$ edges. Each edge represents the ability to transmit and receive packets between the corresponding pair of nodes. We will focus on an underlying network which consists of the entities that participate in the clock synchronization protocol. Let N denote this set of nodes and let $|\mathrm{N}|$ be its cardinality (the number of nodes). The edge connecting nodes Λ_i and Λ_j will be denoted by e_{ij} and the collection of all the edges by ε. We will assume throughout this thesis that all the edges are bidirectional, namely that if $e_{ij} \in \varepsilon$, then $e_{ji} \in \varepsilon$. Let us denote by N_i the set of nodes which are the neighbors of Λ_i, i.e., one edge away from node Λ_i, and let $|\mathrm{N}_i|$ be the number of such neighbors. For simplicity of notation, we exclude the existence of multiple edges between the same pair of nodes and also the edges from a node to itself. We consider a model in which only one out of the N nodes is a "reference time node" (the generalization for several reference time nodes is straightforward). Without loss of generality, we may assume that the reference time node is Λ_1. Our objective is to construct the optimal offset estimate for every node $u \in V \setminus \{1\}$.

The dimensions of the incidence matrix A are N (nodes number) $\times$ m (edges number). In the row corresponding to node Λ_i, we have an entry +1 for all edges of the form (i,*), an entry -1 for all edges of the form (*,i), and 0 otherwise.

For a connected graph, the rank of the incidence matrix is $N-1$, or one less than the number of nodes. Thus, deleting any row from the incidence matrix yields a full row rank matrix, which is called the reduced incidence matrix. Here, we will work with the $(N-1)\times m$ matrix obtained by deleting the row corresponding to the reference node Λ_1. For notational convenience, we use A to henceforth denote the reduced incidence matrix.

We present a simple illustrative example, similar to [41]. Consider the network in Figure 2.3. Here, for the construction of the matrix A, one can randomly choose the direction of the edges without affecting the results. In other words, the links are bidirectional but each link has a single entry in A.

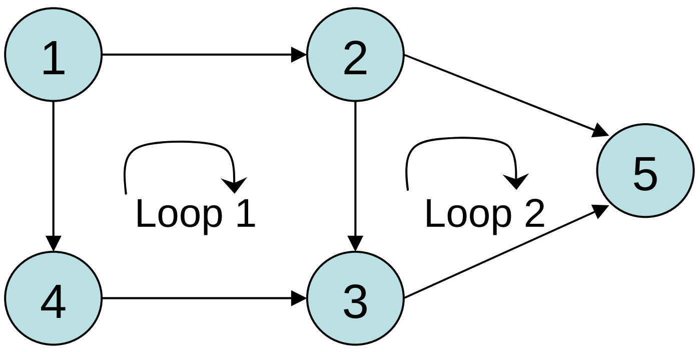


Figure 2.3. Example of a 5-node network.

The corresponding incidence matrix is given by:

$$A = \begin{array}{|c|cccccc|}
 & (1,2) & (2,3) & (3,4) & (1,4) & (2,5) & (3,5) \\
\cancel{1} & \cancel{+1} & \cancel{0} & \cancel{0} & \cancel{+1} & \cancel{0} & \cancel{0} \\
2 & -1 & +1 & 0 & 0 & +1 & 0 \\
3 & 0 & -1 & +1 & 0 & 0 & +1 \\
4 & 0 & 0 & -1 & -1 & 0 & 0 \\
5 & 0 & 0 & 0 & 0 & -1 & -1
\end{array}$$

If node number 1 is the reference, we will delete the first line of the above matrix in order to obtain the reduced incidence matrix A. We will use this matrix to obtain the state space model of the system in Section 4.4.

2.5 Matrix Analysis

In the convergence analysis (Section 6), we will show that our decentralized algorithm can be written in the form: $\overline{\tau}^{(k+1)} = M\overline{\tau}^{(k)}$. This is a standard iteration equation. Further, a sufficient and necessary condition to obtain convergence to zero from any initial guess is one where the spectral radius (the biggest eigen-value in absolute value) of the matrix M is strictly smaller than one:

$$\rho(M) < 1$$

In Non-Negative Matrix Theory (see the chapter on Gersgorin discs in [16]), it is proven that for a non-negative square matrix A, namely:

$$A = \left[a_{ij}\right] \; a_{ij} \geq 0 \quad i, j = 1, 2, .., n$$

We have:

$$\rho(A) \leq \min\left\{\max_i \sum_{j=1}^{n} \left|a_{ij}\right|, \max_j \sum_{i=1}^{n} \left|a_{ij}\right|\right\}$$

In particular:

$$\rho(A) \leq \max_i \sum_{j=1}^{n} \left|a_{ij}\right| \qquad (2.5.1)$$

Therefore, if all the row sums of the matrix A are smaller or equal to 1, and all the entries of A are non-negative, then $\rho(A) \leq 1$ (this is only a sufficient condition).

In addition, the following sharper result can be obtained.

Proposition 2.5.

Given a non-negative square matrix A with the following properties:

a) *All the row sums of the matrix A are smaller or equal to 1.*
b) *At least in one row this sum is strictly smaller than 1.*
c) *The matrix A is irreducible (i.e. we can move from any node to any other node through a direct trajectory).*

Then:

$$\rho(A) < 1$$

The proof of this sufficient condition is well known (see e.g., [16], chapter 7).

In our network model, the condition that the matrix M is irreducible requires the assumption that the graph that corresponds to the network is connected, namely that there exists a path between any pair of nodes in the network. Some additional important assumptions are that each node has at least one neighbor (not including the reference node) and that links are bidirectional (symmetric).

3. Related Work

In this section, we present a review of the papers that are most closely related to our work. First, we review the different accepted time synchronization protocols and then we consider a short literature survey on decentralized estimation (essentially on decentralized Kalman filtering), including consensus algorithms.

3.1 Time Synchronization Protocols

An early landmark paper in computer clock synchronization is Lamport's work [23] that elucidates the importance of virtual clocks in systems where causality is more important than absolute time. A distributed algorithm is proposed for synchronizing a system of logical clocks that can be used to totally order the events. Although this work focused on giving to the events a total order rather than qualifying the time difference between them, it has emerged as an important influence in sensor networks.

There is a large literature written on the art of synchronizing clocks in traditional networked systems. As we previously mentioned, the Network Time Protocol (NTP) is the widely accepted standard for synchronizing clocks over the Internet [28-30] and is notable for being scalable, self-configuring and robust to failures, in addition to being thoroughly tested. Nevertheless, this approach is vulnerable to sending delays and asymmetries in paths, and does not take advantage of the special properties of sensornet broadcasts. NTP is a client/server protocol used for synchronizing the internal clock of computers in standard networks and suggests a complete scheme for synchronizing the clocks with respect to the Coordinated Universal Time (UTC). NTP recommends data filtering and peer selection algorithms in order to reduce the offset which is the time difference between the clock and the UTC. Since NTP is used as a comparison benchmark in our simulation results, we briefly describe the procedure and more details can be found in [28-30].

According to NTP, each node Λ_i computes the round trip delay for each probe packet that traverses the edge e_{ij} based on the four timing fields recorded on the packet. Each node is sending probe packets to each one of its neighbors. Time is stamped on packet k_m by the sender Λ_i upon transmission ($T_i(k_m)$), and by the receiver Λ_j upon reception of the packet ($R_j(k_m)$). Then, the node Λ_j retransmits the packet back to the source ($T_j(k_m)$) and the source stamps its local time when receiving back the packet ($R_i(k_m)$). The computed round trip delay for packet k_m is given by:

$$RTT_{ij}(k_m) = \left(R_j(k_m) - T_i(k_m)\right) + \left(R_i(k_m) - T_j(k_m)\right).$$

The clock offset of node Λ_i relative to node Λ_j's clock is estimated as:

$$\frac{1}{2}\left[\left(R_j(k_m) - T_i(k_m)\right) - \left(R_i(k_m) - T_j(k_m)\right)\right].$$

NTP suggests the "minimum filter", which selects from the n most recent samples the sample with the lowest round trip delay. Each node estimates its relative clock offset with respect to a selected group between its neighbors, where neighbors which are hop count closer to the reference node are preferred, giving NTP its hierarchical nature. Finally, the offsets are averaged.

In 2003, the Classless Time Protocol (CTP) was proposed [14]. This protocol reduces the offset errors using a novel non-hierarchical approach that employs a peer to peer protocol in which each node sends and receives probe packets only to and from its neighbors. The approach exploits convex optimization theory in order to evaluate the impact of each clock offset on the overall objective function. In addition, the authors suggest the separation of the round-trip delays to one way components in order to obtain a filtered measurement and to increase the accuracy of the synchronization procedure. It was shown that CTP substantially outperforms hierarchical schemes based on NTP in terms of clock accuracy while preserving similar protocol complexity.

Solis, Borkar and Kumar [41, 12] have proposed an approach based on the concept of Least-Squares method, to smooth the set of estimates obtained by a packet exchange procedure. The accuracy of clock synchronization was improved by exploiting global network-wide constraints and the use of a completely asynchronous distributed algorithm employing only local broadcasts. The problem that results can be formulated as a distributed parameter estimation problem. They provide an alternate proof of the connection between the LS optimal set of estimates and electrical resistances in an equivalent resistive network. In addition, they analyze the convergence properties of the distributed synchronization algorithm they proposed. In fact, one can easily show that the CTP algorithm and the LS method are equivalent; the mathematical procedure is similar but written in two different ways.

In the scheme Reference-Broadcast Synchronization (RBS) described in [9, 10], an intermediate node transmits a reference packet and the other nodes record the time at which they receive it. They then exchange this recorded time to find the differences between their clocks. The fundamental property of this scheme is that it synchronizes a set of receivers with one another, as opposed to traditional protocols in which senders synchronize with receivers. Hence, RBS is quite accurate because it is completely insensitive to transmission delays and asymmetries. The most significant limitation of RBS is that it requires a network with a physical broadcast channel. It cannot be used, for example, in networks that employ point-to-point links as considered in this thesis. In [9], the authors argue that the time synchronization schemes, like NTP were developed for traditional networks (e.g., the Internet) and are not very efficient in Wireless Sensor Networks (WSNs) applications, where many assumptions have changed. Then, they design the requirements and the principles for WSN time synchronization.

More recently, in [26], the Flooding Time Synchronization Protocol (FTSP) is proposed; it uses MAC layer time stamping capabilities to eliminate several sources of error on the time synchronization process, and linear regression to compensate for the possible drifts in the clocks. A leader is elected through message exchanges and the global time is passed from the root to all the other nodes via flooding.

Karp, Elson, Estrin and Shenker [21] have considered the problem of minimum variance estimation based on global information, particularly for the RBS scheme of [9], and have shown that it satisfies the transitive property of offsets, i.e., the sum of optimal estimate of the offsets between the node pairs $\left(\Lambda_i,\Lambda_j\right)$ and $\left(\Lambda_j,\Lambda_k\right)$ is the optimal estimate of the offset between the node pair $\left(\Lambda_i,\Lambda_k\right)$. They have also analyzed the optimal error variance and related it to the resistance distance in the corresponding graph. Moreover, they show that the optimal pairwise synchronization and the globally consistent synchronization have the same technical answer and they treat clock skew and clock offset on different time scales.

There are several proposals for synchronizing clocks within a single broadcast domain (e.g., [31]). These methods exploit the special properties of broadcast media and achieve high precision. However, nodes that do not lie within the same broadcast domain cannot be synchronized. Since our focus is on global clock synchronization, these local approaches are not an efficient solution to our problem.

The most straightforward approach to synchronize clocks is to use the Global Positioning System (GPS), a constellation of satellites operated by the U.S. Department of Defense [19]. GPS provides accurate time synchronization relative to UTC [25], but its use is scarce in computer networks. GPS requires sensornet nodes to be equipped with special receivers, clear sky view and continuous reception of multiple satellites which is hard to accomplish inside buildings, underwater or beneath dense foliage. In addition, it may be too large, costly or high-power to a small and cheap sensor node.

Another quite different approach is that taken in [37], which does not directly synchronize clocks but instead refers to events in terms of their age. When exchanging these timestamps, they are updated to reflect the passage of time.

The last topic we present is related to time synchronization procedures using Kalman filtering. Two different schemes are considered. In [46], a time synchronization model on the Internet using Kalman filtering is proposed. The authors argue that the algorithm is more stable, more accurate and less sensitive to packet loss than the Simple Network Time Protocol (SNTP). SNTP [48] is a simplified version of NTP. The work in [20] is a heuristic approach based on adaptive Kalman filtering. The method is focused on a stochastic model of the network, which employs a KF and redundancy paths to achieve both an improved time and rate synchronization. The tests considered show an improvement of approximately two orders of magnitude in comparison to NTP. The schemes in [46, 20] are strongly related to our work but a number of essential differences exists. The problem considered in these works assumes that the network is composed of only two distinct computers and that just one set of measurements is available. Moreover, the proposed algorithms are totally centralized. On the other hand, in this research we are interested in large-scale systems with numerous nodes and the synchronization procedure has to be decentralized. Moreover, we will investigate the multiple measurement case through a recursive algorithm.

In this review, we evoke only the main algorithms to synchronize clocks in computer networks, or more precisely to estimate the offsets at each node with respect to the universal time. We did not present the accepted techniques for adjusting the clocks physically, because it is a solved problem and beyond the scope of this research.

3.2 Decentralized Estimation

In this part, we review several articles on decentralized (and distributed) estimation and more precisely on Decentralized Kalman Filter (DKF) algorithms. We investigate only dynamical state stochastic estimation and not the various literature on decentralized estimation of a deterministic unknown parameter corrupted by a noise.

Centralized implementation of the Kalman Filter [18], although optimal, does not provide robustness and scalability when it comes to complex large-scale dynamical systems with their measurements distributed on a large geographical region. This is the reason why several distributed estimation algorithms using the KF framework were proposed. Much of the existing research on distributed Kalman Filters focuses on sensor networks monitoring low dimensional systems [36]. This scenario addresses the problem on how to efficiently incorporate the distributed observations, also referred to in the literature as 'data fusion' [15].

Among the first works on DKF, the work in [15] assumes the presence of a fusion center or a central coordinator for combining the information from the various local processors. Then, the algorithm in [36] does not require any form of central processing facility. Each sensing node implements its own local KF to arrive at a partial decision which it then broadcasts to every other node. This algorithm leads to the same optimal centralized KF solution and is highly resilient to loss of one or more sensing nodes. The main drawback of this method is that a fully connected network is considered, i.e., each node must talk to every other node and the design of a convenient communication topology is problematic. In [39], the authors present some results on the problem of optimally combining static estimates from different sensors locations when the measurement noise processes are correlated. The works of [7, 44] extend the previous existing theory to the entire class of Luenberger observers. In [7], a necessary and sufficient condition for combining local KF estimates into a global KF estimate is proved. It is shown that decentralized estimators work by combining local estimates through weighting matrices. The low-power filtering scheme described in [44] implements Luenberger observers. By allowing the local stations to communicate at the rate that the estimates are desired, instead of the faster measurement rate, it saves power while maintaining robustness and optimality. All the previous solutions require that the local filters propagate a state vector that is the size of the global state and the knowledge of the sensor network topology.

The work in [4] explains the difference between decentralized and distibuted estimation. A decentralized network has no central facilities whereas a distributed network uses reduced order nodes operating in parallel to process local observations. Combining distribution and decentralization gives a new more efficient result and reduces the communication requirements. In [38], the DKF is applied to solve a common robotic application: cooperative localization. A new approach is presented where the centralized KF is decomposed in M modified Kalman filters each running on a separate robot. Moreover, the cross correlation terms between the different agents are computed in a distributed manner as well. A simulation example is considered where the authors show that an improvement in the localization accuracy is provided.

In [3], the authors consider the problem of estimating vector valued variables from noisy relative measurements in a decentralized fashion. The time synchronization and the sensor localization are obtained as special cases of this more general framework. Two different algorithms are proposed to compute the optimal estimate in a distributed and iterative manner. The case of temporary communication failures is treated and the algorithms are based on the Jacobi iterative method. The latter and the works performed in [14, 41, 12] formed the basis of the research presented in this thesis.

More recently, several methods for sparsely connected large-scale networks were considered. For example, in [17], a computationally efficient, sub-optimal distributed KF is used to estimate a sparsely connected, large-scale dynamical system monitored by a network of N sensors. Local Kalman Filters of much lower dimension than the global state are implemented at each sensor node. Shared observations and estimates across different local models are fused using bipartite fusion graphs and consensus averaging algorithms. The advantage of this scheme is that a low order KF is implemented at each sensor and the structure of the centralized error covariances is conserved. In other words, the proposed solution contrasts with existing methods for sensor networks that either replicate a KF (whose dimension is equal to the global state vector) at each sensor node or reduce the model dimension at the expense of decoupling the field dynamics into lower-dimensional models (non-optimal).

In non-linear and non-Gaussian scenarios, the DKF becomes inapplicable. Extended Kalman Filters, grid-based methods and Gaussian-sum filters are possible alternatives, but these all have limitations and information exchange is not as simple. The class of sequential Monte Carlo methods (or particle filtering) is attractive because of its power and flexibility. In [8], distributed implementations of particle filters are proposed. Since our domain of interest here is focused on DKF, we are not providing more details on these methods. The paper in [11] investigates several estimator architectures for determining the fleet state in the formation flying problem. This latter is non-linear and includes correlated states. The analysis shows that the proposed decentralized reduced order filters (like Schmidt-Kalman Filter) provide near optimal estimation results without excessive communication or computation and are preferable when compared to centralized and full order methods. The work in [32] presents an efficient method of multi-sensor estimation for systems with asynchronous observations. The architecture proposed is totally decentralized and each individual loop does not need to know about the other loops in the system. The resulting estimates are equivalent to the optimal centralized filter when the loops incorporate all the information available in the system.

Recently, Alriksson et al. [2] addresses the problem of distributed Kalman filtering, with focus on limiting the required communication bandwidth. The authors refer to a scenario where all the nodes desire an estimate of the full state of the observed system, there is no central utility and the communication takes place only between neighbors. The nodes merge their estimates by a weighted average of the neighbouring estimates. The weights are optimized off-line to yield a small estimation error covariance. This problem was generalized to time varying states in [42, 34] using consensus filters.

Next, we present a short literature survey on consensus algorithms.

3.3 Consensus Algorithms

We restrict the review to methods that are associated with data fusion in networks. Consensus problems are related to situations in which all the network members are required to achieve a common output value, using only local interactions and without access to a global coordinator. Consensus filters are distributed algorithms that allow calculation of average-consensus of time-varying signals. Two fundamental papers on consensus algorithms are given in [45, 35].

Xiao and Boyd [45] consider the problem of finding a linear iteration that yields distributed averaging consensus over a network, i.e., that asymptotically computes the average of the initial values given at the nodes. This article is the basis of various works in this field. In [35], the authors provide a theoretical framework for analysis of consensus algorithms for multi-agent networked systems. This is a general overview that investigates static and dynamic topologies as well as the continuous and discrete time cases. In addition, it provides diverse applications that are related to consensus problems and presents the simulation results for three different applications.

The work in [42] describes a dynamic consensus in order to obtain a distributed Kalman Filter for a network of agents. The algorithm consists of two loops: an outer loop for the KF and an inner loop for the weighted average consensus updating. Dynamic consensus allows to track in time different quantities and then to use them for distributed estimation. Olfati-Saber [34] solves the problem of distributed Kalman filtering for sensor networks by reducing it to two separate dynamic consensus problems in terms of weighted measurements and inverse covariance matrices. These problems were solved in a distributed way using a low-pass consensus filter for the fusion of the measurements and a band-pass consensus filter for the fusion of the inverse covariance matrices. It leads to an approximate distributed Kalman filtering algorithm that converges to the centralized optimal solution. Recently, in [6] the authors considered the problem of estimating the state of a scalar linear system from distributed noisy measurements. The estimation is performed via a two stage procedure which consists in (i) a standard decentralized Kalman-like measurement update and (ii) estimate fusion using consensus strategies. In order to attain this purpose two design parameters; the Kalman gain and the consensus matrix are designed by optimizing the steady state prediction error.

4. Model and Problem Definitions

4.1 Clock Model

We suppose that the clock drift at a node follows the linear form: $T_i(t) = \alpha_i t + \tau_i$, where α_i and τ_i are the skew (rate deviation) and the offset parameters respectively, t is the real time (or the reference time) and $T_i(t)$ is the local time (at node Λ_i). The above model is known as the two parameters linear model and is considered as a common way to model a clock in this context (see [41] and the references therein).

The time synchronization problem relates to the task of setting the clocks in the network so that they all agree upon a particular epoch with respect to a Coordinated Universal Time (UTC). Without loss of generality, one can assume that node Λ_1 is synchronized to the reference time:

$$\tau_1 = 0 \text{ and } \alpha_1 = 1$$

In our model, the clock synchronization relates to two different aspects: the rate synchronization (identical α_i) and the time offset synchronization (identical τ_i). For simplicity, we initially assume that all the clocks run at the same speed ($\alpha_i = \alpha_j = 1, \forall i, j$) and then we will relax this assumption in Section 9.

4.2 The Measurements

Each node is sending probe packets to each one of its neighbors. Figure 4.1 depicts the situation for the pair of neighboring nodes Λ_i and Λ_j. Time is stamped on packet k_m by the sender Λ_i upon transmission ($T_i(k_m)$) and by the receiver Λ_j upon reception of the packet ($R_j(k_m)$). Then, the node Λ_j is retransmitting the packet back to the source ($T_j(k_m)$) and the source stamps its local time when receiving the packet back ($R_i(k_m)$).

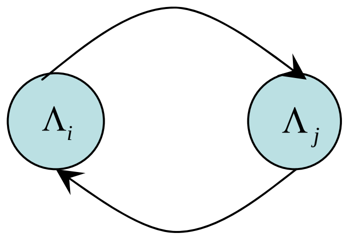


Figure 4.1. Communication between two neighboring nodes.

We intend to estimate the clock offsets by using these measurements data. Let us denote by $\Delta T_{ij}(k_m)$ the time difference between the transmission of probe packet k_m by node Λ_i, according to Λ_i clock, and the receiving time of the packet at node Λ_j according to its own clock. This implies:

$$\Delta T_{ij}(k_m) = R_j(k_m) - T_i(k_m) = x_{ij}(k_m) - \tau_i + \tau_j + \tilde{\varepsilon}_{ij} \tag{4.2.1}$$

Here, $x_{ij}(k_m)$ is the propagation delay and $\tilde{\varepsilon}_{ij}$ is an additive unknown noise that represents the random queuing delay and the other unknown influences. In other words, $\Delta T_{ij}(k_m)$ is equal to the one-way link delay experienced by probe packet k_m when traveling from node Λ_i to Λ_j ($x_{ij}(k_m)$), plus the difference between the two clock offsets.

Assuming that $x_{ij}(k_m) = x_{ji}(k_m)$ (the propagation delay is symmetric) we notice that:

$$\frac{1}{2}\left(\Delta T_{ij} - \Delta T_{ji}\right) = \tau_j - \tau_i + \varepsilon_{ij}$$

where:

$$\varepsilon_{ij} = \frac{1}{2}\left(\tilde{\varepsilon}_{ij} - \tilde{\varepsilon}_{ji}\right)$$

4.3 Problem Formulation

Our objective is to synchronize all the clocks in the network with the reference time. This is equivalent to estimate (using the Kalman Filter framework) α_i and τ_i at each network node. The algorithm is required to be decentralized and to converge to the optimal centralized solution. We initially assume that the offsets are time-invariant and that all the clocks run at the same speed (there is no skew), namely:

$$\alpha_i = \alpha_j = 1, \forall i, j\,.$$

These assumptions are reasonable if the clock synchronization procedure is applied at small enough time intervals. In the extensions section, we treat the case in which the previous assumptions are relaxed.

The first step is to find the state space model applied to our problem. Then, we will write the Kalman Filter equations and develop the results in a centralized form. This result will represent the optimal clock offset vector in the MMSE sense. Later, we proceed to develop a decentralized implementation of the preceding filtering algorithm. The last steps include extending our algorithm to the case of different clock skews and to treat several interesting extensions.

4.4 State Space Model

Let us define the state vector by the following column vector:

$$\underline{x}(n) \triangleq \left(\tau_1 = 0, \tau_2, \ldots \tau_N\right)^T$$

Here, τ_i is the offset of the node Λ_i. In this part, the offsets are assumed to be time invariant and we consider that all the clocks run exactly at the same speed (i.e., $\alpha_i = \alpha_j = 1$ there is no skew).
As we explained before, the measurements data for each pair of neighbor nodes is given by:

$$y_{ij} = \hat{O}_{ij} \triangleq \frac{1}{2}\left(\Delta T_{ij} - \Delta T_{ji}\right) = \tau_j - \tau_i + \varepsilon_{ij} \qquad (4.4.1)$$

Remark: $\hat{O}_{ij}$ is the conventional notation for this category of measurements.
Equation (7) states that the relative measurement y_{ij} for each pair of neighboring nodes is given by the difference between the offsets plus an additive noise ε_{ij}.

According to the previous notation (see Section 2.4), the measurement equation of the state space model is related to the reduced incidence matrix A. Consequently, the state space model is given by:

$$\begin{cases} \underline{x}(n+1) = \underline{x}(n) + \underline{w}(n) \\ y(n) = A^T \underline{x}(n) + \underline{v}(n) \end{cases} \tag{4.4.2}$$

Here, $\underline{w}(n)$ and $\underline{v}(n)$ are the system and measurement noises respectively. We assume that the offsets are time invariant, so we will neglect the process noise $\underline{w}(n)$ for the time being. In Section 8.2, we will consider the more general case where $\underline{w}(n)$ is incorporated back.
$\underline{v}(n)$ is the measurement noise and is assumed to be in accordance with the Kalman Filter assumptions (see Section 2.2). For simplicity, in all the state space representations it is assumed that the sampling time is uniform and equal to one.

5. Centralized and Decentralized Algorithms: Single Measurement Update

In this section, we first present the Kalman Filter equations applied to our problem using two different approaches. Then, we develop a decentralized version of this filtering algorithm, and we will prove its convergence to the optimal centralized solution in the next section. By using the KF framework, we obtain the existing results from the literature as special cases and extend them to a more general framework. We consider separately the cases where the initial inverse covariance matrix is diagonal and non-diagonal. We assume here that only one set of measurements is available. We will extend our results to the case of multiple measurement sets in Section 7 by the use of a recursive decentralized algorithm.

5.1 Kalman Filter Equations

Let us present the KF equations applied to our case. As we previously stated, the state vector is defined by:

$$\underline{x}(n) \triangleq \left(\tau_1 = 0, \tau_2, \ldots \tau_N\right)^T$$

Here, τ_i is the offset of the node Λ_i. In this part, the offsets are assumed to be time invariant and we consider that all the clocks run exactly at the same speed (i.e., $\alpha_i = \alpha_j = 1$, there is no skew). As we have previously explained, the state space model is given by:

$$\begin{cases} \underline{x}(n+1) = \underline{x}(n) + \underline{w}(n) \\ \underline{y}(n) = A^T \underline{x}(n) + \underline{v}(n) \end{cases}$$

In the previous notation, we have: $\Phi = I$, $H = A^T$. The Kalman Filter equations are therefore:

Time update (prediction):

$$\begin{cases} \hat{\underline{x}}(n+1 \mid n) = \hat{\underline{x}}(n \mid n) \\ P(n+1 \mid n) = P(n \mid n) + Q \end{cases}$$

Measurement update:

$$\begin{cases} \hat{\underline{x}}(n+1 \mid n+1) = \hat{\underline{x}}(n+1 \mid n) + K(n+1)\left[\underline{y}(n+1) - A^T \hat{\underline{x}}(n+1 \mid n)\right] \\ K(n+1) = P(n+1 \mid n) A \left[A^T P(n+1 \mid n) A + R\right]^{-1} \\ P(n+1 \mid n+1) = \left[I - K(n+1) A^T\right] P(n+1 \mid n) \end{cases}$$

Let us substitute the time update equation into the measurement update equation:

$$\begin{cases} \hat{\underline{x}}(n+1 \mid n+1) = \hat{\underline{x}}(n+1 \mid n) + \left[P(n \mid n) + Q\right] A \left[A^T \left(P(n \mid n) + Q\right) A + R\right]^{-1} \left[\underline{y}(n+1) - A^T \hat{\underline{x}}(n+1 \mid n)\right] \\ P(n+1 \mid n+1) = \left\{I - \left(P(n \mid n) + Q\right) A \left[A^T \left(P(n \mid n) + Q\right) A + R\right]^{-1} A^T\right\} \left(P(n \mid n) + Q\right) \end{cases}$$

Then, the combined Kalman Filter equation (using the information form) is given by:

$$\underline{\hat{x}}(n+1\,|\,n+1)=\underline{\hat{x}}(n\,|\,n)+\left[\left(P(n\,|\,n)+Q\right)^{-1}+AR^{-1}A^{T}\right]^{-1}AR^{-1}\left[y(n+1)-A^{T}\underline{\hat{x}}(n\,|\,n)\right] \qquad (5.1.1)$$

Next, we will obtain the above equation through the Least-Squares optimization approach. Afterwards, we will develop a decentralized version (requiring only local broadcast) of this filtering algorithm and in Section 6, we will prove its convergence to the optimal centralized solution.

5.2 The LS approach

We consider the single measurement update case (only one set of measurements is available) and later on, we will extend the algorithms to the multiple measurement case by proposing a recursive algorithm (Section 7). We start with the pair of parameters $\overline{x}_0, P_0$ and our goal is to find $\underline{\hat{\tau}}_{opt}$ by using the Kalman Filter equations in a centralized fashion. $\overline{x}_0$ and P_0 represent the a-priori knowledge and we want to include this information together with the measurements to find an optimal estimate. This is important, because this initial knowledge can improve the quality of the estimation.

The KF solution is equivalent to the minimum of the following expression (see the proof in Appendix A):

$$J=(\underline{x}-\overline{x}_0)^{T}P_0^{-1}(\underline{x}-\overline{x}_0)+(y-A^{T}\underline{x})^{T}R^{-1}(y-A^{T}\underline{x})\xrightarrow{\min}\hat{x}(0) \qquad (5.2.1)$$

The first term of the objective function is related to the initial knowledge whereas the second term is associated with the single set of measurements and its corresponding covariance matrix.

It is preferable to solve the above deterministic LS problem than to solve the KF equations directly. The main reason is that the KF solution gives only a centralized algorithm and it is difficult to decentralize the procedure, as we will see in the next section.

We want to find the vector $\underline{\hat{x}}_{opt}$ that minimizes the above objective function. Hence, we have to compute the gradient with respect to the vector $\underline{x}$ and set it to zero:

$$\nabla_{\underline{x}}J=\left(AR^{-1}A^{T}+P_0^{-1}\right)\underline{x}-\left(AR^{-1}\right)y-P_0^{-1}\overline{x}_0=0$$

$$\underline{\hat{x}}_{opt}=\left(AR^{-1}A^{T}+P_0^{-1}\right)^{-1}\left(AR^{-1}y+P_0^{-1}\overline{x}_0\right) \qquad (5.2.2)$$

$\underline{\hat{x}}_{opt}$ represents the optimal vector of offsets. One can note that the above equation is equivalent to the combined Kalman Filter equation that we obtained previously. In addition, the posterior error covariance matrix is as follows:

$$E\left[\left(\underline{x}-\underline{\hat{x}}\right)\left(\underline{x}-\underline{\hat{x}}\right)^{T}\right]=\left(AR^{-1}A^{T}+P_0^{-1}\right)^{-1} \qquad (5.2.3)$$

In the statistical WLS case, we substitute $P_0^{-1} = 0$ in the objective function. The centralized optimal solution is known as the BLUE (Best Linear Unbiased Estimator) and is given by:

$$\underline{\hat{x}} = \left(AR^{-1}A^T\right)^{-1}\left(AR^{-1}y\right)$$

In addition, the error covariance matrix is as follows:

$$E\left[\left(\underline{x} - \underline{\hat{x}}\right)\left(\underline{x} - \underline{\hat{x}}\right)^T\right] = \left(AR^{-1}A^T\right)^{-1}$$

In order to compute the optimal estimate directly (in a centralized manner), one seems to need all the measurements associated with their error covariances and the topology of the entire network. For networks with a large number of measurements, doing so will be prohibitively expensive in terms of energy consumption, bandwidth and communication time. Hence, it is more preferable to compute the estimates in a decentralized fashion, employing only local broadcasts. By decentralized we mean that at every step, each node computes its own estimate and the data required are obtained through communication with its one-hop neighbors.

5.3 Decentralized Algorithm: Optimality Equations

Our purpose is to develop a decentralized algorithm that estimates the offsets of each node over the network. In this section, we consider only the single measurement case. The multiple measurement scenario will be considered later in Section 7. As in the previous section, we start with the parameters $\overline{x}_0, P_0$ and our goal is to find $\underline{\hat{\tau}}_{opt}$ by using the Kalman Filter in a decentralized fashion. In other words, we are looking for the minimizing solution $\hat{x}\left(0\,|\,0\right)$ of the following objective function:

$$J = (\underline{x} - \overline{x}_0)^T P_0^{-1} (\underline{x} - \overline{x}_0) + (y - A^T \underline{x})^T R^{-1} (y - A^T \underline{x})$$

a) The Basic LS Algorithm

We first demonstrate our solution approach for the simplest case where only the second term is present i.e., $P_0^{-1} = 0$ and all the measurements are equally weighted ($R = R^{-1} = I$). This corresponds to the Least-Squares solution presented in Section 2.1.

In this case, the objective function is given by:

$$J = (y - A^T \underline{x})^T (y - A^T \underline{x}) = \sum_{\substack{i,j \\ j \in N_i}} \left(\hat{O}_{ji} - \tau_i + \tau_j\right)^2$$

Differentiating J with respect to each one of the coordinates τ_i leads to:

$$\frac{\partial J}{\partial \tau_i} = \left(AA^T\right)_i \underline{x} - A_i y = -2\sum_{j \in N_i}\left(\hat{O}_{ji} - \tau_i + \tau_j\right) = -2\left[-\tau_i \cdot \sum_{j \in N_i} 1 + \sum_{j \in N_i}\left(\hat{O}_{ji} + \tau_j\right)\right] = 0$$

Let us substitute the following relations:

$$\begin{cases} \left(AA^T\right)_i \underline{x} = |N_i|\tau_i - \sum\limits_{j\in N_i} \tau_j \\ A_i y = \sum\limits_{j\in N_i} \hat{O}_{ji} \end{cases}$$

$$\frac{\partial J}{\partial \tau_i} = |N_i|\tau_i - \sum_{j\in N_i}\left(\hat{O}_{ji} + \tau_j\right) = 0$$

From this, we get:

$$\boxed{\tau_i = \frac{1}{|N_i|}\sum_{j\in N_i}\left(\hat{O}_{ji} + \tau_j\right)} \qquad (5.3.1)$$

The above equation must be satisfied by the optimal solution of the offset estimation problem. While this is a set of linear equations, a direct solution cannot be carried out in a decentralized manner. Instead, we will implement a decentralized iterative algorithm and show its convergence to the optimal centralized solution. This algorithm follows the Jacobi iteration that was described in Section 2.1. We will define the iterative procedure for the general case at the end of this section and we will prove its convergence in the next chapter. The above equation has a very simple interpretation. Each node computes its offset estimate as the average of all its neighbors' estimates plus the corresponding relative measurements. This procedure is the same as in [41, 12] and one can easily show that this is equivalent to the algorithm in [14]. Our objective is to extend the previous result to a wider framework and we will obtain this procedure as a special case of a more general algorithm.

b) Weighted Least-Squares

Now, we incorporate the weighting matrix while assuming that R^{-1} is a diagonal and positive definite matrix. These assumptions are reasonable since the matrix R represents the covariance of the uncorrelated measurement noise.

$$J = (y - A^T\underline{x})^T R^{-1}(y - A^T\underline{x}) = \sum_{\substack{i,j \\ j\in N_i}} \frac{1}{r_{ji}}\left(\hat{O}_{ji} - \tau_i + \tau_j\right)^2$$

$$\frac{\partial J}{\partial \tau_i} = \left(AR^{-1}A^T\right)_i \underline{x} - \left(AR^{-1}\right)_i y = -2\sum_{j\in N_i}\frac{1}{r_{ji}}\left(\hat{O}_{ji} - \tau_i + \tau_j\right) = 0$$

$$\tau_i \cdot \sum_{j\in N_i}\frac{1}{r_{ji}} - \sum_{j\in N_i}\frac{1}{r_{ji}}\left(\hat{O}_{ji} + \tau_j\right) = 0$$

From this, we get:

$$\boxed{\tau_i = \frac{1}{\sum\limits_{j\in N_i}\frac{1}{r_{ji}}} \cdot \sum_{j\in N_i}\frac{1}{r_{ji}}\left(\hat{O}_{ji} + \tau_j\right)} \qquad (5.3.2)$$

We observe that the above formula is quite logical and consistent with our expectations namely, each measurement is multiplied by a weight equal to $\left(r_{ji}\right)^{-1}$ according to its quality. Moreover, we can check easily that if $R = I$, *i.e.*, $r_{jk} = 1 \quad \forall j,k \in N_j$ the result reduces to the previous LS case. The coefficients $\left(r_{ji}\right)^{-1}$ are related to the measurement $\hat{O}_{ji}$, therefore $\left(r_{ji}\right)^{-1} = \left(r_{ij}\right)^{-1}$ since each measurement is issued by a bidirectional exchange between the pair of nodes.

Let us rewrite the previous equation in the following way:

$$\left(\sum_{j\in N_i}\frac{1}{r_{ji}}\right)\cdot\tau_i = \sum_{j\in N_i}\frac{1}{r_{ji}}\left(\hat{O}_{ji}+\tau_j\right) = \sum_{j\in N_i}\frac{1}{r_{ji}}\hat{O}_{ji} + \sum_{j\in N_i}\frac{1}{r_{ji}}\tau_j$$

The above equation can be solved iteratively using a decentralized version of the Jacobi method (coordinate-wise), and converges to the optimal centralized solution (see the proof in [3]).

Next, we will consider the general framework that includes the initial covariance matrix P_0 in the objective function. The analysis is divided in two cases: diagonal and non-diagonal initial covariance matrix.

c) General Framework: Diagonal P_0

Finally, let us solve the original problem where the objective function is composed of two distinct terms:

$$J = (\underline{x}-\overline{x}_0)^T P_0^{-1}(\underline{x}-\overline{x}_0) + (y - A^T\underline{x})^T R^{-1}(y - A^T\underline{x})$$

Now,

$$\frac{\partial J}{\partial \tau_i} = \left(AR^{-1}A^T\right)_i \underline{x} - \left(AR^{-1}\right)_i y + \left(P_0^{-1}\right)_{i*}\underline{x} - \left(P_0^{-1}\right)_{i*}\overline{x}_0 = 0$$

$$\tau_i\sum_{j\in N_i}\frac{1}{r_{ji}} - \sum_{j\in N_i}\frac{1}{r_{ji}}\left(\hat{O}_{ji}+\tau_j\right) + \left(P_0^{-1}\right)_{i*}\hat{x} - \left(P_0^{-1}\right)_{i*}\overline{x}_0 = 0$$

Here, $\left(P_0^{-1}\right)_{i*}$ is the i-th row of the matrix P_0^{-1}.

One can make the logical assumption that the initial inverse covariance matrix P_0^{-1} is a diagonal matrix. Indeed, P_0^{-1} represents the initial correlation between the different clocks in the network, and there is no reason to have some a-priori knowledge of the cross correlation terms but only on the variances of each clock (diagonal terms).

In the case where P_0^{-1} is a diagonal matrix, we can obtain:

$$\tau_i\left(\sum_{j\in N_i}\frac{1}{r_{ji}} + \frac{1}{p_i}\right) = \sum_{j\in N_i}\frac{1}{r_{ji}}\left(\hat{O}_{ji}+\tau_j\right) + \frac{1}{p_i}\cdot\tau_i(0)$$

This implies:

$$\boxed{\tau_i = \frac{1}{\left(\sum_{j\in N_i}\frac{1}{r_{ji}}+\frac{1}{p_i}\right)}\cdot\left[\sum_{j\in N_i}\frac{1}{r_{ji}}\left(\hat{O}_{ji}+\tau_j\right)+\frac{\tau_i(0)}{p_i}\right]} \tag{5.3.3}$$

In other words, τ_i is given by the weighting average between the adjacent measurements and the initial knowledge related to it. We can notice that if the matrix P_0^{-1} is identically equal to zero, we obtain the same equation as the previous Weighted Least-Squares case (5.3.2).

d) General Framework: Non-Diagonal P_0

In general, the initial covariance matrix P_0 need not be a diagonal matrix. As we will explain at the end of Section 6, even if the initial covariance matrix P_0 is chosen to be diagonal, after the first iteration of the Kalman Filter, the inverse covariance matrix will not preserve its diagonal structure. Hence, if we have a-priori knowledge of the system or if multiple sets of measurements are available, we must consider the case in which the covariance matrix is not assumed to be diagonal.

In this more general case, we get:

$$\frac{\partial J}{\partial \tau_i} = \tau_i\sum_{j\in N_i}\frac{1}{r_{ji}} - \sum_{j\in N_i}\frac{1}{r_{ji}}\left(\hat{O}_{ji}+\tau_j\right)+\sum_{k=1}^{N}\left(P_0^{-1}\right)_{ik}\left(\tau_k-\tau_k(0)\right)=0$$

$$\tau_i\sum_{j\in N_i}\frac{1}{r_{ji}} - \sum_{j\in N_i}\frac{1}{r_{ji}}\left(\hat{O}_{ji}+\tau_j\right)+\sum_{\substack{k=1\\k\neq i}}^{N}\left(P_0^{-1}\right)_{ik}\left(\tau_k-\tau_k(0)\right)+\left(P_0^{-1}\right)_{ii}\left(\tau_i-\tau_i(0)\right)=0$$

$$\tau_i\left(\sum_{j\in N_i}\frac{1}{r_{ji}}+\left(P_0^{-1}\right)_{ii}\right)=\sum_{j\in N_i}\frac{1}{r_{ji}}\left(\hat{O}_{ji}+\tau_j\right)+\left(P_0^{-1}\right)_{ii}\tau_i(0)-\sum_{\substack{k=1\\k\neq i}}^{N}\left(P_0^{-1}\right)_{ik}\left(\tau_k-\tau_k(0)\right)$$

This implies:

$$\boxed{\tau_i = \frac{1}{\left(\sum_{j\in N_i}\frac{1}{r_{ji}}+\left(P_0^{-1}\right)_{ii}\right)}\left[\sum_{j\in N_i}\frac{1}{r_{ji}}\left(\hat{O}_{ji}+\tau_j\right)+\left(P_0^{-1}\right)_{ii}\tau_i(0)-\sum_{\substack{k=1\\k\neq i}}^{N}\left(P_0^{-1}\right)_{ik}\left(\tau_k-\tau_k(0)\right)\right]} \tag{5.3.4}$$

The main problem in equation (5.3.4) is that each node needs to communicate with all the other nodes and not only with its neighbors. Thus, in the case where the matrix P_0 is not diagonal, each node has to know the global topology of the entire network. As we previously explained, the initial covariance matrix P_0 can be assumed to be diagonal. However, after applying the Kalman Filter equations, the covariance matrix $P(k\,|\,k)$ will not be diagonal anymore.

In summary, we have presented the optimality equations for the four different cases. In this section, we only considered the case with a single set of measurements. In order to implement the optimality equations, we propose an iterative decentralized algorithm presented next.

5.4 Decentralized Algorithm: Iterative Equations

The four decentralized optimality equations we previously presented can be applied to a network in order to estimate the clock offsets at each node with respect to the reference time. The suggested distributed optimization is iterative. There are many iterative methods that can be used [43, 22]. Each iterative algorithm can be implemented either in a synchronous manner or in an asynchronous way, see [5] for more details about parallel and distributed computations. In the remainder of this work, we focus on the synchronous versions of the different algorithms. Convergence can be accelerated by over-relaxation techniques which are standard in numerical analysis [5]. In this research thesis, we will not be concerned with the number of iterations and rate of convergence, as long as convergence is achieved after an infinite number of iterations.

As we previously mentioned, if only one set of measurements is available, we can apply the algorithm assuming that P_0 is a diagonal matrix. The reason is that at the beginning, it is reasonable to assume that the a-priori covariance matrix is diagonal, i.e., the initial offsets are uncorrelated. In Section 7, we will present a recursive version of the previous algorithms for the multiple measurement case.

For the single measurement case, the corresponding decentralized synchronous algorithm that implements the optimal equation in (5.3.3) is given by (assuming that P_0 is diagonal):

$$\hat{\tau}_i^{(k+1)} = \frac{1}{\left(\sum_{j\in N_i} \frac{1}{r_{ji}} + \frac{1}{p_i}\right)} \cdot \left[\sum_{j\in N_i} \frac{1}{r_{ji}}\left(\hat{O}_{ji} + \hat{\tau}_j^{(k)}\right) + \frac{\tau_i(0)}{p_i}\right] \qquad (5.4.1)$$

Initialization: $\hat{\tau}_i^{(0)} = \tau_i(0) \quad i = 2,3,...N$. Here, $k \geq 0$ is the iteration number.

In other words, we obtained in (5.3.3) the optimal equation that computes the offsets at each node in a decentralized fashion. In order to implement this optimal equation, we propose the iterative synchronous algorithm in (5.4.1). The procedure in (5.4.1) requires only local broadcasts (communication with neighbors) and as we will show in the next section, converges to the optimal centralized solution (after an infinite number of iterations in all the nodes).

Let us summarize the procedure for applying the algorithm in (5.4.1). First of all, we assume that the nodes detect their neighbors and exchange bilateral packets to obtain their relative measurements $\hat{O}_{ji}$. In addition, they exchange their relative inverse covariances $\left(r_{ji}\right)^{-1}$ and the inverse initial covariance $\left(p_i\right)^{-1}$. The initial offset $\tau_i(0)$ are known at each node, so the quantity $\sum_{j\in N_i} \frac{1}{r_{ji}} + \frac{1}{p_i}$ can be computed at the beginning of the procedure.

After the deployment of the network, the reference node initializes its estimate to 0 and never changes it. Every other node in the network initializes its estimate to an arbitrary value. At the start of iteration $k+1$, each node sends its most recent estimate $\hat{\tau}_i^{(k)}$ to its neighbors along with the corresponding iteration number. It also gathers the estimates of its neighbors, $\hat{\tau}_j^{(k)} \quad \Lambda_j \in N_i$ and then updates its own estimate by applying the above equation for $\hat{\tau}_i^{(k+1)}$. The algorithm is summarized in Table 5.1.

Name: Decentralized Synchronous Kalman Filter

Assumptions: - The inverse initial covariance matrix P_0^{-1} is diagonal.

- Only one set of measurements is available.

Goal: Compute in a decentralized manner the offsets estimates at each network node (except for the reference) that approach the optimal centralized estimates.

Initialization: $\tau_1^{(k)} = 0 \ \forall k$, $\hat{\tau}_i^{(0)}$ is arbitrary for $i \in V \setminus \{1\}$.

After deployment, each node $i \in V \setminus \{1\}$ performs:

1. Detect its neighbors N_i.
2. Identify the inverse initial covariance $\frac{1}{p_i}$ and the initial offset $\tau_i(0)$.
3. Obtain one set of relative measurements $\hat{O}_{ji}$ and the associated inverse covariances $\frac{1}{r_{ji}}$ for every $j \in N_i$. Compute $\sum_{j \in N_i} \frac{1}{r_{ji}} + \frac{1}{p_i}$.

At every iteration k, each node Λ_i performs:

4. Send $\hat{\tau}_i^{(k)}$ and k to its neighbors $j \in N_i$. Obtain $\hat{\tau}_j^{(k)} \quad j \in N_i$.
5. Compute $\hat{\tau}_i^{(k+1)}$ from the previous quantities, using (5.4.1).

Table 5.1. Summary of the Decentralized Synchronous Kalman Filter Algorithm.

The end of the algorithm is determined according to a termination condition on the absolute difference between the estimates at two successive iterations:

$$\left| \hat{\tau}_i^{(k+1)} - \hat{\tau}_i^{(k)} \right| < \varepsilon \quad i \in N \setminus \{1\} \qquad (5.4.2)$$

In the next section, we will prove the convergence of the previous iterative decentralized algorithm to the optimal centralized solution.

6. Convergence Analysis

Let us recall that the clock synchronization problem considered in this thesis (with a single set of measurements) can be summarized into the following LS optimization problem:

$$J = (\underline{x} - \overline{x}_0)^T P_0^{-1} (\underline{x} - \overline{x}_0) + (y - A^T \underline{x})^T R^{-1} (y - A^T \underline{x})$$

In order to find the optimal solution, one has to minimize the above objective function with respect to the vector $\underline{x}$. In the previous chapter, we divided the analysis into four special cases and we obtained the optimal solution for each case. Then, one can implement the optimal solution using an iterative decentralized algorithm (equivalent to the Jacobi method). We now prove the convergence of the previous decentralized clock synchronization algorithms to the optimal centralized solution. We consider the most general case where the initial covariance matrix P_0 is not assumed to be a diagonal matrix. In this case, the synchronous decentralized algorithm is given by:

$$\hat{\tau}_i^{(k+1)} = \frac{1}{\left(\sum_{j \in N_i} \frac{1}{r_{ji}} + \left(P_0^{-1}\right)_{ii} \right)} \left[\sum_{j \in N_i} \frac{1}{r_{ji}} \left(\hat{O}_{ji} + \hat{\tau}_j^{(k)} \right) + \left(P_0^{-1}\right)_{ii} \tau_i(0) - \sum_{\substack{m=1 \\ m \neq i}}^{N} \left(P_0^{-1}\right)_{im} \left(\hat{\tau}_m^{(k)} - \tau_m(0) \right) \right] \quad (6.1)$$

Initialization: $\hat{\tau}_i^{(0)}(1) = \tau_i(0) \quad i = 2,3,...N$. Here, $k \geq 0$ is the iteration number.

Theorem 6.1

Suppose that:

a) A single set of measurements is available.

b) The matrix R is diagonal and PSD, that is: $\infty > \left(r_{ji}\right)^{-1} \geq 0 \; \forall i, j$.

c) The offsets are time-invariant.

d) The initial state vector $\overline{x}_0$ *is* known.

e) The initial covariance matrix P_0 is known *and is an M-matrix, namely:*

$$\begin{cases} \sum_j \left(P_0^{-1}\right)_{ij} \geq 0 \\ \left(P_0^{-1}\right)_{ii} \geq 0 \\ \left(P_0^{-1}\right)_{ij} \leq 0 \; \left(i \neq j\right) \end{cases} \quad (6.2)$$

f) The clock adjustment operation in (6.1) is applied synchronously by all nodes ($i = 2,3,...N$*) in all iterations.*

Then, the iterated estimators $\hat{\tau}_i^{(k)}(n) \quad i = 2,3,...N$ *converge (as* $k \to \infty$*) to the optimal offsets that minimize the objective function:*

$$J = (\underline{x} - \overline{x}_0)^T P_0^{-1} (\underline{x} - \overline{x}_0) + (y - A^T \underline{x})^T R^{-1} (y - A^T \underline{x})$$

namely, the set of offsets that would have been obtained by performing the centralized optimal protocol.

In the remainder of this section, we present the proof of this result. For the first two special cases (Least-Squares and Weighted Least-Squares), the convergence can be proven in the following way. The first step consists of proving that the objective function to be minimized $J\left(\underline{\tau}\right)$ is non-increasing in time. The second step is to show that if the clock adjustment operation is applied by all nodes in all iterations, the set of estimated offsets converges to the set of offsets that minimize the objective function i.e., the set of offsets that would have been obtained by performing the centralized protocol (for the LS case, see the proof in [14]). For the WLS algorithm the convergence condition is that: $\infty > \left(r_{ji}\right)^{-1} \geq 0 \ \forall i, j$ or in other words, a sufficient condition is that the matrix R is diagonal and PSD (Positive Semi-Definite).

Here, we will use another more convenient technique to prove the convergence of the proposed algorithm, in the general case. Let us recall that the general objective function is given by:

$$J = (\underline{x} - \overline{x}_0)^T P_0^{-1} (\underline{x} - \overline{x}_0) + (y - A^T \underline{x})^T R^{-1} (y - A^T \underline{x})$$

Let us analyze the convergence properties of the general case, where P_0 is not necessarily assumed to be a diagonal matrix. We note that the iteration (6.1) cannot be easily decentralized when P_0 is not diagonal as we explained in Section 5. However, the iteration is still well defined mathematically.

The synchronous iteration can be written in vector form:

$$\hat{\tau}^{(k+1)} = \hat{\tau}^{(k)} - \left(\tilde{D} + \tilde{P}_0\right)^{-1} \left(AR^{-1}A^T \hat{\tau}^{(k)} - AR^{-1}y - P^{-1}\overline{x}_0 + P_0^{-1}\hat{\tau}^{(k)}\right) \tag{6.3}$$

Here: $$\left(\tilde{D}^{-1}\right)_{ij} = \begin{cases} \dfrac{1}{\sum\limits_{j \in N_i} \dfrac{1}{r_{ji}}} & i = j \\ 0 & otherwise \end{cases} \quad \text{and} \quad \left(\tilde{P}_0\right)_{ij} = \begin{cases} \left(P_0^{-1}\right)_{ij} & i = j \\ 0 & otherwise \end{cases}$$

The optimal solution (equivalent to perform the centralized protocol) is the same as before:

$$\tau^* = \left(AR^{-1}A^T + P_0^{-1}\right)^{-1} \left(AR^{-1}y + P_0^{-1}\overline{x}_0\right)$$

Let us define: $$\overline{\tau}^{(k)} \triangleq \hat{\tau}^{(k)} - \tau^* \tag{6.4}$$

Then we get:

$$\begin{aligned} \overline{\tau}^{(k+1)} &\doteq \hat{\tau}^{(k+1)} - \tau^* = \hat{\tau}^{(k)} - \left(\tilde{D} + \tilde{P}_0\right)^{-1} \left(AR^{-1}A^T \hat{\tau}^{(k)} - AR^{-1}y - P_0^{-1}\overline{x}_0 + P_0^{-1}\hat{\tau}^{(k)}\right) - \\ &\quad - \left(AR^{-1}A^T + P_0^{-1}\right)^{-1} \left(AR^{-1}y + P_0^{-1}\overline{x}_0\right) \\ \overline{\tau}^{(k+1)} &= \left[I - \left(\tilde{D} + \tilde{P}_0\right)^{-1} \left(AR^{-1}A^T + P_0^{-1}\right)\right] \hat{\tau}^{(k)} - \left(AR^{-1}A^T + P_0^{-1}\right)^{-1} \left(AR^{-1}y + P_0^{-1}\overline{x}_0\right) + \\ &\quad + \left(\tilde{D} + \tilde{P}_0\right)^{-1} \left(AR^{-1}A^T + P_0^{-1}\right) \underbrace{\left(AR^{-1}A^T + P_0^{-1}\right)^{-1} \left(AR^{-1}y + P_0^{-1}\overline{x}_0\right)}_{\tau^*} \\ \overline{\tau}^{(k+1)} &= \left[I - \left(\tilde{D} + \tilde{P}_0\right)^{-1} \left(AR^{-1}A^T + P_0^{-1}\right)\right] \overline{\tau}^{(k)} \end{aligned}$$

We denote:

$$M \triangleq I - \left(\tilde{D} + \tilde{P}_0\right)^{-1}\left(AR^{-1}A^T + P_0^{-1}\right) \tag{6.4}$$

We have the equivalent iteration:

$$\overline{\tau}^{(k+1)} = M\overline{\tau}^{(k)}$$

Thus, the convergence of the sequence $\hat{\tau}^{(k)}$ to $\overline{\tau}^*$ is equivalent to the convergence of $\overline{\tau}^{(k)}$ to the zero vector, which is determined by the matrix M given in (6.4). The necessary and sufficient condition for this to happen is that the spectral radius of M is strictly smaller than 1.

According to proposition 2.5, the above iteration equation converges to zero if the sufficient conditions apply for the matrix iteration M. In other words, the row sums of the matrix M are less or equal than 1 (and at least in one row this sum is strictly smaller than 1), all the entries of the matrix M are non-negative and that the matrix M is irreducible; i.e., we can move from any node to any other node by a direct trajectory. In our network model, this last condition that the matrix M is irreducible requires the assumption that the graph that corresponds to the network is connected, namely there exists a path between any pair of nodes in the network. Some additional important assumptions are that each node has at least one neighbor (not including the reference node) and that the links are bidirectional (symmetric).

The structure of the matrix M can be determined by inspection as the following:

$$M_{ii} = 0 \quad M_{ij} = \begin{cases} \dfrac{-\left(P_0^{-1}\right)_{ij} + \dfrac{1}{r_{ji}}}{\sum\limits_{j\in N_i} \dfrac{1}{r_{ji}} + \left(P_0^{-1}\right)_{ii}} & i \neq j \; and \; i,j \; are \; neighbors \\ \dfrac{-\left(P_0^{-1}\right)_{ij}}{\sum\limits_{j\in N_i} \dfrac{1}{r_{ji}} + \left(P_0^{-1}\right)_{ii}} & otherwise \end{cases}$$

In order to show that the spectral radius of the iteration matrix is strictly smaller than 1, we will require that the matrix M is both non-negative and sub-stochastic (see Section 2.5). Let us find the conditions for the row sums of the matrix M to be smaller or equal to 1:

$$\sum_j M_{ij} = \frac{\sum\limits_{j\in N_i}\left[-\left(P_0^{-1}\right)_{ij} + \dfrac{1}{r_{ji}}\right] - \sum\limits_{\substack{j\notin N_i \\ j\neq i}}\left(P_0^{-1}\right)_{ij}}{\sum\limits_{j\in N_i}\dfrac{1}{r_{ji}} + \left(P_0^{-1}\right)_{ii}} =$$

$$= \frac{\sum\limits_{j\in N_i}\dfrac{1}{r_{ji}} - \sum\limits_{\forall j\neq i}\left(P_0^{-1}\right)_{ij}}{\sum\limits_{j\in N_i}\dfrac{1}{r_{ji}} + \left(P_0^{-1}\right)_{ii}} \leq 1$$

From this we get:

$$\sum_{j\in N_i}\frac{1}{r_{ji}}-\sum_{j\neq i}\left(P_0^{-1}\right)_{ij}\leq\sum_{j\in N_i}\frac{1}{r_{ji}}+\left(P_0^{-1}\right)_{ii}$$

$$\left(P_0^{-1}\right)_{ii}+\sum_{j\neq i}\left(P_0^{-1}\right)_{ij}\geq 0$$

This implies: $\boxed{\sum_{j}\left(P_0^{-1}\right)_{ij}\geq 0}$

In other words, we obtained that the necessary condition is that for each node Λ_i, the row sum of the matrix P_0^{-1} has to be non-negative. In addition, the condition: $\infty>\left(r_{ji}\right)^{-1}\geq 0\ \forall i,j$ ($r_{ji}=r_{ij}$) must hold too.

Requiring that all the entries of the matrix M are non-negative leads to:

$$\begin{cases}\left(P_0^{-1}\right)_{ii}\geq 0\\ \left(P_0^{-1}\right)_{ij}\leq 0\ \ (i\neq j)\end{cases}$$

Hence, we can write:

$$\left(P_0^{-1}\right)_{ii}\geq-\sum_{\forall j\neq i}\left(P_0^{-1}\right)_{ij}\geq 0$$

The above requirement can be seen as a diagonal dominance condition over the matrix P_0^{-1}. In the case that the node Λ_i is adjacent to the reference node, the corresponding row sum of the M matrix is given by:

$$\frac{\sum_{j\in N_i}\left[-\left(P_0^{-1}\right)_{ij}+\frac{1}{r_{ji}}\right]-\left[\frac{1}{r_{1i}}-\left(P_0^{-1}\right)_{i1}\right]+\sum_{\substack{j\notin N_i\\ j\neq i}}-\left(P_0^{-1}\right)_{ij}}{\sum_{j\in N_i}\frac{1}{r_{ji}}+\left(P_0^{-1}\right)_{ii}}=$$

$$=\frac{\left[\sum_{j\in N_i}\frac{1}{r_{ji}}-\sum_{\forall j\neq i}\left(P_0^{-1}\right)_{ij}\right]-\left[\frac{1}{r_{1i}}-\left(P_0^{-1}\right)_{i1}\right]}{\sum_{j\in N_i}\frac{1}{r_{ji}}+\left(P_0^{-1}\right)_{ii}}<1$$

In the case that the node Λ_i is not adjacent to the reference node, the corresponding row sum of the M matrix is given by:

$$\frac{\sum_{j\in N_i}\left[-\left(P_0^{-1}\right)_{ij}+\frac{1}{r_{ji}}\right]+\sum_{\substack{j\notin N_i\\ j\neq i}}-\left(P_0^{-1}\right)_{ij}-\left(P_0^{-1}\right)_{i1}}{\sum_{j\in N_i}\frac{1}{r_{ji}}+\left(P_0^{-1}\right)_{ii}}=$$

$$=\frac{\left[\sum_{j\in N_i}\frac{1}{r_{ji}}-\sum_{\forall j\neq i}\left(P_0^{-1}\right)_{ij}\right]-\left(P_0^{-1}\right)_{i1}}{\sum_{j\in N_i}\frac{1}{r_{ji}}+\left(P_0^{-1}\right)_{ii}}<1$$

Hence, we have shown that at least in one row, the row sum of M is strictly smaller than 1. Actually, we proved that the iteration matrix verifies all the sufficient conditions for convergence. Namely, the row sums of the matrix M are less or equal than 1 (and at least in one row this sum is strictly smaller than 1), the matrix M is irreducible and all its entries are non-negative. ■

As a result, we proved the convergence of the decentralized algorithm to the optimal solution performed by the centralized Kalman Filter for the most general case. Now, we will conclude the same for the other methods as special cases of the previous general framework. If the matrix P_0 is assumed to be diagonal and PSD, the decentralized algorithm converges. To see that, we have just to substitute $p_{ij}=0 \quad i\neq j$ and to check that the condition $\rho(M)<1$ is still verified . If $P_0=0$, we will obtain the Weighted-Least-Squares case, and it is easy to check that in this case too, the convergence of the decentralized algorithm is achieved. The proof for the most basic case (equivalent to LS) is provided in the literature by Giridhar et al. [12] but all the other cases are to the best of our knowledge original.

To sum up, the convergence conditions are given by:

$$\boxed{\begin{cases} R \text{ is diagonal and PSD, that is: } \infty>\left(r_{ji}\right)^{-1}\geq 0 \ \forall i,j \\ \sum_j\left(P_0^{-1}\right)_{ij}\geq 0 \\ \left(P_0^{-1}\right)_{ii}\geq 0 \\ \left(P_0^{-1}\right)_{ij}\leq 0 \ \left(i\neq j\right) \end{cases}}$$

We end this section by an additional lemma that will be of importance in the next chapter.

Lemma 6.1

The convergence conditions have to be checked only once at the beginning of the procedure. In other words, the Kalman Filter operations preserve the convergence properties, that is: if P_0 is an M-matrix, then P_n is an M-matrix for all $n \geq 1$.

Proof

In the information form of the discrete time Kalman Filter, the measurement update equation of the inverse covariance matrix is given by:

$$\left(P(n+1\,|\,n)\right)^{-1} = \left(P(n\,|\,n)\right)^{-1} + H^T R^{-1} H \qquad (6.5)$$

In our case $H = A^T$, hence we obtain: $\left(P(n+1\,|\,n)\right)^{-1} = \left(P(n\,|\,n)\right)^{-1} + AR^{-1}A^T$.

Recalling that the matrix R^{-1} is assumed to be diagonal, let us analyze the properties of the matrix $AR^{-1}A^T$. For the regular incidence matrix, we have:

$$AR^{-1}A^T \begin{pmatrix} 1 \\ \cdot \\ \cdot \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ \cdot \\ \cdot \\ 0 \end{pmatrix}$$

and for the reduced incidence matrix we have:

$$AR^{-1}A^T \begin{pmatrix} 1 \\ \cdot \\ \cdot \\ 1 \end{pmatrix} = \underline{v}$$

Here, $\underline{v}$ is a vector with non-negative components.

The structure of the matrix $AR^{-1}A^T$ is as follows:

$$\begin{cases} \left(AR^{-1}A^T\right)_{ii} > 0 \\ \left(AR^{-1}A^T\right)_{ij} \leq 0 \end{cases}$$

The row sums are $\sum_{\forall j} \left(AR^{-1}A^T\right)_{ij} = 0$ for each node Λ_i that is not adjacent to the reference node. Besides, if the node Λ_i is adjacent to the reference node, this sum is a strictly positive number. Hence, we can draw the conclusion that if the a-priori inverse covariance matrix $\left(P(n\,|\,n)\right)^{-1}$ verifies the convergence conditions, the a-posteriori inverse covariance matrix $\left(P(n+1\,|\,n)\right)^{-1}$ will verify them too. As a consequence, we have to check these conditions only once at the beginning of the procedure. This result will be useful in the next section. ■

In the next section, we extend the analysis to the case of multiple measurement sets and we propose an optimal decentralized recursive algorithm.

7. Recursive Algorithms: Multiple Measurement Update

In the previous sections, only the case of one measurement update was investigated. The next step is to consider the multiple measurement case and to present a recursive version of the previous decentralized algorithms. Several sets of measurements become successively available and our goal is to develop an algorithm that estimates the offsets after each set of measurements. The first alternative is to solve this problem by proposing a recursive algorithm. The latter is required to be decentralized and to depend only on the last measurement and on the previous estimates. The second option to solve the multiple measurement case is to wait for all the measurements and then to perform the estimation procedure. We will consider this case at the end of this section by proposing a decentralized non-recursive algorithm.

In the subsequent analysis, we still focus on the case where the initial covariance matrix P_0 is diagonal. The main problem is that after the first measurement update in the KF equations, the covariance matrix is not diagonal anymore. Then, each node has to communicate with all the other nodes over the network and not only with the one-hop neighbors. First, we present the centralized Kalman Filter algorithm and then we propose an optimal decentralized procedure that is equivalent to the KF solution. Afterward, we suggest a decentralized sub-optimal algorithm and a decentralized non-recursive method.

7.1. The Centralized KF Algorithm

In Section 5.2, we obtained that the centralized Kalman Filter optimal solution (using the LS approach) for a single set of measurements is given by:

$$\underline{\hat{x}} = \left(AR^{-1}A^T + P_0^{-1}\right)^{-1}\left(AR^{-1}y + P_0^{-1}\overline{x}_0\right)$$

We will now develop the corresponding recursive version given n sets of measurements. By repeating the previous derivation, we obtain:

$$\begin{cases} \underline{\hat{x}}(n) = \left(n\cdot AR^{-1}A^T + P_0^{-1}\right)^{-1}\left(\sum_{k=1}^{n} AR^{-1}y(k) + P_0^{-1}\overline{x}_0\right) \\ \underline{\hat{x}}(n-1) = \left((n-1)\cdot AR^{-1}A^T + P_0^{-1}\right)^{-1}\left(\sum_{k=1}^{n-1} AR^{-1}y(k) + P_0^{-1}\overline{x}_0\right) \end{cases}$$

This implies:

$$\boxed{\underline{\hat{x}}(n) = \left(n\cdot AR^{-1}A^T + P_0^{-1}\right)^{-1}\left[\left((n-1)\cdot AR^{-1}A^T + P_0^{-1}\right)\underline{\hat{x}}(n-1) + AR^{-1}y(n)\right]} \qquad (7.1.1)$$

Hence, we obtained a recursive relation for the estimated vector of offsets. The equation in (7.1.1) corresponds to the centralized version of the recursive algorithm. We can simplify this equation in the following way:

$$\underline{\hat{x}}(n) = \left(n \cdot AR^{-1}A^T + P_0^{-1}\right)^{-1}\left[\left((n-1)\cdot AR^{-1}A^T + P_0^{-1}\right)\underline{\hat{x}}(n-1) + AR^{-1}y(n)\right]$$

$$\underline{\hat{x}}(n) = \underline{\hat{x}}(n-1) + \left(n \cdot AR^{-1}A^T + P_0^{-1}\right)^{-1}\left[AR^{-1}y(n) - \left(AR^{-1}A^T\right)\underline{\hat{x}}(n-1)\right]$$

$$\underline{\hat{x}}(n) = \underline{\hat{x}}(n-1) + P(n\,|\,n)AR^{-1}\left[y(n) - A^T\underline{\hat{x}}(n-1)\right]$$

Let us now discuss several applications related to this centralized protocol. The centralized algorithm can be applied to a variety of situations. For example, one can imagine a set of fully intercommunicating nodes or the situation where a central unit is in charge. We can also apply the centralized protocol not solely on the entire network but also locally. Namely, we can estimate the offsets of a group of nodes that are regrouped geographically. For each group, we will assign a single processor that can communicate with all the group members. Another important application is the scenario in which a node needs to estimate simultaneously several different variables. For instance, in the sensor localization problem, each node Λ_i has to estimate his position by evaluating the three coordinate$\left(\hat{x}_i, \hat{y}_i, \hat{z}_i\right)$. In this case, each node needs to estimate a vector of several variables so it is worthwhile to extend our algorithm to the vector case (centralized version).

Similarly to the single measurement update, applying the centralized KF algorithm for networks with a large number of measurements, will be prohibitively expensive in terms of energy consumption, bandwidth and communication time. In other words, in order to compute the optimal estimate directly (in a centralized manner), one seems to need all the measurements associated with their error covariances and the topology of the entire network (because the covariance matrix $P(n\,|\,n)$ is non-diagonal). The solution we propose is a recursive decentralized algorithm based on the same LS approach as in the previous chapters.

7.2 The Optimal Decentralized Algorithm

We first consider the two measurements case and then extend to the general framework of n sets of measurements. We start with the initial parameters $\left(\overline{x}_0, P_0\right)$, where P_0 is chosen to be a diagonal matrix. When the first set of measurements, say $y(1)$, arrives we can estimate the offset vector $\hat{\tau}(1)$ and calculate the a-posteriori inverse covariance matrix $P(1)^{-1}$. This matrix will not be diagonal in general. The next step begins when the second set of measurements $y(2)$ arrives. As before, we can estimate the offset vector $\hat{\tau}(2)$ and calculate the a-posteriori inverse covariance matrix $P(2)^{-1}$. But here, as the matrix $P(1)^{-1}$ is not diagonal, the synchronization procedure is more difficult than the previous one because the Jacobi method does not lead to a distributed algorithm. During the second step, each node has to communicate with the entire network and not only with its neighbors. This was discussed in Section 5.2. On the other hand, we can wait that the two sets of measurement $\left(y(1), y(2)\right)$ arrive and then estimate the offset vector $\hat{\tau}(2)$ and calculate the a-posteriori inverse covariance matrix $P(2)^{-1}$ by applying the Kalman Filter equations to the combined measurement vector. This one step method will give the same result as the two steps method, but the procedure is easier because P_0 is diagonal and then the algorithm is decentralized. Our goal is to develop a scheme that combines the better features of both methods: namely, a recursive estimation algorithm that is still decentralized.

Let us solve the problem when two sets of measurements are available. Then, the objective function is given by:

$$J=(\underline{x}-\overline{x}_0)^T P_0^{-1}(\underline{x}-\overline{x}_0)+(y(1)-A^T\underline{x})^T R^{-1}(y(1)-A^T\underline{x})+(y(2)-A^T\underline{x})^T R^{-1}(y(2)-A^T\underline{x})$$

First, we compute the optimal offset estimate of node Λ_i given the first set of measurements. When the matrix P_0 is assumed to be diagonal and only one set of measurements is considered, we have obtained previously the following clock synchronization algorithm:

$$\hat{\tau}_i^{(k+1)}(1)=\frac{1}{\left(\sum_{j\in N_i}\frac{1}{r_{ji}}+\frac{1}{p_i}\right)}\cdot\left[\sum_{j\in N_i}\frac{1}{r_{ji}}\left(\hat{O}_{ji}^{(1)}+\hat{\tau}_j^{(k)}(1)\right)+\frac{\tau_i(0)}{p_i}\right];\quad i=2,3,\ldots N \qquad (7.2.1)$$

Initialization: $\hat{\tau}_i^{(0)}(1)=\tau_i(0)\quad i=2,3,\ldots N$. Here, $k\geq 0$ is the iteration number.

Equation (7.2.1) corresponds to a synchronous decentralized iterative algorithm. As we previously have shown, when the above procedure is applied by all nodes in all iterations, the set of offset estimates converges to the optimal centralized solution (if the matrix P_0^{-1} is chosen according to the conditions convergence). After repeating this algorithm an infinite number of iterations, we will obtain the optimal offset estimate $\hat{\tau}_i(1)$. In other words, $\hat{\tau}_i(1)$ is the optimal offset estimate (after the convergence of the above algorithm) of the node Λ_i given the set of measurements $y(1)$.

The next step is to compute the optimal offset estimate of node Λ_i given the sets of measurement $\left(y(1),y(2)\right)$. By repeating the one-measurement derivation, we obtain:

$$\hat{\tau}_i^{(k+1)}(2)=\frac{1}{2\sum_{j\in N_i}\frac{1}{r_{ji}}+\frac{1}{p_i}}\cdot\left[\sum_{j\in N_i}\frac{1}{r_{ji}}\left(\hat{O}_{ji}^{(1)}+\hat{\tau}_j^{(k)}(2)\right)+\frac{\tau_i(0)}{p_i}+\sum_{j\in N_i}\frac{1}{r_{ji}}\left(\hat{O}_{ji}^{(2)}+\hat{\tau}_j^{(k)}(2)\right)\right] \qquad (7.2.2)$$

Let us try to express $\hat{\tau}_i(2)$ as a function of $\hat{\tau}_i(1)$ (recursively). In equation (7.2.1), after an infinite number of iterations in all the nodes, we will get the following steady-state equations:

$$\hat{\tau}_i(1)=\frac{1}{\left(\sum_{j\in N_i}\frac{1}{r_{ji}}+\frac{1}{p_i}\right)}\cdot\left[\sum_{j\in N_i}\frac{1}{r_{ji}}\left(\hat{O}_{ji}^{(1)}+\hat{\tau}_j(1)\right)+\frac{\tau_i(0)}{p_i}\right]$$

Let us find an expression for $\sum_{j\in N_i}\frac{1}{r_{ji}}\hat{O}_{ji}^{(1)}$ from the above equation:

$$\sum_{j\in N_i}\frac{1}{r_{ji}}\hat{O}_{ji}^{(1)}=\left(\sum_{j\in N_i}\frac{1}{r_{ji}}+\frac{1}{p_i}\right)\cdot\hat{\tau}_i(1)-\sum_{j\in N_i}\frac{1}{r_{ji}}\hat{\tau}_j(1)-\frac{\tau_i(0)}{p_i}$$

Now, let us replace in equation (7.2.2):

$$\hat{\tau}_i^{(k+1)}(2)=\frac{1}{2\sum_{j\in N_i}\frac{1}{r_{ji}}+\frac{1}{p_i}}\cdot\{\left(\sum_{j\in N_i}\frac{1}{r_{ji}}+\frac{1}{p_i}\right)\cdot\hat{\tau}_i(1)-\sum_{j\in N_i}\frac{1}{r_{ji}}\hat{\tau}_j(1)-\cancel{\frac{\tau_i(0)}{p_i}}+$$
$$+\sum_{j\in N_i}\frac{1}{r_{ji}}\hat{\tau}_j^{(k)}(2)+\cancel{\frac{\tau_i(0)}{p_i}}+\sum_{j\in N_i}\frac{1}{r_{ji}}\left(\hat{O}_{ji}^{(2)}+\hat{\tau}_j^{(k)}(2)\right)\}$$

From this, we get:

$$\hat{\tau}_i^{(k+1)}(2)=\frac{1}{2\sum_{j\in N_i}\frac{1}{r_{ji}}+\frac{1}{p_i}}\cdot\left[\sum_{j\in N_i}\frac{1}{r_{ji}}\left(\hat{\tau}_i(1)-\hat{\tau}_j(1)\right)+\frac{\hat{\tau}_i(1)}{p_i}+\sum_{j\in N_i}\frac{1}{r_{ji}}\left(\hat{O}_{ji}^{(2)}+2\hat{\tau}_j^{(k)}(2)\right)\right]$$

which is the required expression.

Now, let us generalize this idea for n sets of measurements, i.e., to find a recursive relation between $\hat{\tau}_i^{(k+1)}(n)$ and $\hat{\tau}_i(n-1)$. We consider the general case in which the matrix R^{-1} may be different for each set of measurements. Hence, the objective function is given by:

$$J=(\underline{x}-\overline{x}_0)^T P_0^{-1}(\underline{x}-\overline{x}_0)+\sum_{k=1}^{n}(y(k)-A^T\underline{x})^T R(k)^{-1}(y(k)-A^T\underline{x}) \qquad (7.2.3)$$

First, differentiate the objective function (7.2.3) with respect to the offsets and set the partial derivatives to zero:

$$\frac{\partial J}{\partial\tau_i}=\left[\sum_{k=1}^{T}\sum_{j\in N_i}\frac{1}{r_{ji}(k)}\right]\cdot\tau_i-\sum_{k=1}^{n}\left[\sum_{j\in N_i}\frac{1}{r_{ji}(k)}\left(\hat{O}_{ji}^{(k)}+\tau_j\right)\right]+\left(P_0^{-1}\right)_{i*}\underline{x}-\left(P_0^{-1}\right)_{i*}\overline{x}_0=0$$

As usual, we assume that the initial covariance matrix P_0 is diagonal and obtain the corresponding iterative algorithm with combined measurements:

$$\hat{\tau}_i^{(k+1)}(n)=\frac{1}{\sum_{k=1}^{n}\left[\sum_{j\in N_i}\frac{1}{r_{ji}(k)}\right]+\frac{1}{p_i}}\left\{\sum_{k=1}^{n}\left[\sum_{j\in N_i}\frac{1}{r_{ji}(k)}\left(\hat{O}_{ji}^{(k)}+\hat{\tau}_j^{(k)}(n)\right)\right]+\frac{\tau_i(0)}{p_i}\right\}$$

Initialization: $\hat{\tau}_i^{(0)}(n)=\tau_i(0)\quad i=2,3,...N$. Here, $k\geq 0$ is the iteration number.

The above formula corresponds to a synchronous decentralized iterative algorithm. As we previously have shown, when the above procedure is applied by all nodes in all iterations, the set of offset estimates converges to the optimal centralized solution.

The same equation holds for $n-1$. After an infinite number of iterations in all the nodes, we will get the following steady-state equations:

$$\hat{\tau}_i(n-1)=\frac{1}{\sum_{k=1}^{n-1}\left[\sum_{j\in N_i}\frac{1}{r_{ji}(k)}\right]+\frac{1}{p_i}}\cdot\left\{\sum_{k=1}^{n-1}\left[\sum_{j\in N_i}\frac{1}{r_{ji}(k)}\left(\hat{O}_{ji}^{(k)}+\hat{\tau}_j(n-1)\right)\right]+\frac{\tau_i(0)}{p_i}\right\}$$

As before, let us find an expression for $\sum_{j\in N_i}\frac{1}{r_{ji}(n-1)}\hat{O}_{ji}{}^{(n-1)}$ from the above equation:

$$\sum_{j\in N_i}\frac{1}{r_{ji}(n-1)}\hat{O}_{ji}{}^{(n-1)}=\left[\sum_{k=1}^{n-1}\left[\sum_{j\in N_i}\frac{1}{r_{ji}(k)}\right]+\frac{1}{p_i}\right]\cdot\hat{\tau}_i(n-1)-\frac{\tau_i(0)}{p_i}-$$
$$\sum_{k=1}^{n-2}\left[\sum_{j\in N_i}\frac{1}{r_{ji}(k)}\left(\hat{O}_{ji}{}^{(k)}+\hat{\tau}_j(n-1)\right)\right]-\sum_{j\in N_i}\frac{1}{r_{ji}(n-1)}\hat{\tau}_j(n-1)$$

Now, replace the above expression in the equation of $\hat{\tau}_i^{(k+1)}(n)$:

$$\hat{\tau}_i^{(k+1)}(n)=\frac{1}{\sum_{k=1}^{n}\left[\sum_{j\in N_i}\frac{1}{r_{ji}(k)}\right]+\frac{1}{p_i}}\cdot\{\sum_{k=1}^{n-2}\left[\sum_{j\in N_i}\frac{1}{r_{ji}(k)}\left(\cancel{\hat{O}_{ji}^{(k)}}+\hat{\tau}_j{}^{(k)}(n)\right)\right]+\cancel{\frac{\tau_i(0)}{p_i}}+\sum_{j\in N_i}\frac{1}{r_{ji}(k)}\left(\hat{\tau}_j{}^{(k)}(n)\right)+$$
$$+\left[\sum_{k=1}^{n-1}\left(\sum_{j\in N_i}\frac{1}{r_{ji}(k)}\right)+\frac{1}{p_i}\right]\cdot\hat{\tau}_i(n-1)-\sum_{k=1}^{n-2}\left[\sum_{j\in N_i}\frac{1}{r_{ji}(k)}\left(\cancel{\hat{O}_{ji}^{(k)}}+\hat{\tau}_j(n-1)\right)\right]-$$
$$-\cancel{\frac{\tau_i(0)}{p_i}}-\sum_{j\in N_i}\frac{1}{r_{ji}(k)}\hat{\tau}_j(n-1)+\sum_{j\in N_i}\frac{1}{r_{ji}(k)}\left(\hat{O}_{ji}^{(n)}+\hat{\tau}_j{}^{(k)}(n)\right)\}$$

From this, we get:

$$\hat{\tau}_i^{(k+1)}(n)=\frac{1}{\sum_{k=1}^{n}\left[\sum_{j\in N_i}\frac{1}{r_{ji}(k)}\right]+\frac{1}{p_i}}\cdot\{\sum_{k=1}^{n-1}\left(\sum_{j\in N_i}\frac{1}{r_{ji}(k)}\left[\left(\hat{\tau}_i(n-1)-\hat{\tau}_j(n-1)\right)+\hat{\tau}_j{}^{(k)}(n)\right]\right)+$$
$$+\sum_{j\in N_i}\frac{1}{r_{ji}(n)}\left(\hat{O}_{ji}^{(n)}+\hat{\tau}_j{}^{(k)}(n)\right)+\frac{\hat{\tau}_i(n-1)}{p_i}\}$$

Now, let us continue to arrange the previous expression so that $\hat{\tau}_i^{(k+1)}(n)$ will be given by the sum of $\hat{\tau}_i(n-1)$ and an additional term related to the last set of measurements in order to obtain a recursive relation.

$$\hat{\tau}_i^{(k+1)}(n) = \hat{\tau}_i(n-1) + \frac{1}{\sum_{k=1}^{n}\left[\sum_{j\in N_i}\frac{1}{r_{ji}(k)}\right] + \frac{1}{p_i}}\{\sum_{j\in N_i}\frac{1}{r_{ji}(n)}\left[\hat{O}_{ji}^{(n)} - \left(\hat{\tau}_i(n-1) - \hat{\tau}_j^{(k)}(n)\right)\right] + \\ + \sum_{k=1}^{n-1}\left[\sum_{j\in N_i}\frac{1}{r_{ji}(k)}\left(\hat{\tau}_j^{(k)}(n) - \hat{\tau}_j(n-1)\right)\right]\}$$

$$\hat{\tau}_i^{(k+1)}(n) = \hat{\tau}_i(n-1) + \frac{1}{\sum_{j\in N_i}\left[\sum_{k=1}^{n}\frac{1}{r_{ji}(k)}\right] + \frac{1}{p_i}}\cdot\{\sum_{j\in N_i}\frac{1}{r_{ji}(T)}\left[\hat{O}_{ji}^{(n)} - \underbrace{\left(\hat{\tau}_i(n-1) - \hat{\tau}_j(n-1)\right)}_{\text{the estimated measurement}}\right] + \\ + \sum_{j\in N_i}\left[\sum_{k=1}^{n}\frac{1}{r_{ji}(k)}\right]\left(\hat{\tau}_j^{(k)}(n) - \hat{\tau}_j(n-1)\right)\}$$

For the special case where the matrix R^{-1} is similar for each set of measurements, we obtain:

$$\boxed{\hat{\tau}_i^{(k+1)}(n) = \hat{\tau}_i(n-1) + \frac{1}{n\sum_{j\in N_i}\frac{1}{r_{ji}} + \frac{1}{p_i}}\cdot\left\{\sum_{j\in N_i}\frac{1}{r_{ji}}\left[\hat{O}_{ji}^{(n)} - \underbrace{\left(\hat{\tau}_i(n-1) - \hat{\tau}_j(n-1)\right)}_{\text{the estimated measurement}} + n\left(\hat{\tau}_j^{(k)}(n) - \hat{\tau}_j(n-1)\right)\right]\right\}}$$

The above algorithm is decentralized, recursive and iterative. For each set of measurements, it performs an infinite number of iterations in order to converge to the optimal solution. Moreover, $\hat{\tau}_i^{(k+1)}(n)$ depends only on the last measurement and on the previous estimates. We point out that the last equation slightly deviates from the standard structure of a recursive algorithm due to the term n (time explicit index) in the denominator and in the internal term.

Let us define the following quantity:

$$\left[I_i(n)\right]^{-1} = \frac{1}{p_i} + \sum_{j\in N_i}\left[\sum_{k=1}^{n}\frac{1}{r_{ji}(k)}\right]$$

Hence, the following recursive relation holds:

$$\begin{cases}\left[I_i(n)\right]^{-1} = \left[I_i(n-1)\right]^{-1} + \sum_{j\in N_i}\frac{1}{r_{ji}(n)} \\ \left[I_i(0)\right]^{-1} = \left[P_i(0)\right]^{-1} = \frac{1}{p_i}\end{cases} \qquad (7.2.4)$$

We note that the elements of $\left[I_i(n)\right]^{-1}$ in (7.2.4) are the diagonal entries of the inverse covariance matrix in the Kalman Filter equations. Thus, the error estimation variances of the estimates at each step are obtained. This is a desirable property since it gives information on the estimation quality. Observe however that we do not compute the non-diagonal elements of the inverse covariance matrix.

Using this notation, the recursive version of the decentralized algorithm in its final form is given by:

$$\boxed{\begin{array}{l} \hat{\tau}_i^{(k+1)}(n) = \hat{\tau}_i(n-1) + \dfrac{1}{\left[I_i(n)\right]^{-1}} \cdot \left\{ \sum\limits_{j \in N_i} \dfrac{1}{r_{ji}} \cdot \left[\hat{O}_{ji}^{(n)} - \left(\hat{\tau}_i(n-1) - \hat{\tau}_j^{(k)}(n) \right) \right] + (n-1) \cdot \left[\sum\limits_{j \in N_i} \dfrac{1}{r_{ji}} \cdot \left(\hat{\tau}_j^{(k)}(n) - \hat{\tau}_j(n-1) \right) \right] \right\} \\ \left[I_i(n)\right]^{-1} = \left[I_i(n-1)\right]^{-1} + \sum\limits_{j \in N_i} \dfrac{1}{r_{ji}} = \dfrac{1}{p_i} + n \cdot \sum\limits_{j \in N_i} \dfrac{1}{r_{ji}} \qquad i = 2, \dots, N \end{array}} \tag{7.2.5}$$

where: $\left[I_i(0)\right]^{-1} = \left[P_i(0)\right]^{-1} = \dfrac{1}{p_i}$.

We consider the case where the matrix R^{-1} is similar for each set of measurements for notation simplicity. Moreover, this assumption is quite logical and this scenario can be considered as the most representative case. The above set of equations in (7.2.5) correspond to our main algorithm that is summarized in Table 7.1. It is a decentralized, synchronous and recursive algorithm that computes at each step, the estimated offsets in addition to the corresponding error variances. The main advantage of this algorithm is its local nature; each network node needs to communicate only with its neighbors.

We now describe in words the iterative procedure in (7.2.5). At time n, we assume that the estimate of $\hat{\tau}_i(n-1)$ is given. Then, $\hat{\tau}_i^{(k)}(n)$ $k = 1, 2, \dots$ is computed based on $\hat{\tau}_i(n-1)$ and the last measurement set $y(n)$. We assume that a sufficient number of iterations is performed at each time n, so that the estimate $\hat{\tau}_i(n)$ is accurate.

The proposed algorithm in (7.2.5) may be derived in two ways, which lead to the same optimal equations:

1. Differentiate $J(n-1)$ and $J(n)$ with respect to the offsets vector x and set the partial derivatives to zero.
2. Algebraic manipulations of the standard recursive extension of (5.3.4), with the following KF update inverse covariance equation:

$$\left(P_{k+1}\right)^{-1} = \left(P_k\right)^{-1} + AR^{-1}A^T \tag{7.2.6}$$

In other words, the set of iterative equations in (7.2.5) is mathematically equivalent to perform (5.3.4) separately for each measurement set in addition to (7.2.6). This result will be useful in the proof of Theorem 7.1. This can be shown easily by some appropriate mathematical manipulations and is not presented.
Now, let us show the convergence of the set of equations in (7.2.5) to the optimal centralized solution.

Theorem 7.1.

Suppose that:

a) The matrix R is diagonal and Positive Semi-Definite, that is: $0 \le \left(r_{ji}\right)^{-1} < \infty \ \forall i, j$.

b) The initial covariance matrix P_0 is an M-matrix, namely:

$$\begin{cases} \sum_j \left(P_0^{-1}\right)_{ij} \ge 0 \\ \left(P_0^{-1}\right)_{ii} \ge 0 \ \ and \ \left(P_0^{-1}\right)_{ij} \le 0 \ \left(i \ne j\right) \end{cases}$$

c) The clock adjustment operation in (7.2.5) is applied synchronously by all nodes ($i = 2,3,...N$) in all iterations, recursively for n sets of measurements .

d) A sufficient number of iterations is performed after each measurement set n, so that $\hat{\tau}_i^{(k)}(n)$ converges to $\hat{\tau}_i(n)$.

Then, for each $n \ge 1$, the iterated estimators $\hat{\tau}_i^{(k)}(n) \ \ i = 2,3,...N$ *converge (as $k \to \infty$) to the optimal offsets that minimize the objective function in (7.2.3).*

We next present the proof of Theorem 7.1. Our proof relies on Lemma 6.1.

Lemma 6.1 immediately implies the convergence of the recursive extension (for several measurement sets) of equation (5.3.4) to the optimal solution, where at each step, the new covariance matrix is computed according to (7.2.6). Since the iterations in (7.2.5) are equivalent to the procedure in (5.3.4), we obtain the claimed convergence in Theorem 7.1.■

Remarks:

- We can also apply the recursive procedure to the case where at each stage, just some of the pair-wise offsets are measured (e.g., $y(1)$ is only the measurement of $\hat{O}_{34}$). In this case, the matrix A^T for that stage will have a structure in accordance with the measurements.

- The main advantage of this recursive method is that we have the same diagonal matrix P_0^{-1} during the whole procedure, thus it is easier to prove convergence. Moreover, we do not need that each node communicates with all the other nodes but only with its neighbors. In fact, when a new set of measurement arrives, we exploit the estimate of the previous iteration and add only the information contained in the new measurement.

```
Name: Decentralized Synchronous Recursive Kalman Filter
Assumptions: - The inverse initial covariance matrix $P_0^{-1}$ is diagonal.
             - $n$ sets of measurements are available.
             - The matrix $R^{-1}$ is the same for each set of measurements.
Goal: Compute in a decentralized manner the offsets estimates at each network node
      (except for the reference) that approach the optimal centralized estimates.
Initialization: $\tau_1^{(k)} = 0 \ \forall k$, $\hat{\tau}_i^{(0)}$ is arbitrary for $i \in V \setminus \{1\}$.

After deployment, each node $i \in V \setminus \{1\}$ performs:
    6. Detect its neighbors $N_i$.
    7. Identify the inverse initial covariance $\frac{1}{p_i}$ and the initial offset $\tau_i(0)$.
    8. Obtain the first set of relative measurements $\hat{O}_{ji}^{(1)}$ and the associated inverse
       covariances $\frac{1}{r_{ji}}$ for every $j \in N_i$. Compute $[I_i(1)]^{-1} = \sum_{j \in N_i} \frac{1}{r_{ji}} + \frac{1}{p_i}$.
    9. Send $\tau_i(0)$ to its neighbors $j \in N_i$. Obtain $\tau_j(0)$ $j \in N_i$ and keep in memory $\tau_i(0)$.

At every time $n$ that a new set of measurement arrives:
    10. Compute recursively $[I_i(n)]^{-1} = [I_i(n-1)]^{-1} + \sum_{j \in N_i} \frac{1}{r_{ji}}$ and send $n$ to every node.
    11. At every iteration $k$, each node $\Lambda_i$ performs:
        a. Send $\hat{\tau}_i^{(k)}(n)$ and $k$ to its neighbors $j \in N_i$. Obtain $\hat{\tau}_j^{(k)}(n)$ $j \in N_i$.
        b. Compute $\hat{\tau}_i^{(k+1)}(n)$ from the previous quantities, using (7.2.5).
    12. Send the final values of $\hat{\tau}_i(n)$ to its neighbors $j \in N_i$. Obtain $\hat{\tau}_j(n)$ $j \in N_i$ and
        keep in memory $\hat{\tau}_i(n)$.
```

Table 7.1. Summary of the Decentralized Synchronous Recursive KF Algorithm.

So far, we developed the recursive decentralized algorithm for multiple sets of measurements using the Least-Squares approach. Next, we will show that one can obtain the same algorithm through the Kalman Filter equations.

7.3 Equivalence with the KF Equations

We will show that the previous recursive algorithm can be obtained by applying the Kalman Filter equations through appropriate manipulations. By applying the information form of the Kalman Filter in our context, we obtained the following recursive equation (see in Section 5.1):

$$\underline{\hat{x}}(n\,|\,n) = \underline{\hat{x}}(n-1\,|\,n-1) + \left[\left(P(n-1\,|\,n-1)+Q\right)^{-1} + AR^{-1}A^T\right]^{-1} AR^{-1}\left[y(n) - A^T\underline{\hat{x}}(n-1\,|\,n-1)\right]$$

Now, we will develop a decentralized version of the above equation and as expected, we will show that this approach leads to the same recursive algorithm that we obtained in the last section.

Let us recall that $\underline{x}(n) \doteq \left(\tau_1 = 0, \tau_2, \ldots \tau_N\right)^T$ (where τ_i is the offset of the node Λ_i) and that $\left[P_i(0)\right]^{-1} = \frac{1}{p_i}$ (the initial inverse covariance matrix is diagonal). Moreover, we assume as usual that the matrix R is diagonal. For the case where $Q = 0$ (there is no process noise), due to the special structure of the matrices $AR^{-1}A^T$ and AR^{-1}, one can write the previous equation for each node separately and obtain the following decentralized estimates:

$$\hat{\tau}_i(n) = \hat{\tau}_i(n-1) + \frac{1}{\frac{1}{p_i} + n\cdot\sum_{j\in N_i}\frac{1}{r_{ji}}}\cdot\left\{\sum_{j\in N_i}\frac{1}{r_{ji}}\cdot\left[\hat{O}_{ji}^{(n)} - \hat{\tau}_i(n-1)\right] + n\cdot\left[\sum_{j\in N_i}\frac{1}{r_{ji}}\cdot\left(\hat{\tau}_j(n) - \hat{\tau}_j(n-1)\right)\right]\right\} \quad i = 2,\ldots,N$$

As before, we can define:

$$\begin{cases} \left[I_i(n)\right]^{-1} = \left[I_i(n-1)\right]^{-1} + \sum_{j\in N_i}\frac{1}{r_{ji}} = \frac{1}{p_i} + n\cdot\sum_{j\in N_i}\frac{1}{r_{ji}} \\ \left[I_i(0)\right]^{-1} = \left[P_i(0)\right]^{-1} = \frac{1}{p_i} \end{cases}$$

and then the recursive algorithm is given by:

$$\boxed{\begin{array}{l} \hat{\tau}_i^{(k+1)}(n) = \hat{\tau}_i(n-1) + \frac{1}{\left[I_i(n)\right]^{-1}}\cdot\left\{\sum_{j\in N_i}\frac{1}{r_{ji}}\cdot\left[\hat{O}_{ji}^{(n)} - \left(\hat{\tau}_i(n-1) - \hat{\tau}_j^{(k)}(n)\right)\right] + (n-1)\cdot\left[\sum_{j\in N_i}\frac{1}{r_{ji}}\cdot\left(\hat{\tau}_j^{(k)}(n) - \hat{\tau}_j(n-1)\right)\right]\right\} \\ \left[I_i(n)\right]^{-1} = \left[I_i(n-1)\right]^{-1} + \sum_{j\in N_i}\frac{1}{r_{ji}} = \frac{1}{p_i} + n\cdot\sum_{j\in N_i}\frac{1}{r_{ji}} \qquad i = 2,\ldots,N \end{array}}$$

where: $\left[I_i(0)\right]^{-1} = \left[P_i(0)\right]^{-1} = \frac{1}{p_i}$.

This is an iterative (synchronous) algorithm that computes recursively the estimated offset for each node Λ_i ($k \geq 0$ is the iteration number) in a decentralized manner. The above algorithm is exactly similar to the procedure that we obtained in (7.2.5).

In summary, we have shown as expected that the decentralized recursive algorithm we developed is exactly equivalent to the Kalman Filter. Indeed, we solve the problem of finding an optimal decentralized algorithm by two different approaches that lead to the same optimal solution. This is not a surprise because in the different derivations we are looking for the linear optimal estimate in the MMSE sense and the optimal solution is given by the Kalman Filter. The special structure of the centralized version of the Kalman Filter equations enables us to derive a decentralized version just with some mathematical manipulations. This is not always possible, as we will see for the case where a white noise process is incorporated in the equations. The problem of the Kalman Filter equations was the fact that the covariance matrix does not keep its diagonal initial structure. Hence, each node has to communicate with the entire network (or equivalently to make use of a central unit). In the decentralized version, the requirement is that each node communicates only with its one-hop neighbors. In addition, the terms $\left[I_i(n)\right]^{-1}$ are equal to the diagonal terms of the inverse covariance matrix. In other words, this algorithm computes the variance of each estimate, so we can know the quality of the estimation at each node.

Next, we propose a simple sub-optimal algorithm that computes the offsets for the case where multiple sets of measurements are available.

7.4 A Sub-Optimal Decentralized Algorithm

Another approach to solving the decentralized estimation problem can be considered. At the end of the section 5, we obtained a general decentralized equation in the case of a non-diagonal initial covariance matrix (for a single set of measurements):

$$\hat{\tau}_i^{(k+1)} = \frac{1}{\left(\sum_{j\in N_i} \frac{1}{r_{ji}} + \left(P_0^{-1}\right)_{ii}\right)} \left[\sum_{j\in N_i} \frac{1}{r_{ji}} \left(\hat{O}_{ji} + \hat{\tau}_j^{(k)}\right) + \left(P_0^{-1}\right)_{ii} \tau_i(0) - \sum_{\substack{m=1\\ m\neq i}}^{N} \left(P_0^{-1}\right)_{im} \left(\hat{\tau}_m^{(k)} - \tau_m(0)\right) \right] \quad (7.4.1)$$

As expected, the estimated offset of node Λ_i depends on all the other offsets and not only those of its neighbors. One can think about the naïve sub-optimal algorithm that neglects the off-diagonal terms of the inverse covariance matrix:

$$\hat{\tau}_i^{(k+1)} = \frac{1}{\left(\sum_{j\in N_i} \frac{1}{r_{ji}} + \left(P_0^{-1}\right)_{ii}\right)} \left[\sum_{j\in N_i} \frac{1}{r_{ji}} \left(\hat{O}_{ji} + \hat{\tau}_j^{(k)}\right) + \left(P_0^{-1}\right)_{ii} \tau_i(0) \right]$$

We obtained a decentralized sub-optimal algorithm for the case where a single set of measurements is available. Let us generalize for the multiple measurement scenario. In this case, the centralized optimal algorithm is given by:

$$\underline{\hat{x}}(n\,|\,n) = \underline{\hat{x}}(n-1\,|\,n-1) + \left[P^{-1}(n-1\,|\,n-1) + AR^{-1}A^T\right]^{-1} AR^{-1}\left[y(n) - A^T\underline{\hat{x}}(n-1\,|\,n-1)\right]$$

If we neglect the off-diagonal terms of the inverse covariance matrix at time $n-1$, we have (as usual, we assumed that the initial covariance matrix is diagonal):

$$\underline{\hat{x}}(n\,|\,n) = \underline{\hat{x}}(n-1\,|\,n-1) + \left[diag\left(P^{-1}(n-1\,|\,n-1)\right) + AR^{-1}A^T\right]^{-1} AR^{-1}\left[y(n) - A^T\underline{\hat{x}}(n-1\,|\,n-1)\right]$$

$$diag\left(P^{-1}(n-1\,|\,n-1)\right) = \left(P_0^{-1}\right)_{ii} + (n-1)\cdot\sum_{j\in N_i}\frac{1}{r_{ji}} = \frac{1}{p_i} + (n-1)\cdot\sum_{j\in N_i}\frac{1}{r_{ji}} = \left[I_i(n-1)\right]^{-1}$$

Here, we neglect the off-diagonal terms before inverting the information matrix $\left(P_{n-1}\right)^{-1}$, in the goal to improve the complexity. In this case, we will invert a diagonal matrix and hence the time computation will significantly decrease.

Therefore, the decentralized sub-optimal algorithm is given by:

$$\hat{\tau}_i^{(k+1)}(n) = \hat{\tau}_i(n-1) + \frac{1}{\left(P_0^{-1}\right)_{ii} + n \cdot \sum_{j \in N_i} \frac{1}{r_{ji}}} \left\{ \sum_{j \in N_i} \frac{1}{r_{ji}} \left[\hat{O}_{ji}^{(n)} - \left(\hat{\tau}_i(n-1) - \hat{\tau}_j^{(k)}(n) \right) \right] \right\} \qquad (7.4.2)$$

We can interpret equation (7.4.2) in a very logical manner. The new estimate is given by the sum of the previous estimate and a correction term. This correction term is composed of the latest measurement minus the estimated measurement multiplied by the measurement variance and the total is normalized by the accumulative variance.

In the numerical results section, we will compare the decentralized recursive algorithm that converges to the optimal solution to the above sub-optimal scheme. The sub-optimal algorithm is summarized in Table 7.2.

Name: Decentralized (recursive) Sub-Optimal Algorithm
Assumptions: - The inverse initial covariance matrix P_0^{-1} is diagonal.
- n sets of measurements are available.
- The matrix R^{-1} is the same for each set of measurements.

Goal: Compute in a decentralized manner the offsets estimates at each network node (except for the reference) in a logical sub-optimal manner.
Initialization: $\tau_1^{(k)} = 0 \ \forall k$, $\hat{\tau}_i^{(0)}$ is arbitrary for $i \in V \setminus \{1\}$.

After deployment, each node $i \in V \setminus \{1\}$ performs:

1. Detect its neighbors N_i.
2. Identify the inverse initial covariance $\frac{1}{p_i}$ and the initial offset $\tau_i(0)$.
3. Obtain the first set of relative measurements $\hat{O}_{ji}^{(1)}$ and the associated inverse covariances $\frac{1}{r_{ji}}$ for every $j \in N_i$. Compute $\left[I_i(1)\right]^{-1} = \sum_{j \in N_i} \frac{1}{r_{ji}} + \frac{1}{p_i}$.
4. Send $\tau_i(0)$ to its neighbors $j \in N_i$. Obtain $\tau_j(0)$ $j \in N_i$ and keep in memory $\tau_i(0)$.

At every time n that a new set of measurement arrives:

5. Compute recursively $\left[I_i(n)\right]^{-1} = \left[I_i(n-1)\right]^{-1} + \sum_{j \in N_i} \frac{1}{r_{ji}}$.
6. At every iteration k, each node Λ_i performs:
   a. Send $\hat{\tau}_i^{(k)}(n)$ and k to its neighbors $j \in N_i$. Obtain $\hat{\tau}_j^{(k)}(n)$ $j \in N_i$.
   b. Compute $\hat{\tau}_i^{(k+1)}(n)$ from the previous quantities, using (7.4.2).
7. Send the final values of $\hat{\tau}_i(n)$ to its neighbors $j \in N_i$. Obtain $\hat{\tau}_j(n)$ $j \in N_i$ and keep in memory $\hat{\tau}_i(n)$.

Table 7.2. Summary of the Decentralized Sub-Optimal Algorithm.

Next, we will present a decentralized non-recursive algorithm that solves the time synchronization problem in the multiple measurement update case.

7.5 Decentralized Non-Recursive Algorithm

Let us present an alternative method for the estimation algorithm in the multiple measurement case. The objective function is the same as before with the usual assumptions. Namely, the matrices P_0 and R are assumed to be diagonal and we suppose that the matrix R^{-1} is similar for all the measurement sets.

$$J = (\underline{x} - \overline{x}_0)^T P_0^{-1} (\underline{x} - \overline{x}_0) + \sum_{k=1}^{n} (y(k) - A^T \underline{x})^T R^{-1} (y(k) - A^T \underline{x})$$

In Section 5, we have obtained the following synchronous decentralized iterative clock synchronization algorithm (for a single set of measurements):

$$\hat{\tau}_i^{(k+1)} = \frac{1}{\left(\sum_{j \in N_i} \frac{1}{r_{ji}} + \frac{1}{p_i}\right)} \cdot \left[\sum_{j \in N_i} \frac{1}{r_{ji}} \left(\hat{O}_{ji}^{(1)} + \hat{\tau}_j^{(k)}\right) + \frac{\tau_i(0)}{p_i}\right] \qquad i = 2,3,...N \qquad (7.5.1)$$

Initialization: $\hat{\tau}_i^{(0)} = \tau_i(0) \quad i = 2,3,...N$. Here, $k \geq 0$ is the iteration number.

Our goal is to develop a new non-recursive estimation method (batch algorithm) for the multiple measurement case. In the latter, we will define the equivalent measurement and the corresponding equivalent covariance. In fact, we wait for all the measurement sets and then, we compute the equivalent measurement and its corresponding covariance as follows (we recall that the different measurement sets have independent noises):

$$\begin{cases} \widehat{\hat{O}_{ji}} = \dfrac{\sum_{k=1}^{n} \frac{1}{r_{ji}(k)} \hat{O}_{ji}^{(k)}}{\sum_{k=1}^{n} \frac{1}{r_{ji}(k)}} = \dfrac{1}{n} \sum_{k=1}^{n} \hat{O}_{ji}^{(k)} \\ \dfrac{1}{\widehat{r_{ji}}} = \sum_{k=1}^{n} \dfrac{1}{r_{ji}(k)} = n \dfrac{1}{r_{ji}} \end{cases}$$

Now, we can consider that we have just a single set of measurements and we can use the decentralized clock synchronization procedure in (7.5.1).

Let us replace the term: $\sum_{j \in N_i} \frac{1}{r_{ji}} \hat{O}_{ji}^{(1)}$ by the above equivalent expressions and then we obtain:

$$\hat{\tau}_i^{(k+1)} = \frac{1}{\left(\sum_{j \in N_i} \frac{1}{\widehat{r_{ji}}} + \frac{1}{p_i}\right)} \cdot \left[\sum_{j \in N_i} \frac{1}{\widehat{r_{ji}}} \widehat{\hat{O}_{ji}} + \sum_{j \in N_i} \frac{1}{\widehat{r_{ji}}} \hat{\tau}_j^{(k)} + \frac{\tau_i(0)}{p_i}\right]$$

$$\boxed{\hat{\tau}_i^{(k+1)} = \frac{1}{\left(n \sum_{j \in N_i} \frac{1}{r_{ji}} + \frac{1}{p_i}\right)} \cdot \left[\sum_{j \in N_i} \frac{1}{r_{ji}} \left(\sum_{k=1}^{n} \hat{O}_{ji}^{(k)} + n \cdot \hat{\tau}_j^{(k)}\right) + \frac{\tau_i(0)}{p_i}\right] \qquad i = 2,3,...N} \qquad (7.5.2)$$

The equation given in (7.5.2) is an additional decentralized algorithm for estimating the offset at each network node with respect to the reference time, using a Kalman Filter framework. In fact, instead of using the first measurement in the synchronization procedure, we employ the sum of all the measurements normalized by the number of sets. One can also extend the previous procedure to the case in which the matrix R^{-1} is different for each set of measurements. Actually, the equivalent measurement is given by the weighting average of all the measurements pre-multiplied by their variances.

In summary, we have developed several algorithms in order to compute the optimal estimated offsets in a network, including in the multiple measurement case. The first option is to apply the centralized Kalman Filter equations but in this case, each node is required to communicate with every other node and not only with its neighbors. The second option is to wait for all the sets of measurements and to apply the non-recursive algorithm we have presented in the last section. In this case, the estimates are optimal and the communication is only between neighbors but it cannot be implemented in real-time applications. The third possibility is the algorithm that we developed; it requires only local communication and gives an on-line optimal solution but we need to know the parameter n (the number of measurement sets) and to run an iterative procedure. The fourth and last option is to apply the sub-optimal algorithm (by neglecting the off-diagonal terms of the inverse covariance matrix). This solution is the simplest in the sense that it does not require an iterative algorithm and require local communication however, the solution is only sub-optimal.

In the next section, we will extend our results to several interesting directions: the incorporation of a discount factor (in order to compensate for the time-invariance assumption), the addition of a process noise to the state space equations and the extension to temporary communication failures environment.

8. Extensions

8.1 Incorporating a Discount Factor

In this case, the objective function to be minimized is given by (n represents the number of measurement sets):

$$J_n = \gamma^n (\underline{x} - \overline{x}_0)^T P_0^{-1} (\underline{x} - \overline{x}_0) + \sum_{k=1}^{n} \left[\gamma^{n-k} (y(k) - A^T \underline{x})^T R^{-1} (y(k) - A^T \underline{x}) \right] \xrightarrow{\min} \underline{\hat{x}}_{opt}(n) \qquad (8.1.1)$$

Here, $0 < \gamma < 1$ is the discount factor that gives a higher weight to the more recent measurements. In other words, this factor can compensate for the assumption that the offsets are time-invariant. An additional point of view consists of incorporating the discount factor to the state space model or to the measurement noise covariance matrix. The corresponding state space model in this case is given by:

$$\begin{cases} \underline{x}(n+1) = \underline{x}(n) \\ y(n) = A^T \underline{x}(n) + \gamma^{\frac{n}{2}} \underline{v}(n) \end{cases} \qquad \underline{v}(n) \sim N(0, R)$$

The second alternative is to consider the standard state space model and to incorporate the discount factor in the measurement noise covariance matrix. In this case:

$$\underline{v}(n) \sim N(0, R_n)$$
$$R_n = R \cdot \gamma^n$$

In other words, the measurement noise decreases in time.

Our goal is to find the optimal offsets. Hence, we compute the derivative of the objective function with respect to the offsets vector and set it to zero. As usual, we assume that the initial inverse covariance matrix P_0^{-1} is diagonal.

$$\frac{\partial J}{\partial x_i} = \sum_{k=1}^{n} \gamma^{n-k} \left[\left(AR^{-1}A^T\right)_i \underline{x} - \left(AR^{-1}\right)_i y(k) \right] + \gamma^n \left[\left(P^{-1}\right)_i \underline{x} - \left(P^{-1}\right)_i \overline{x}_0 \right] = 0 \qquad i = 2,3,...N$$

$$\sum_{k=1}^{n} \gamma^{n-k} \left[\sum_{j \in N_i} \frac{1}{r_{ji}} \left(\tau_i - \tau_j\right) - \sum_{j \in N_i} \frac{1}{r_{ji}} \hat{O}_{ji}^{(k)} \right] + \gamma^n \left[\frac{1}{p_i} \tau_i - \frac{1}{p_i} \tau_i(0) \right] = 0$$

$$\tau_i \left[\sum_{k=1}^{n} \gamma^{n-k} \cdot \sum_{j \in N_i} \frac{1}{r_{ji}} + \gamma^n \cdot \frac{1}{p_i} \right] = \sum_{k=1}^{n} \left[\gamma^{n-k} \sum_{j \in N_i} \frac{1}{r_{ji}} \left(\hat{O}_{ji}^{(k)} + \tau_j \right) \right] + \gamma^n \cdot \frac{1}{p_i} \tau_i(0) = 0$$

This implies:

$$\boxed{\tau_i(n) = \frac{1}{\left(\sum_{j \in N_i} \frac{1}{r_{ji}} \cdot \sum_{k=1}^{n} \gamma^{n-k} + \gamma^n \cdot \frac{1}{p_i} \right)} \left\{ \sum_{k=1}^{n} \left[\gamma^{n-k} \sum_{j \in N_i} \frac{1}{r_{ji}} \left(\hat{O}_{ji}^{(k)} + \tau_j(n) \right) \right] + \gamma^n \cdot \frac{1}{p_i} \tau_i(0) \right\} \qquad i = 2,3,...N} \qquad (8.1.2)$$

Equation (8.1.2) represents the optimal offset of node Λ_i given n sets of measurements. We notice that the case $\gamma = 1$ coincides with the case we studied in the previous section (without discount factor). The above equation can be implemented by a synchronous iterative algorithm, as in the previous cases:

$$\hat{\tau}_i^{(k+1)}(n) = \frac{1}{\left(\sum_{j \in N_i} \frac{1}{r_{ji}} \cdot \sum_{m=1}^{n} \gamma^{n-m} + \gamma^n \cdot \frac{1}{p_i} \right)} \left\{ \sum_{m=1}^{n} \left[\gamma^{n-m} \sum_{j \in N_i} \frac{1}{r_{ji}} \left(\hat{O}_{ji}^{(m)} + \hat{\tau}_j^{(k)}(n) \right) \right] + \gamma^n \cdot \frac{1}{p_i} \tau_i(0) \right\} \quad (8.1.3)$$

Next, we will develop a recursive version of this synchronous iterative algorithm in a similar way to the basic case.

The synchronous iterative algorithm that computes the optimal offset at node Λ_i given $n-1$ sets of measurements is given by:

$$\hat{\tau}_i^{(k+1)}(n-1) = \frac{1}{\left(\sum_{j \in N_i} \frac{1}{r_{ji}} \cdot \sum_{m=1}^{n-1} \gamma^{n-1-m} + \gamma^{n-1} \cdot \frac{1}{p_i} \right)} \left\{ \sum_{m=1}^{n-1} \left[\gamma^{n-1-m} \sum_{j \in N_i} \frac{1}{r_{ji}} \left(\hat{O}_{ji}^{(m)} + \hat{\tau}_j^{(k)}(n-1) \right) \right] + \gamma^{n-1} \cdot \frac{1}{p_i} \tau_i(0) \right\}$$

After an infinite number of iterations in all the nodes, we will obtain the following steady-state equations:

$$\hat{\tau}_i(n-1) = \frac{1}{\left(\sum_{j \in N_i} \frac{1}{r_{ji}} \cdot \sum_{m=1}^{n-1} \gamma^{n-1-m} + \gamma^{n-1} \cdot \frac{1}{p_i} \right)} \left\{ \sum_{m=1}^{n-1} \left[\gamma^{n-1-m} \sum_{j \in N_i} \frac{1}{r_{ji}} \left(\hat{O}_{ji}^{(m)} + \tau_j(n-1) \right) \right] + \gamma^{n-1} \cdot \frac{1}{p_i} \tau_i(0) \right\}$$

Let us find the expression for $\sum_{j \in N_i} \frac{1}{r_{ji}} \hat{O}_{ji}^{(n-1)}$ from the above equation:

$$\sum_{j \in N_i} \frac{1}{r_{ji}} \hat{O}_{ji}^{(n-1)} = \left[\sum_{j \in N_i} \frac{1}{r_{ji}} \cdot \sum_{m=1}^{n-1} \gamma^{n-1-m} + \frac{1}{p_i} \cdot \gamma^{n-1} \right] \cdot \hat{\tau}_i(n-1) - \gamma^{n-1} \cdot \frac{\tau_i(0)}{p_i} -$$
$$- \sum_{m=1}^{n-2} \left[\gamma^{n-1-m} \sum_{j \in N_i} \frac{1}{r_{ji}} \left(\hat{O}_{ji}^{(m)} + \hat{\tau}_j(n-1) \right) \right] - \sum_{j \in N_i} \frac{1}{r_{ji}} \hat{\tau}_j(n-1)$$

Now, let us replace the last expression in the equation of $\hat{\tau}_i^{(k+1)}(n)$ and after some simple algebraic operations, we will obtain the following decentralized recursive algorithm (the steps are very similar to the previous case, hence are not detailed here):

$$\boxed{\begin{array}{l} \hat{\tau}_i^{(k+1)}(n) = \hat{\tau}_i(n-1) + \dfrac{1}{\left[I_i(n)\right]^{-1}} \cdot \left\{ \sum_{j \in N_i} \dfrac{1}{r_{ji}} \cdot \left[\hat{O}_{ji}^{(n)} - \left(\hat{\tau}_i(n-1) - \hat{\tau}_j^{(k)}(n) \right) \right] + \left(\sum_{m=1}^{n-1} \gamma^{n-m} \right) \cdot \left[\sum_{j \in N_i} \dfrac{1}{r_{ji}} \cdot \left(\hat{\tau}_j^{(k)}(n) - \hat{\tau}_j(n-1) \right) \right] \right\} \\ \left[I_i(n)\right]^{-1} = \gamma \cdot \left[I_i(n-1)\right]^{-1} + \sum_{j \in N_i} \dfrac{1}{r_{ji}} \qquad\qquad i = 2,3,\ldots N \end{array}} \tag{8.1.4}$$

where: $\left[I_i(0)\right]^{-1} = \left[P_i(0)\right]^{-1} = \dfrac{1}{p_i}$.

As expected, the algorithm in (8.1.4) reduces to the previous one when $\gamma = 1$ (without discount factor). The above algorithm is optimal (in the MMSE sense) given the previous model measurements, since we proved that it is equivalent to the Kalman Filter algorithm. Moreover, this algorithm is easy to implement as it requires communication only with neighbors, allowing us to implement it locally.

In the equation of $\left[I_i(n)\right]^{-1}$, the interpretation of the discount factor is clear. Since $\left[I_i(n)\right]^{-1}$ is a measure of the inverse covariance matrix of the error estimate, it is logical that the new information depends on the previous information multiplied by γ (like a forgetting factor) and on the inverse covariance of the measurements. We note that it is straightforward to obtain the above algorithm in the case where the matrix R^{-1} is different for each set of measurements.

8.2 Addition of Process White Noise

Now, we will include a white noise $\underline{w}(n)$ to the clocks readings. In this case, the state space model will be given by:

$$\begin{cases} \underline{x}(n+1) = \underline{x}(n) + \underline{w}(n) \\ y(n) = A^T \underline{x}(n) + \underline{v}(n) \end{cases} \tag{8.2.1}$$

We can interpret the noise $\underline{w}(n)$ as a measure of the unknown difference between two successive offsets or as a compensation of the uniform skew assumption and the time-invariant offsets. This is different from the incorporation of a discount factor because it gives more flexibility through the choice of the noise statistics parameters.

We assume that $\underline{w}(n)$ is modeled as a white Gaussian noise with zero mean and covariance $Q(n) \geq 0$. Namely:

$$E\left[\underline{w}(n)\right] = 0 \quad Cov\left[\underline{w}(k), \underline{w}(l)\right] = Q(k) \cdot \delta_{kl}$$

Moreover, we assume that the process noise and the measurement noise are statistically independent:

$$Cov\left[\underline{w}(k), \underline{v}(l)\right] = 0 \quad \forall k, l$$

In this case, the optimal estimate obtained by applying the Kalman Filter is equivalent to the constrained minimization of the deterministic objective function (see the proof in Appendix A):

$$\left.\begin{aligned} J\left(\underline{x}(0), \underline{x}(1), ..., \underline{x}(n)\right) = (\underline{x}(0) - \overline{x}_0)^T P_0^{-1} (\underline{x}(0) - \overline{x}_0) + \underline{w}^T Q^{-1} \underline{w} + \\ + \sum_{k=0}^{n} (y(k) - A^T \underline{x}(k))^T R^{-1} (y(k) - A^T \underline{x}(k)) \\ subject\ to: \underline{x}(n+1) = \underline{x}(n) + \underline{w}(n) \qquad\qquad\qquad \end{aligned}\right\} \xrightarrow{\min} \hat{\underline{x}}_{opt}$$

We note that this is the same objective function as before, with the addition of the last term. By using the constraint, we can write this last term in the following way:

$$\underline{w}^T Q^{-1} \underline{w} = \sum_{k=1}^{n} \underline{w}_k Q_k^{-1} \underline{w}_k = \sum_{k=1}^{n} \left(\underline{x}(k) - \underline{x}(k-1)\right)^T Q_k^{-1} \left(\underline{x}(k) - \underline{x}(k-1)\right)$$

As usual, we will assume that the initial inverse covariance matrix P_0^{-1} is diagonal. In addition, we will assume that the covariance matrix of the process noise is diagonal and time invariant, i.e., $Q(k) = Q \ \ \forall k = 1, ..., n$.

In this new case, the offsets are time varying and the optimal solution is time dependent:

$$\begin{pmatrix} \hat{\underline{x}}_{opt}(0) \\ . \\ . \\ . \\ \hat{\underline{x}}_{opt}(n) \end{pmatrix} = \begin{pmatrix} \hat{\underline{x}}_{0|n} \\ . \\ . \\ . \\ \hat{\underline{x}}_{n|n} \end{pmatrix}$$

The preceding method cannot be applied to obtain the optimal solution in this case; the time dependency makes the estimation problem more difficult.

With the purpose of finding these optimal time varying offsets, we may compute the derivatives of the objective function with respect to the offsets vectors at time 0,1,...,n and to set them to zero:

$$\begin{cases} \dfrac{\partial J}{\partial \underline{x}(0)} = P^{-1}\underline{x}(0) - P^{-1}\overline{x}_0 - \left(AR^{-1}\right)y(0) + \left(AR^{-1}A^T\right)\underline{x}(0) + Q^{-1}\underline{x}(0) - Q^{-1}\underline{x}(1) = 0 \\ . \\ . \\ . \\ \dfrac{\partial J}{\partial \underline{x}(k)} = \left(AR^{-1}A^T\right)\underline{x}(k) - \left(AR^{-1}\right)y(k) + \cancel{Q^{-1}\underline{x}(k)} - Q^{-1}\underline{x}(k-1) - \cancel{Q^{-1}\underline{x}(k)} + Q^{-1}\underline{x}(k+1) = 0;\ \forall k = 1,...,n-1 \\ \dfrac{\partial J}{\partial \underline{x}(n)} = \left(AR^{-1}A^T\right)\underline{x}(n) - \left(AR^{-1}\right)y(n) + Q^{-1}\underline{x}(n) - Q^{-1}\underline{x}(n-1) = 0 \end{cases}$$

One can observe that the solution is not simple since each vector depends on the estimates at different times. Indeed, we see that the estimate at time 0 depends on that at time 1, the estimate at time k ($\forall k = 1,...,n-1$) depends on its adjacent estimates (at times k+1 and k-1), and finally the one at time n depends on that at time n-1.

Our goal is to develop a recursive relation of the following form:

$$\hat{\underline{x}}_{n+1|n+1} = f\left(\hat{\underline{x}}_{n|n}, y(n+1)\right)$$

The method described above is not efficient to solve this problem. Hence, we choose to write the Kalman Filter equations. In the previous notation, we have: $\Phi = I,\ H = A^T$.
The KF equations are therefore:

Time update (prediction): $\begin{cases} \hat{\underline{x}}(n+1|n) = \hat{\underline{x}}(n|n) \\ P(n+1|n) = P(n|n) + Q \end{cases}$

Measurement update:

$$\begin{cases} \hat{\underline{x}}(n+1|n+1) = \hat{\underline{x}}(n+1|n) + K(n+1)\left[\underline{y}(n+1) - A^T\hat{\underline{x}}(n+1|n)\right] \\ K(n+1) = P(n+1|n)A\left[A^TP(n+1|n)A + R\right]^{-1} = P(n+1|n+1)AR^{-1} \\ P(n+1|n+1) = \left[I - K(n+1)A^T\right]P(n+1|n) \end{cases}$$

The recursive combined Kalman Filter equations are given by:

$$\boxed{\begin{cases} \underline{\hat{x}}(n+1|n+1) = \underline{\hat{x}}(n|n) + P(n+1|n+1)AR^{-1}\left[y(n+1) - A^T\underline{\hat{x}}(n|n)\right] \\ P^{-1}(n+1|n+1) = P^{-1}(n+1|n) + AR^{-1}A^T = \left(P(n|n)+Q\right)^{-1} + AR^{-1}A^T \end{cases}} \qquad (8.2.2)$$

We can see that if $Q = 0$ (there is no process noise), we achieve the same result as in the basic case. As we previously mentioned, the inverse covariance matrix is not diagonal after one step (due to the addition of the term $AR^{-1}A^T$). Combining these two equations leads to the following recursive centralized algorithm:

$$\underline{\hat{x}}(n+1|n+1) = \underline{\hat{x}}(n|n) + \left[\left(P(n|n)+Q\right)^{-1} + AR^{-1}A^T\right]^{-1} AR^{-1}\left[y(n+1) - A^T\underline{\hat{x}}(n|n)\right] \qquad (8.2.3)$$

The only difference with the previous case is the presence of the covariance matrix Q. In the case without process noise, we succeeded in developing a decentralized version of this recursive centralized algorithm. However, when a process noise is incorporated to the state space model, the new structure of the inverse covariance matrix $P^{-1}(n+1|n+1)$ does not enable the application of the same procedure. Moreover, a decentralized non-recursive (batch) algorithm is not an option, as there is a correlation between the different measurements, so we cannot compute the equivalent measurement easily. The single solution we propose is to apply the recursive centralized Kalman Filter algorithm given in (8.2.3). It will lead to an optimal on-line solution, but requires communication between all the nodes over the network (or the existence of a central unit).

In summary, in the case of the presence in the system of a process white noise, we have not developed a recursive decentralized algorithm to optimally estimate the offsets at each network node. The approaches that we employed in the previous case are not applicable here, and this problem is still unsolved. On the other hand, the addition of a process noise in our model is not compulsory since the offsets and the skew can be assumed to be constant over some known time intervals. Moreover, this assumption can be compensated through the incorporation of a discount factor as explained in the previous section.

8.3 Faulty Communication Environment

One desirable attribute of any decentralized algorithm is robustness to communication failures, such failures being unavoidable in practice. So far, we considered several algorithms that iteratively compute the optimal estimates, assuming perfect communication channels (no failures). In this section, we improve the algorithms to handle with dynamic changes in the communication topology by considering temporary link failures. We slightly modify our algorithm to become robust to temporary communication failures. Since a neighbor may become unavailable at any time, every node stores in its local memory the estimates of its neighbors' variables recorded from the last successful communication exchange. We denote by $\left(\hat{\tau}_i\right)_j^{(k)}$ the estimate of Λ_i's clock offset kept in Λ_j's local memory at the end of the k-th iteration. If the last successful communication between Λ_i and Λ_j took place during the l-th iteration, while $l < k$, then $\left(\hat{\tau}_i\right)_j^{(k)} = \hat{\tau}_i^{(l)}$.

Let $\bar{N}_i^{(k)} \subseteq N_i$ be the subset of Λ_i's neighbors that send and receive data successfully during the k-th iteration. In other words, node Λ_i gets from every $\Lambda_j \in \bar{N}_i^{(k)}$ the most recent estimates and updates its copy of its neighbors' estimates. For the rest of the neighbors, the communication fails, so the local copies remain unchanged.

In mathematical notation:

$$\left(\hat{\tau}_i\right)_j^{(k)} = \begin{cases} \hat{\tau}_i^{(k)}; \ \forall \Lambda_i \in \bar{N}_j^{(k)} \\ \hat{\tau}_i^{(k-1)}; \ \forall \Lambda_i \in N_j \setminus \bar{N}_j^{(k)} \end{cases} \tag{8.3.1}$$

Then, node Λ_i computes its own estimate at the $k+1$ iteration by one of the algorithms developed in the previous sections. The only difference is that the estimates of the neighbors are taken according to the above formula and depend on the failures in the communication links.

This extension was inspired by [3], where the authors apply this method for solving the time synchronization problem in a faulty communication environment for the WLS case ($P_0^{-1} = 0$). They also show its convergence to the optimal estimates when certain mild conditions on the failure rate are satisfied. Namely, they consider that there exists a positive integer $p < \infty$ such that the number of consecutive communication failures between every pair of neighboring nodes is less than p. One can apparently extend the result for the general Kalman Filter framework, but a convergence analysis must be investigated. In this research, we did not consider the latter analysis. Moreover, in [3] the authors propose an initialization scheme that improves the accuracy of the estimates. The corresponding details are omitted here.

In summary, we propose a modification to our algorithm so it may work even in the presence of faulty communication. In fact, the WLS algorithm is proved (in [3]) to converge to the optimal solution even in the presence of link failures, whereas for the decentralized KF, such a proof was not investigated here and can be considered as a future research direction.

9. Clock Skew Estimation

In the previous analysis, we considered a simple model where all the clocks progress at the same rate ($\alpha_i = \alpha_j = 1$, i.e., there is no skew), but have arbitrary offsets. In other words, the estimation problem was reduced to the estimation of the clock offsets. In this section, we extend the results to the case of general clock skew, when the clock offset parameters are still assumed to be time-invariant and our objective is to estimate both the clock offsets and the rate offsets at each network node. Naturally, the importance of estimating the clock skew increases as the measurements (and estimates) are conducted over longer time intervals. If all the measurements are obtained simultaneously, the skew parameter is irrelevant. We still suppose that the clock drift at a node follows the linear form: $T_i(t) = \alpha_i t + \tau_i$, where α_i and τ_i are the skew (rate offset) and the offset parameters respectively, t is the real time (or the reference time) and T_i is the local time (at node Λ_i). Our goal is to estimate the parameters α_i that describe the rate of the local clocks relative to the reference clock ($\alpha_1 = 1$) in addition to the offsets τ_i. If one knew the constants α_i, then one could estimate τ_i as in the previous sections by first dividing all the local clocks readings by α_i. Thus, we must now describe how to obtain estimates of these skew values α_i, and do so without prior knowledge of the offsets τ_i or to propose an algorithm that estimates both parameters simultaneously.

In this section, we will divide the analysis into two different approaches. The first approach estimates the time offsets and the rate offsets simultaneously, whereas the second is based on a separate estimation of both parameters. The latter treats the clock offset and the clock skew on different time scales like the scheme in [21]. First, we will describe a procedure to estimate simultaneously both the offsets and the skews using a combined Kalman Filter algorithm. We will consider a centralized optimal algorithm and describe briefly its decentralized implementation. Second, we treat the case of separate skew and offset estimation problem. We propose three different algorithms in order to estimate the skew parameter at each node over the network with respect to the reference clock. The procedure of each method will be detailed, before comparing their characteristics with respect to one another. The first method was introduced in the literature (see [41] and [21]) and is characterized by the application of the logarithmic function on the skew parameters. The second method is mathematically equivalent to the first one, but does not require the application of the logarithm. The latter is named the multiplicative method due to its multiplicative nature. Both these methods are based on Least-Squares minimization of appropriate cost functions. The third method uses the same measurement model as the offset estimation problem and in opposition to the two other methods, is only sub-optimal. The second and the third method are to the best of our knowledge original.

We note that the exposition in this chapter is relatively brief, and we avoid repeating details that are similar to previous chapters.

9.1 Combined Skew and Offset Estimation

Our purpose is to develop a unified algorithm in order to estimate the clock parameters (both clock offset and skew) simultaneously. First, we write the state space model of our problem that includes the clock skew influence. Then, we will optimally solve the estimation problem by applying the Kalman Filter algorithm to the state space vector that contains all the clock parameters (both clock offset and skew) at each network node. In this part, we consider the same measurement model as in the offset estimation problem (see Figure 4.1).

In our clock model, the local time (at node Λ_i) is given by: $T_i(t) = \alpha_i t + \tau_i$, where α_i and τ_i are the skew (rate deviation) and the clock offset parameters respectively and t is the real time (or the reference time). Let us define the state vector $\underline{x}(t)$ with its i-th element given by:

$$x_i(t) = T_i(t) - t = (\alpha_i - 1)t + \tau_i$$

Let us now perform a uniform discretization of the previous continuous-time equation (for simplicity, we assumed that the sampling interval is uniform and denoted by T_S):

$$x_i(n) = (\alpha_i - 1)T_S n + \tau_i = x_i(0) + T_S n \cdot (\alpha_i - 1) \tag{9.1.1}$$

We define: $b_i \triangleq (\alpha_i - 1)$ as a constant random bias at node Λ_i and $\underline{b} = (b_1 = 0, b_2, ..., b_N)^T$. In other words, the bias $\underline{b}$ is equivalent to the skew parameter minus 1. In vector form, we obtain:

$$\underline{x}(n) = \underline{x}(0) + nT_S \cdot \underline{b}$$

or, equivalently:

$$\begin{aligned} &\underline{x}(n) = \underline{x}(n-1) + T_S \cdot \underline{b} \\ &\underline{x}(0) = \underline{\tau} \end{aligned}$$

Namely, we have incorporated a constant random bias to the state space dynamical equation. Then, the state space model that includes the clock skew is given by:

$$\begin{cases} \underline{x}(n+1) = \underline{x}(n) + T_S \cdot \underline{b} \\ y(n) = A^T \underline{x}(n) + \underline{v}(n) \end{cases} \tag{9.1.2}$$

Here, we assume that $\underline{b} \sim N(0, B)$ where B is the bias covariance matrix and is assumed to be diagonal. Moreover, $\underline{x}(0)$ and $\underline{b}$ are assumed to be statistically independent.

In brief, we showed that incorporating an additive bias to the dynamical equation of the state space model is equivalent to add a multiplicative skew parameter to the clock model.

Let us define the augmented state space vector by:

$$X(n) = \begin{pmatrix} \underline{x}(n) \\ \underline{b} \end{pmatrix} \tag{9.1.3}$$

For this augmented state space vector, we will have the following state space model:

$$\begin{cases} X(n+1) = \begin{pmatrix} \underline{x}(n+1) \\ \underline{b} \end{pmatrix} = \begin{pmatrix} I & T_S I \\ 0 & I \end{pmatrix} \begin{pmatrix} \underline{x}(n) \\ \underline{b} \end{pmatrix} = \Phi X(n) \\ y(n) = \begin{pmatrix} A^T & 0 \end{pmatrix} X(n) + \underline{v}(n) \end{cases} \qquad (9.1.4)$$

It is now possible to apply the standard KF equations in order to obtain a centralized optimal estimate of both skew $\underline{b}$ and offset $\underline{x}(n)$. Our objective is to estimate the entire augmented vector given in (9.1.3) (that is, the offset and the skew at each node over the network) in an optimal way. In addition, we will attempt to develop a decentralized version that hopefully converges to the optimal centralized solution.

Remark

An alternative easier method is to estimate the vector $\begin{pmatrix} \underline{x}(0) \\ \underline{b} \end{pmatrix}$. This is exactly equivalent because if one knows the vector $\begin{pmatrix} \underline{x}(0) \\ \underline{b} \end{pmatrix}$, one can easily compute the general vector $\begin{pmatrix} \underline{x}(n) \\ \underline{b} \end{pmatrix}$ using the relation:

$$\underline{x}(n) = \underline{x}(0) + nT_S \cdot \underline{b}$$

a) Centralized Kalman Filter Algorithm

In order to find the optimal parameters, we define the corresponding objective function (Least-Squares approach) and we set its gradients with respect to $\underline{\tau}$ and $\underline{b}$ to zero. An additional alternative is to write the Kalman Filter equations for the augmented state space vector. Then, we will obtain a vector equation for estimating the optimal offsets together with the optimal skew parameters at each node over the network. The details are hereby omitted for the simple reason that the procedure is very similar to the basic case, and the mathematical manipulations are of no particular interest.

b) Decentralized Implementation

We briefly discuss the decentralized implementation of the optimal centralized solution. Proceeding similarly to Section 5, it is possible to develop a decentralized Jacobi-like iteration for this problem. Unfortunately, this algorithm generally diverges (the spectral radius of the iteration matrix is bigger than 1). Actually, we obtained a decentralized algorithm that allows us to estimate both the skew and the offset and we have shown (by a simple numerical example) that this algorithm does not converge. We omit the details here, as they do not contain any insights. As a future research direction, it can be interesting to check if the decentralized iterative algorithm converges to the optimal centralized solution in the case of a reduced step size. Namely, if instead of the Jacobi iterative procedure, we try to employ a relaxed Jacobi algorithm (or equivalently, the gradient method) with small step size (see Section 2.1).

As it stands, the problem is better solved by estimating the offsets and the skews simultaneously by the centralized optimal algorithm. The main advantage of this approach relies in that it will lead to the optimal solution; however its central drawback is that each node has to communicate with every other node. So far, a decentralized optimal algorithm for the combined estimation problem was not obtained. Therefore, we may consider the separate skew and offset estimation for which several decentralized methods are proposed in the next section.

9.2 Separate Skew and Offset Estimation

In this section, we propose to solve the time synchronization problem by estimating separately the clock offset and the clock skew (rate offset) at each network node. The clock offset estimation problem can be solved by one of the preceding algorithms (see sections 5 and 7 for the single measurement and the multiple measurement cases respectively). Three different methods are presented in order to solve the clock skew estimation problem.

a) The Logarithmic Method

In the subsequent analysis, the measurements are obtained in a different way than in the offset estimation problem. Namely, each node is sending a pair of probe packets located at significant time intervals (i.e., large compared to the variances of the individual measurements) to each one of its neighbors. Figure 9.1 depicts the situation for the pair of neighboring nodes Λ_i and Λ_j. Time is stamped on the packets k_1 and k_2 by the sender Λ_i upon transmissions ($T_i(k_1),\ T_i(k_2) \gg T_i(k_1)$) and by the receiver Λ_j upon receiving the packets ($R_j(k_1),\ R_j(k_2)$). Indeed, in order to obtain an accurate estimate of the skew parameter, we need to take a pair of probe packets well spaced in time. In other words, α_i can be regarded as the slope of the local time as a function of the reference time.

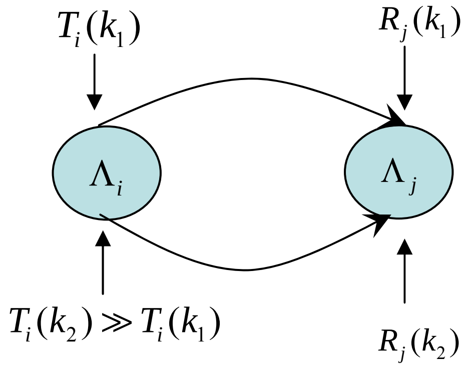


Figure 9.1. Communication between two neighboring nodes for the skew estimation problem.

Let t_m denote the transmission time of packet k_m $m = 1,2$ according to the reference time. Then, up to the time-stamping error (assuming that at time $t = 0$, the offset was τ_i):

$$T_i(k_m) = \alpha_i t_m + \tau_i$$

Similarly, let $\tilde{t}_m$ denote the received time of packet k_m $m = 1,2$. Then:

$$\tilde{t}_m = t_m + x_{ij}(k_m)$$
$$R_j(k_m) = \alpha_j \left(t_m + x_{ij}(k_m)\right) + \tau_j$$

Here, $x_{ij}(k_m)$ is the propagation delay of the packet k_m over the corresponding link. In the above equations, both $\left(t_1,t_2,\tilde{t}_1,\tilde{t}_2\right)$ and $\left(\tau_i,\tau_j\right)$ are unknown, in addition to the skew parameters of interest $\left(\alpha_i,\alpha_j\right)$. We now manipulate our measurements so that these extra unknowns are cancelled.

Let ΔR_j denote the time difference between the reception of probe packet k_2 by node Λ_j and the receiving time of packet k_1 at node Λ_j according to Λ_j's clock, namely:

$$\Delta R_j = R_j(k_2) - R_j(k_1)$$

Let ΔT_i denote the time difference between the transmission of probe packet k_2 by node Λ_i and the transmission time of packet k_1 at node Λ_i according to Λ_i's clock, namely:

$$\Delta T_i = T_i(k_2) - T_i(k_1)$$

If we divide ΔT_i by ΔR_j, we obtain an estimate of the relative skew $\alpha_{ji} = \dfrac{\alpha_i}{\alpha_j}$ (assuming that the transmission delay and the offsets remain constant for the different probe packets):

$$\frac{\Delta T_i}{\Delta R_j} = \frac{T_i(k_2)-T_i(k_1)}{R_j(k_2)-R_j(k_1)} \cong \frac{\left(\alpha_i t_2 + \cancel{\tau_i}\right)-\left(\alpha_i t_1 + \cancel{\tau_i}\right)}{\left(\alpha_j t_2 + \cancel{\tau_j}\right)-\left(\alpha_j t_1 + \cancel{\tau_j}\right)} = \frac{\alpha_i}{\alpha_j} \qquad (9.2.1)$$

It is natural to divide the above quantities in order to cancel the influence of both the offsets and the time interval. If we apply the logarithm function to the quantity $\dfrac{\Delta T_i}{\Delta R_j}$, we will obtain a relative measurement of the skew logarithm:

$$z_{ji} \triangleq \log\frac{\Delta T_i}{\Delta R_j} \cong \log\left(\alpha_i\right) - \log\left(\alpha_j\right) \qquad (9.2.2)$$

The term z_{ji} in (9.2.2) corresponds to the measurement of the node pair Λ_i and Λ_j.

Then, one can employ the previous methodology of Section 5 in order to estimate the skew logarithm at each node. In other words, if we substitute z_{ji} for $\hat{O}_{ji}$ and $\beta_i \triangleq \log\left(\alpha_i\right)$ for τ_i, we obtain the same mathematical problem as the previous offset estimation problem. For example, let us develop the optimal decentralized algorithm for the basic statistical LS case, namely: $R = R^{-1} = I$ and $P_0^{-1} = 0$. The state vector and the objective function to be minimized are given by:

$$\underline{x}(n) \doteq \left(\beta_1 = \log\left(\alpha_1\right) = 0, \beta_2 = \log\left(\alpha_2\right), \ldots \beta_N = \log\left(\alpha_N\right)\right)^T \qquad (9.2.3)$$

$$J = (y - A^T\underline{x})^T(y - A^T\underline{x}) = \sum_{\substack{i,j \\ j\in N_i}} \left(z_{ji} - \beta_i + \beta_j\right)^2$$

In order to minimize J, we may compute the partial derivatives with respect to β_i and set them to zero (the procedure is very similar to the one performed in Section 5):

$$\frac{\partial J}{\partial \beta_i} = \left(AA^T\right)_i \underline{x} - A_i y = -2\sum_{j\in N_i}\left(z_{ji} - \beta_i + \beta_j\right) =$$

$$= -2\left[-\beta_i \cdot \sum_{j\in N_i} 1 + \sum_{j\in N_i}\left(z_{ji} + \beta_j\right)\right] = 0$$

$$\begin{cases} \left(AA^T\right)_i \underline{x} = |N_i|\beta_i - \sum_{j\in N_i}\beta_j \\ A_i y = \sum_{j\in N_i} z_{ji} \end{cases}$$

Substituting into the partial derivatives leads to:

$$\frac{\partial J}{\partial \beta_i} = |N_i|\beta_i - \sum_{j\in N_i}\left(z_{ji} + \beta_j\right) = 0$$

From this, we can as usual employ an iterative (synchronous) algorithm in order to implement the above optimal equation:

$$\boxed{\widehat{\beta_i}^{(k+1)} = \frac{1}{|N_i|}\sum_{j\in N_i}\left(z_{ji} + \widehat{\beta_j}^{(k)}\right)} \tag{9.2.4}$$

Initialization: $\widehat{\beta_i}^{(0)} = \log\left(\alpha_i(0)\right) \quad i = 2,3,...N$. Here, $k \geq 0$ is the iteration number.

The procedure in (9.2.4) represents the decentralized synchronous algorithm that implements the optimal equation in order to estimate the rate offset logarithm at each network node given a single pair of measurements. One can easily generalize for the general Kalman Filter framework in a similar manner. The procedure is exactly the same; we have just to perform the following substitutions:

$$\begin{cases} \hat{O}_{ji} \to z_{ji} \triangleq \log\left(\frac{\Delta T_i}{\Delta R_j}\right) \\ \tau_i \to \beta_i = \log\left(\alpha_i\right) \end{cases} \tag{9.2.5}$$

After applying the above procedure, one can optimally estimate the rate offset logarithm at each node over the network with respect to the reference node. Consequently, we have shown that the clock skew estimation problem reduces to the same mathematical setup as the offset estimation problem under the appropriate substitutions in (9.2.5).

Next, we propose an alternative method in order to estimate the skew parameters without requiring the application of the logarithm function on the measurement sets.

b) The Multiplicative Method

Now, we will develop an additional method to estimate the skew parameters without requiring the application of the logarithm function on the sets of measurements. As we previously explained, the estimation of the skew values α_i has to be done without the prior knowledge of the offsets τ_i. We note that the only way to get rid of the dependence of the real time t is to divide ΔT_i by ΔR_j:

$$\varsigma_{ji} \triangleq \frac{\Delta T_i}{\Delta R_j} = \frac{T_i(k_2) - T_i(k_1)}{R_j(k_2) - R_j(k_1)} \cong \frac{\alpha_i}{\alpha_j} \tag{9.2.6}$$

The term ς_{ji} in (9.2.6) corresponds to the measurement of the node pair Λ_i and Λ_j.

In this case, we do not apply the logarithm function but instead we define the following objective function which is to be minimized:

$$J(\alpha) = \sum_{i,j \in N_i} \left(\varsigma_{ji} - \frac{\alpha_i}{\alpha_j} \right)^2$$

We considered here the basic LS case, but one can easily generalize to the WLS case. Let us now differentiate $J(\alpha)$ according to α_k and set the partial derivatives to zero:

$$\frac{\partial J}{\partial \alpha_k} = -\cancel{2} \sum_{j \in N_k} \frac{1}{\alpha_j} \left(\varsigma_{jk} - \frac{\alpha_k}{\alpha_j} \right) + \cancel{2} \sum_{j \in N_k} \frac{\alpha_i}{(\alpha_k)^2} \left(\varsigma_{ki} - \frac{\alpha_i}{\alpha_k} \right) = 0$$

$$\sum_{j \in N_i} \frac{-\cancel{1}}{\cancel{\alpha}_j} \left(\varsigma_{ji} - \frac{\alpha_i}{\alpha_j} \right) + \sum_{j \in N_i} \frac{\alpha_i}{(\alpha_j)^{\cancel{2}}} \left(\varsigma_{ji} - \frac{\alpha_i}{\alpha_j} \right) = 0$$

$$\sum_{j \in N_i} \left(\frac{\alpha_i}{\alpha_j} - 1 \right) \left(\varsigma_{ji} - \frac{\alpha_i}{\alpha_j} \right) = 0$$

$$(\alpha_i \neq \alpha_j \neq 0) \qquad \sum_{j \in N_i} \left(\alpha_i - \alpha_j \cdot \varsigma_{ji} \right) = 0$$

From this, we can employ an iterative (synchronous) algorithm in order to implement the above optimal equation:

$$\boxed{\hat{\alpha}_i^{(k+1)} = \frac{1}{|N_i|} \sum_{j \in N_i} \varsigma_{ji} \cdot \hat{\alpha}_j^{(k)}} \tag{9.2.7}$$

We can note that the synchronous decentralized algorithm in (9.2.7) has exactly the same form as the previous algorithm with the logarithm function in (9.2.4). The difference is that in this case, the update is multiplicative and in the other case the update was additive.

The main drawback of the two previous methods is that the measurements for the offset estimation and for the skew estimation are not similar. Namely, the measurement model for the offset estimation problem is composed of a fast bilateral exchange (see Figure 4.1). In the measurement model of the skew estimation problem, node Λ_i sends a pair of probe packets located at significant time intervals to node Λ_j (see Figure 9.1). Since we are interested in performing both the offset and the skew synchronizations, it is valuable to develop an algorithm that employs the same measurement format in the different procedures.

Next, we propose an additional sub-optimal method for skew estimation that requires the same measurements format as the offset estimation problem.

c) State-Space based Solution

Now, we consider the same measurement model as in the offset estimation problem (see Figure 4.1). For this measurement model, it was previously shown that the measurement equation of the state space model is given by:

$$y(n) = A^T \underline{x}(n) + \underline{v}(n)$$

The entries of the vector $\underline{x}(n)$ are defined by the offsets at each node at time t:

$$x_i(n) = T_i(n) - n = (\alpha_i - 1)n + \tau_i$$

Here, $n \geq 0$ is the discrete time index and $y(n)$ is the measurement set of every pair of neighboring nodes at time n. We note that n need not refer to the actual time, but rather corresponds to the epoch when the n-th measurement set $y(n)$ become available. The initial state of the system $x(0)$ has the following first and second order statistics: $E\left[x(0)\right] = \overline{x}_0$ $\operatorname{cov}\left[x(0)\right] = P_0$. $\{v(n)\}$ is the measurement noise modeled as a white noise with zero mean and covariance $R(n) = R > 0$. We assume that $\{v(n)\}$ is uncorrelated and therefore the matrix R is diagonal and Positive Semi-Definite (PSD). Its i-j element corresponds to the pair of neighboring nodes Λ_i and Λ_j:

$$\left(R\right)_{ij} = r_{ji}$$

$\{v(n)\}, x(0)$ are uncorrelated, that is: $E\left[x(0)v^T(n)\right] = 0 \;\; \forall n$.

Let us denote:

$$b_i \triangleq (\alpha_i - 1)$$
$$\underline{b} = \left(b_1 = 0, b_2, ..., b_N\right)^T$$

Then:

$$\underline{x}(n) = \underline{b} \cdot n + \underline{\tau}$$
$$y(n) = A^T\left(\underline{b} \cdot n + \underline{\tau}\right) + \underline{v}(n)$$

We consider the multiple measurement case and we propose a decentralized sub-optimal algorithm that estimates the offset and the skew separately. The offsets are estimated after each set of measurements in a recursive way (for more details, see Section 7), and the skew parameters are estimated after $n = T_b$ sets of measurements only (according to the pair of farthest measurements). This is a sub-optimal scheme since the estimate of $\underline{b}$ is not performed in accordance with all the sets of measurements, and we are not taking into account the dependence between $\underline{\tau}$ and $\underline{b}$. We would like to emphasize the fact that this algorithm is a heuristic method (non-optimal) in opposition to the majority of the previous algorithms that were optimal in the sense that they achieve the minimal value of an objective function.

We will now present in more details this decentralized algorithm that estimates the skew parameter at each node over the network. As we explained in the beginning of the present section, it is logical to estimate the skew parameter according to a pair of measurements well spaced in time (because it is like a slope estimation problem), whereas the offsets can be estimated after each set of measurements. Let us compute the difference between $y(T_b)$ and $y(0)$ (assuming that the offsets τ_i are time-invariant):

$$\begin{cases} y(T_b) = A^T\left(\underline{b}\cdot T_b + \underline{\tau}\right) + \underline{v}(T_b) \\ y(0) = A^T\underline{\tau} + \underline{v}(0) \end{cases}$$

$$\frac{y(T_b) - y(0)}{T_b} = A^T \cdot \underline{b} + \frac{\underline{v}(T_b) - \underline{v}(0)}{T_b} \tag{9.2.8}$$

One can easily note that we have obtained in (9.2.8) an equation that is similar to the measurement equation of the offset estimation problem (see Section 4.4). The only difference is that the measurements and the noises are divided by the number of measurement sets T_b. The consequence is that now, the covariance matrix of the noise will be equal to $\frac{2}{(T_b)^2}\cdot R$, i.e., the measurement noise distribution is given by:

$$\underline{\tilde{v}} = \frac{\underline{v}(T_b) - \underline{v}(0)}{T_b} \sim N\left(0, \frac{2}{(T_b)^2}R\right).$$

In other words, we showed that the skew estimation problem considered here reduces to the basic offset estimation problem of Section 5 under the following substitutions:

$$\begin{cases} \underline{x} \to \underline{b} \\ y \to \dfrac{y(T_b) - y(0)}{T_b} \\ \underline{v} \to \underline{\tilde{v}} \end{cases} \tag{9.2.9}$$

For example, the synchronous decentralized algorithm that estimates b_i in the most basic case (Least-Squares estimation) is given by:

$$\hat{b}_i^{(k+1)} = \frac{1}{T_b \cdot |N_i|} \sum_{j \in N_i} \left(\hat{O}_{ji}^{(T_b)} - \hat{O}_{ji}^{(0)} + T_b \cdot \hat{b}_j^{(k)} \right) \qquad (9.2.10)$$

In the general case (DKF), we will obtain the following synchronization procedure:

$$\hat{b}_i^{(k+1)} = \frac{1}{T_b \cdot \left(\sum_{j \in N_i} \frac{1}{2r_{ji}} + \frac{1}{B_i} \right)} \left[\sum_{j \in N_i} \frac{1}{2r_{ji}} \left(\hat{O}_{ji}^{(T_b)} - \hat{O}_{ji}^{(0)} + T_b \cdot \hat{b}_j^{(k)} \right) + T_b \cdot \frac{\bar{b}_i(0)}{B_i} \right] \qquad (9.2.11)$$

Here, we assume as usual that $\underline{b} \sim N(0,B)$ where B is the bias covariance matrix and is assumed to be diagonal.

In practice, we will estimate the offsets according to one of the previous developed algorithms in a recursive way (after each set of measurements). After that enough sets of measurements, say T_b arrived, one can estimate $\hat{b}_i$ at each node over the network according to one of the previous algorithms. While we estimate $\hat{b}_i$ at each node over the network, we can easily compute the skew parameter using the relation:

$$\hat{\alpha}_i = \hat{b}_i + 1.$$

Then, we will assume that the skew parameter remains constant during a known constant time τ_b, so we have just to normalize the measurements by the estimated skew:

$$\frac{T_i}{\hat{\alpha}_i}, \frac{R_i}{\hat{\alpha}_i}.$$

In order to determine the correct value of the constant time τ_b, we have to know how much time the skew can be assumed to stay constant in our model. This can be done according to the literature on how to model a clock and is beyond the scope of this research. After τ_b time units, one can estimate once more the new parameters $\hat{b}_i$ at each node over the network according to the first and the last measurements of this new time period. In other words, after each set of measurements we will estimate $\hat{\tau}_i \left(i = 2,...,N \right)$, whereas $\hat{b}_i \left(i = 2,...,N \right)$ is estimated according to $y\left(T_b\right)$ and $y\left(0\right)$ at the first cycle, $y\left(2T_b\right)$ and $y\left(T_b + 1\right)$ at the second cycle, etc.

<u>Remark:</u> The number T_b can be different for each node namely; each node can update the skew of its own clock at his most appropriate time.

This approach is similar to the scheme in [21], where the authors treat skew synchronization and offset synchronization on different time scales. That is, the parameters α_i are adjusted every τ_{skew} time units, whereas the parameters τ_i are adjusted every τ_{offset} time units, with:

$$\boxed{\tau_{skew} \gg \tau_{offset}}.$$

So far, we have obtained several decentralized algorithms for estimating the skew parameters $\underline{b}$ with no dependence on the offsets $\underline{\tau}$.

In summary, we have developed several decentralized algorithms for clock synchronization that can deal with general clocks, including both offsets and skews. The first option is to estimate the clock offset and skew simultaneously whereas the second alternative estimates them separately. Concerning the combined estimation problem, only a brief description was considered and no concrete results were presented. For the separate estimation method, three algorithms were proposed in order to estimate the skew parameter. The sub-optimal solution is the only one that uses the same measurement format as that of the offset estimation procedure. Nevertheless, the parameter T_b must be known, and the result is only sub-optimal considering that only two sets of measurements are used. In the two other optimal methods, there is no such requirement on T_b (because $t_2 - t_1$ is cancelled), yet the algorithms are based on measurements that are not similar to the offset estimation problem.

In the next section, we present simulation results over several network topologies for evaluating and comparing the accuracy of the proposed time synchronization schemes.

10. Numerical Results

In this section, we implement some of the algorithms that we previously developed for typical problems and we compare the results with the existing algorithms. First of all, let us describe in a concise way the different algorithms that we chose to implement.

10.1 Algorithms Description

CTP: ("Classless Time Protocol" [14]). This algorithm computes each offset by calculating the average of the relative offsets of all the adjacent nodes and is equivalent to performing the Least-Squares statistical method. For example, if one node has two neighbors with relative offsets of (+1) and (–2) respectively, the node adjusts its own clock by: $\frac{+1-2}{2}=-0.5$. In [14], the authors used a measurement filter in order to obtain less noisy measurements (low queuing delay). The procedure is repeated at each node and for each set of measurements until convergence. In simulations, both the centralized and the decentralized versions are considered.

WLS: Similar to CTP, but now the offsets are calculated using the Weighted Least-Squares method. Namely, each offset is estimated as the average over the neighbors' relative measurements, but each measurement is multiplied by a weight according to its accuracy. For example, in the simulations we made the logical assumption that the links with a smaller queuing delay (i.e., there is a light load) have a smaller variance in their measurements. As a consequence, links with small queuing delays are associated to a bigger weight when evaluating the offsets.

DKF: An additional decentralized algorithm based on the Kalman Filter framework. This algorithm is related to the following state space equations:

$$\begin{cases} \underline{x}(n+1)=\underline{x}(n) \\ y(n)=A^T\,\underline{x}(n)+\underline{v}(n) \end{cases}$$

Here, $\underline{x}(n)\doteq\left(\tau_1=0,\tau_2,\ldots\tau_N\right)^T$ (τ_i is the offset of the node Λ_i), $y(n)$ is the measurement at time n and A is the reduced incidence matrix (for more details, see the problem formulation and the scientific background sections). Moreover, we assume the following usual assumptions:

- $\underline{x}(0)$ is the initial Gaussian state of the system with the following first and second order statistics:

$$E\left[\underline{x}(0)\right]=\underline{m}_x(0)\quad \operatorname{cov}\left[\underline{x}(0)\right]=E\left[\left(\underline{x}(0)-\underline{m}_x(0)\right)\left(\underline{x}(0)-\underline{m}_x(0)\right)^T\right]=P_x(0)=P_0$$

- $\{\underline{v}(n)\}$ is the measurement white Gaussian noise with zero mean and covariance $R(n)=R>0$.

- $\{\underline{v}(n),\underline{x}(0)\}$ are uncorrelated for any n.

As we previously show, the decentralized Kalman Filter algorithm that we developed converges to the optimal centralized solution under the appropriate conditions. We chose to implement the DKF algorithm in a synchronous way, with only one set of measurements, but we will examine several different values of the covariance matrices P_0 and R.

CKF and CLS: The centralized Kalman Filter and the centralized Least-Squares algorithms in their standard form. For more details, see Section 5.2.

SOA: The sub-optimal algorithm that neglects the off-diagonal terms of the inverse covariance matrix. For more details, see Section 7.4.

As we previously mentioned, NTP is the most widely accepted standard for synchronizing clocks over the internet [28-30]. The three following algorithms are different hierarchical versions of the Network Time Protocol (NTP) and are used in this section as a benchmark. These three NTP-based hierarchical schemes were inspired by [14].

NTP-1: In this first scheme, each node arbitrarily selects a single neighbor which is one hop closer to the reference node than itself. Node Λ_i adjusted his clock by:

$$\tau_i = \frac{\Delta T_{ij} - \Delta T_{ji}}{2}.$$

We start with nodes that are one hop away from the reference node, move to nodes that are two hops away from it, etc.

NTP-2: This second scheme is similar to NTP-1, but in this case ΔT_{ij} and ΔT_{ji} are selected separately on each directed link. In other words, it tends to find the smallest possible queuing delays for each link. For example, if the queuing delays from node Λ_i to node Λ_k are (2, 3, 6, 5, 3, 4, 5, 6) and back from Λ_k to Λ_i (6, 5, 4, 6, 4, 5, 3, 7), it will select the minimal value on each direction separately, i.e., (2, 3) and calculate the offset based on this modified measurement.

NTP-3: This third scheme is a multi-parent scheme. Each node computes its clock offsets using the average of all its neighbors which are one hop closer to the reference node than itself. This can be interpreted as the CTP algorithm where only the parent nodes are used for calculations.

10.2 Network Topologies

In order to evaluate the results of the different algorithms, we need first to construct the network topology setup. The network topology we choose to implement is based on the random model of [47]. We start with a single root node (also called the "Reference Time Node") and restrict the hop distance of each node to the root to be at most a certain number of hops. The connectivity between the nodes in the network is randomly selected. The propagation delay of each edge is chosen once for both directions (assumed to be symmetric) of any existing edge based on the uniform distribution $U[0,10]$. The queuing delay of each edge is chosen as Erlang (or Gamma) distribution where the number of exponentials (α) and the mean time between events (θ) are randomly selected $U\left[1,10\right]$ and $U\left[0.1,1\right]$ respectively. The parameters are sampled once for each edge. The clock offsets with respect to the "Reference Time Node" are randomly selected based on a uniform distribution $U\left[-10,10\right]$. The offset of the reference node (Λ_1) is set to zero since it is assumed to be synchronized with the UTC.

As suggested by NTP in [30], eight round-trip packets are transmitted over each edge and ΔT_{ij} are measured based on these packets. Then, we have to pick up the best measurement among the eight (the one with the smallest transmission delay). To be compliant with the NTP message format, we suggest using four time stamps in each bilateral transmission. We constrained each node to have at least one neighbor.

In this numerical example, we naturally assumed that all the clocks run at the same speed, i.e., $\alpha_i = \alpha_j = 1, \forall i, j$ and that the offsets are time invariant.

In all the subsequent simulations, we consider a general network as depicted in Figure 10.1, where internal loops are allowed. This is important because the Least-Squares approach improves the estimates by imposing the global constraints for all the loops in the multihop network.

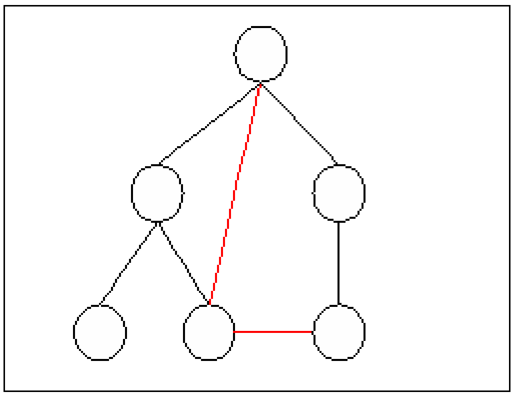


Figure 10.1. A general network with internal loops.

Three different networks are considered:

- Network 1: a 400 node network with relatively high connectivity of 1798 edges.
- Network 2: a 400 node network with 997 edges.
- Network 3: a 170 node network with 1200 edges.

10.3 Graphical Results

In this section, we compare CTP to the WLS and to the DKF algorithms in several interesting cases. Finally, we perform the recursive centralized Kalman Filter (with 50 measurement sets) and compare its performance to the centralized Least-Squares method and to the Sub-Optimal Algorithm (SOA) that neglects the off-diagonal terms of the inverse covariance matrix.

In Appendix C, one can find a performance comparison between the CTP algorithm and three hierarchical versions of NTP, and a convergence analysis of the decentralized version of CTP (see Figures C.1 to C.4). All those simulation results are similar to the work performed by O. Gurewitz et al. in [14] and were repeated in order to constitute the starting point of the subsequent results. The next step consists of implementing the Weighted Least-Squares algorithm in a decentralized manner and to compare the results to the decentralized CTP algorithm. In the WLS algorithm, we decide to take several different values of the weighting matrix R. Then, we will model the queuing delay according to this weighting matrix. For example, we can choose the queuing delay according to the Gamma distribution where the number of exponentials (α) is equal to the upper integer value of R and the mean time between events (θ) is equal to R. The other option is to take a normal distribution with zero mean and covariance matrix equal to the matrix R.

Figure 10.2 presents the comparison between the decentralized CTP and WLS algorithms for the case where the weighting matrix R is randomly distributed $U\left[0.1,12\right]$. The network topology is the same as before (400 nodes with 997 edges) and the queuing delay is randomly chosen according to the Gamma distribution relative to R.

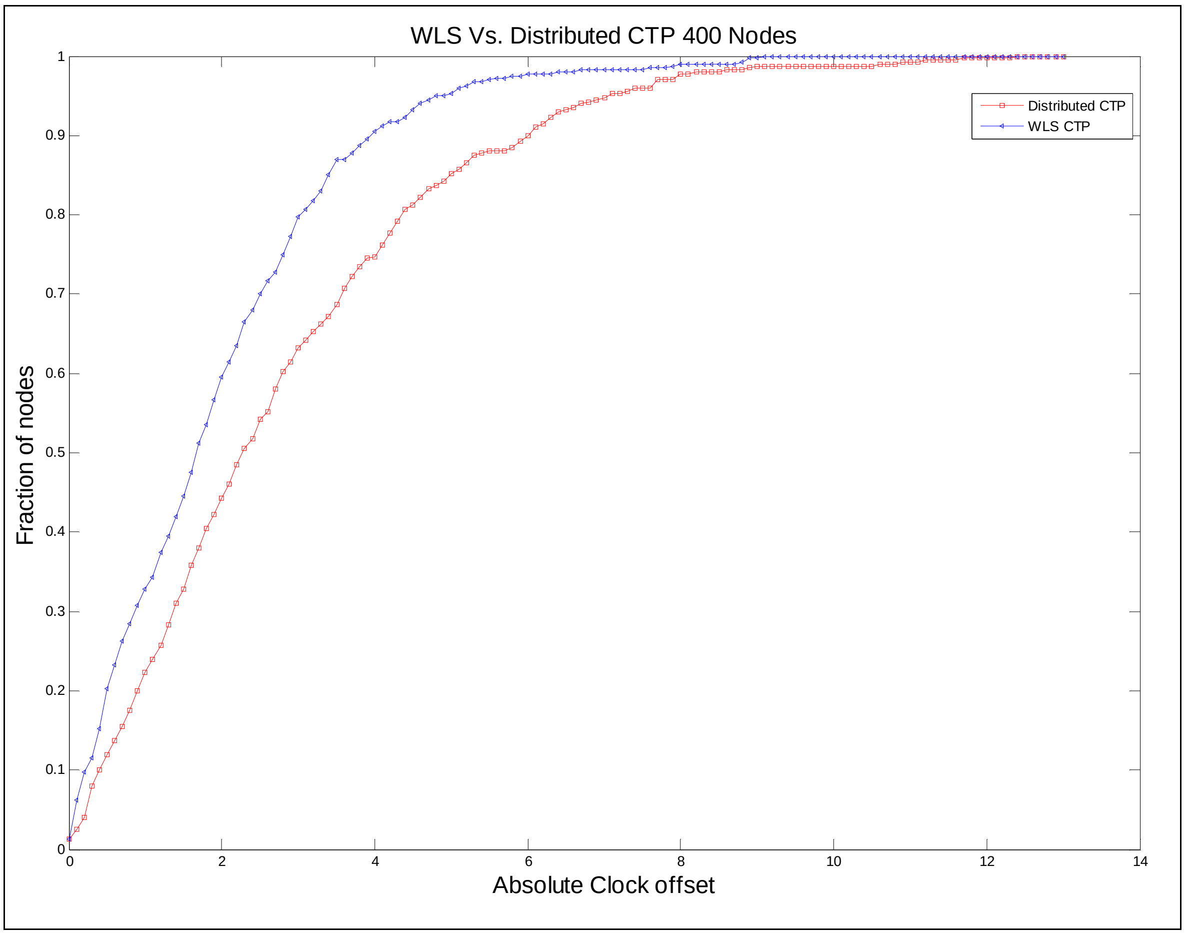


Figure 10.2. Comparison between the decentralized CTP and WLS algorithms in Network 2.

As expected, the WLS method outperforms the CTP (or LS) algorithm. The reason is that in the WLS version, each measurement is multiplied by a weight according to its accuracy that depends on the queuing delay. Since the weights are equal to the variance of the queuing delay; it outperforms the case where the weights are identically equal to one. This scenario is not very realistic and is more theoretic, because in practice we do not know the exact covariance matrix of the delay.

Unsurprisingly, if the weighting matrix R is taken as the identity matrix, we return to the Least-Squares case, equivalent as it is to the CTP algorithm. As we can see in the next figure, the graphs perfectly coincide.

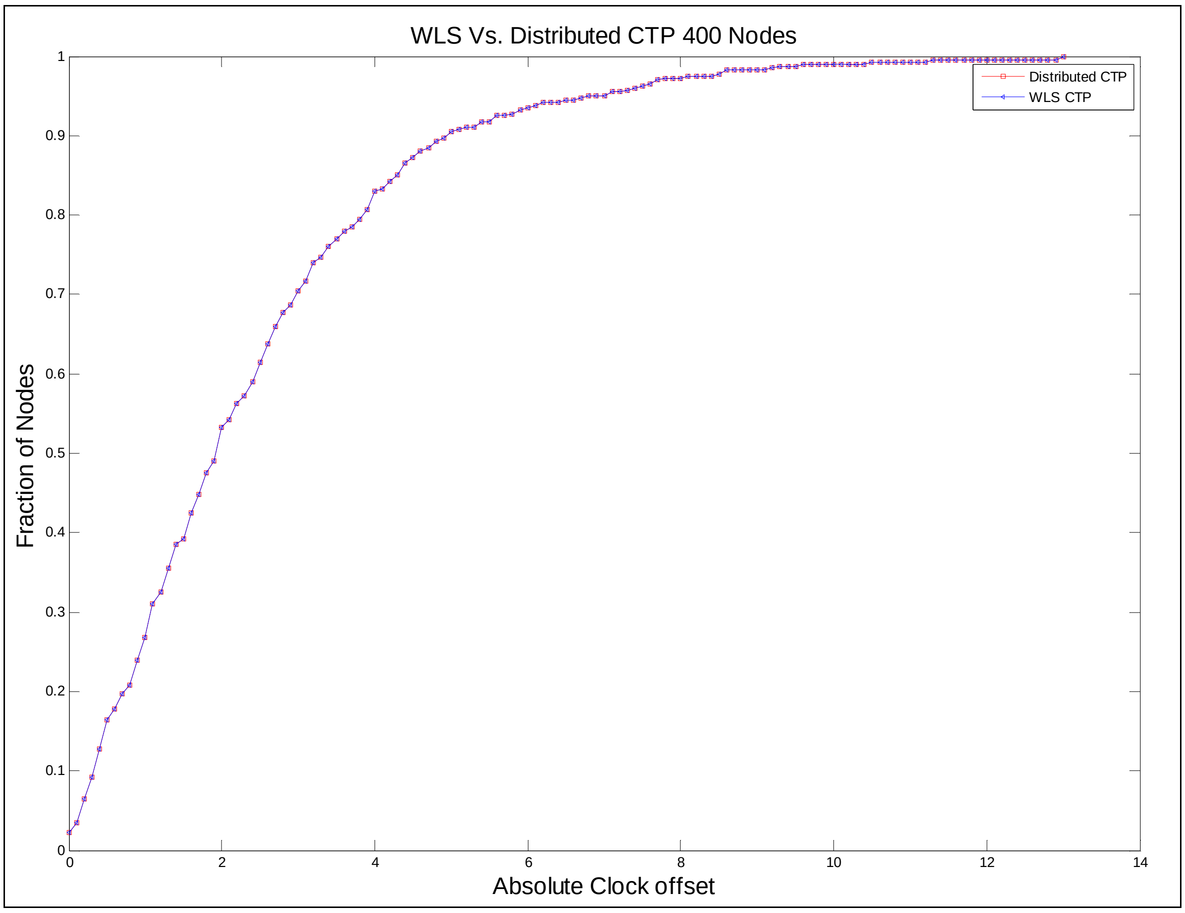


Figure 10.3. Comparison between the decentralized CTP and WLS algorithms in Network 2, with $R = I$.

The next scenario corresponds to the case where half the nodes have a certain value r_{ij} and the remaining have twice that value, i.e., half the nodes are smarter than the others. For example, we chose the values 4.5 and 9. In this case too, the WLS method is more accurate than the CTP algorithm as we can see in the next figure.

We note that in all the relevant figures, the curve describing the CTP algorithm is not exactly the same. Indeed, it slightly depends on the noise realization and each figure was plotted for a random realization. However, this fact does not compromise the comparison between the different algorithms. No matter what the noise realization is, the insight presented by the results remains correct.

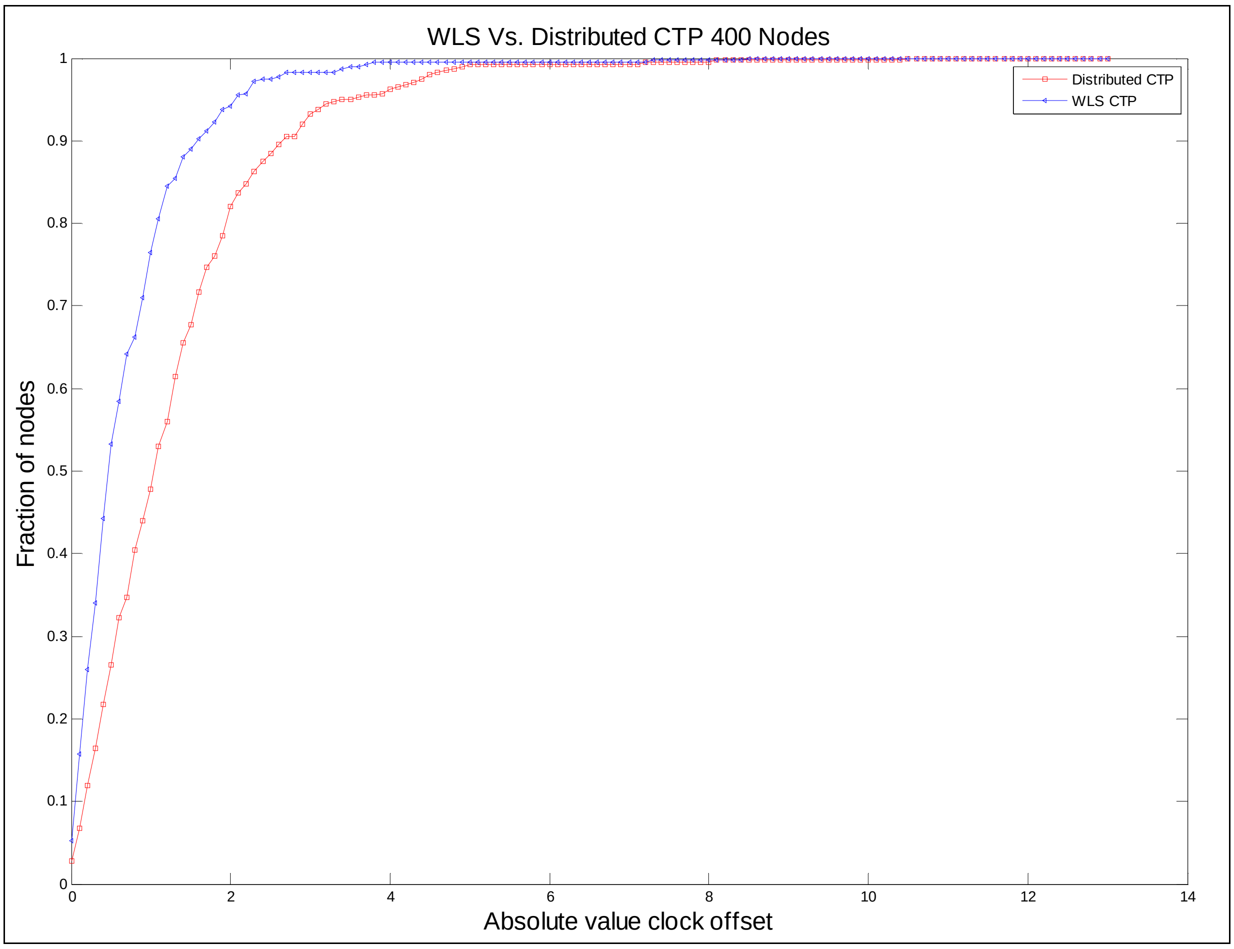


Figure 10.4. Comparison between the decentralized CTP and WLS algorithms in Network 2, with different r_{ij}.

As previously pointed out, we cannot know exactly the covariance matrix R. Hence, we check several cases, where instead of R, we may employ as an example R^2 or the matrix R plus an additive Gaussian Noise (with different variances). In other words, the matrix R is not known exactly, but we can use an approximation.

Figure 10.5 shows that the WLS method with the exact R gives the best results and the CTP algorithm (with $R = I$) achieves the worst results. Interestingly, the WLS algorithm with R^2 gives intermediate results (between LS and WLS). We can infer that even if the exact R is not employed, we can still outperform the results of the basic CTP protocol by using a good approximation of the matrix R. We will see in Figure 10.10 that if the approximation is not quite accurate, the regular LS outperforms the WLS algorithm.

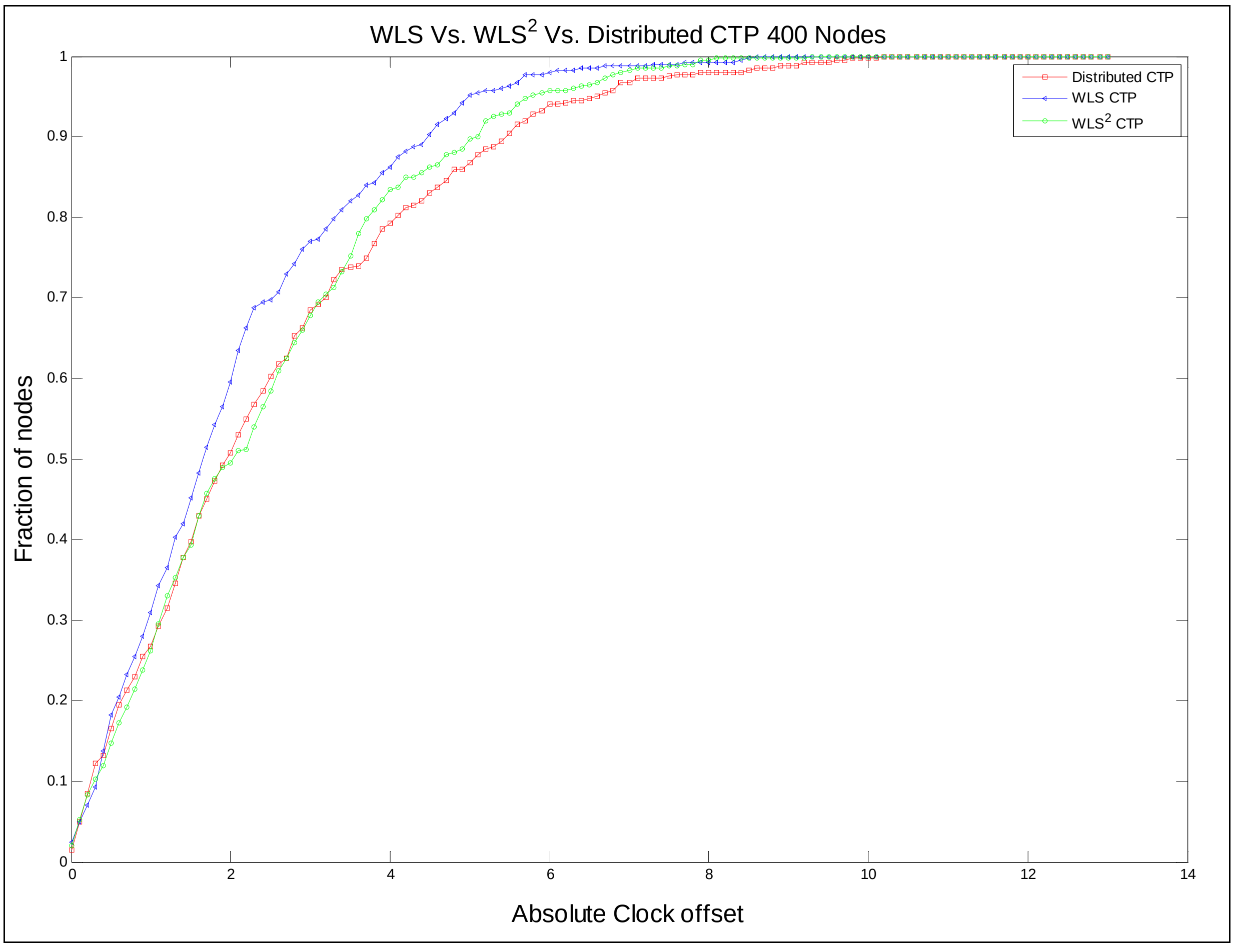


Figure 10.5. Comparison between the decentralized CTP and WLS algorithms (with R and R^2) in Network 2.

Next, let us analyze the robustness of the matrix R in the WLS algorithm. Since the exact R is not known, we implement the WLS algorithm with a weighting matrix equal to R plus an additive Gaussian noise with zero mean and two different standard deviations. We chose the following model:

$$\frac{1}{r_{ij}} + \tilde{n}_{ij}\,.$$

Here, $\tilde{n}_{ij}$ is a Gaussian noise with zero mean and standard deviation equal to 0.05 in the first case and to 0.2 in the second. The results are summarized in the next figure.

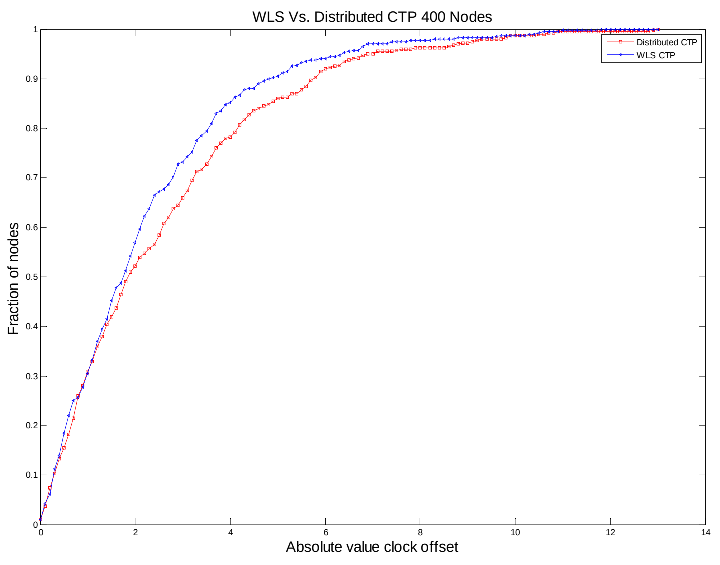


Figure 10.6a. Small noise variance

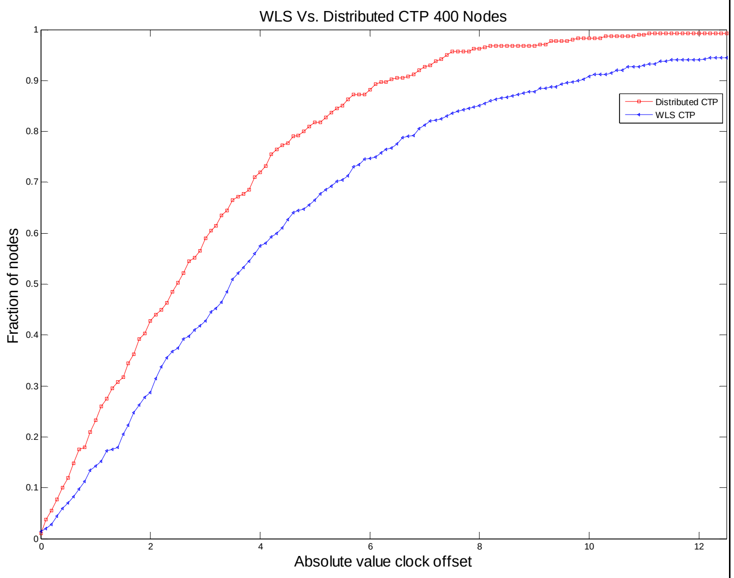


Figure 10.6b. Higher noise variance

Figure 10.6. Comparison between the decentralized CTP and WLS algorithms (with additive Gaussian noises in R) in Network 2.

As we can see from the previous figure, the results depend on the noise intensity. In the first case (Figure 10.6a), the variance of the Gaussian noise is relatively small and the WLS method outperforms the CTP algorithm. In the second case (Figure 10.6b), the variance noise is increased and as a consequence, the WLS is not anymore better than CTP. Therefore, we partially investigate the robustness properties of the exact covariance matrix R, and obtain as expected that if the intensity of the additive Gaussian noise is too high, the WLS method is not appropriate, whereas when the noise variance is relatively low, the WLS algorithm gives better results than CTP.

The next part of our numerical analysis is dedicated to the DKF algorithm. In the latter, we can incorporate some a-priori knowledge of the initial offsets and we compare it to the decentralized CTP algorithm. In Section 6, we showed that the DKF algorithm converges to the optimal solution obtained by performing the centralized Kalman Filter. In addition, the DKF algorithm has the same mathematical structure as the decentralized CTP with the incorporation of a supplementary term related to the a-priori knowledge. First, we check the DKF algorithm in the case where the initial covariance matrix P_0 is an infinite diagonal matrix and R is equal to the identity matrix. As expected, this case reduces to the basic LS case (CTP algorithm) and the graphs are plotted in Figure 10.7.

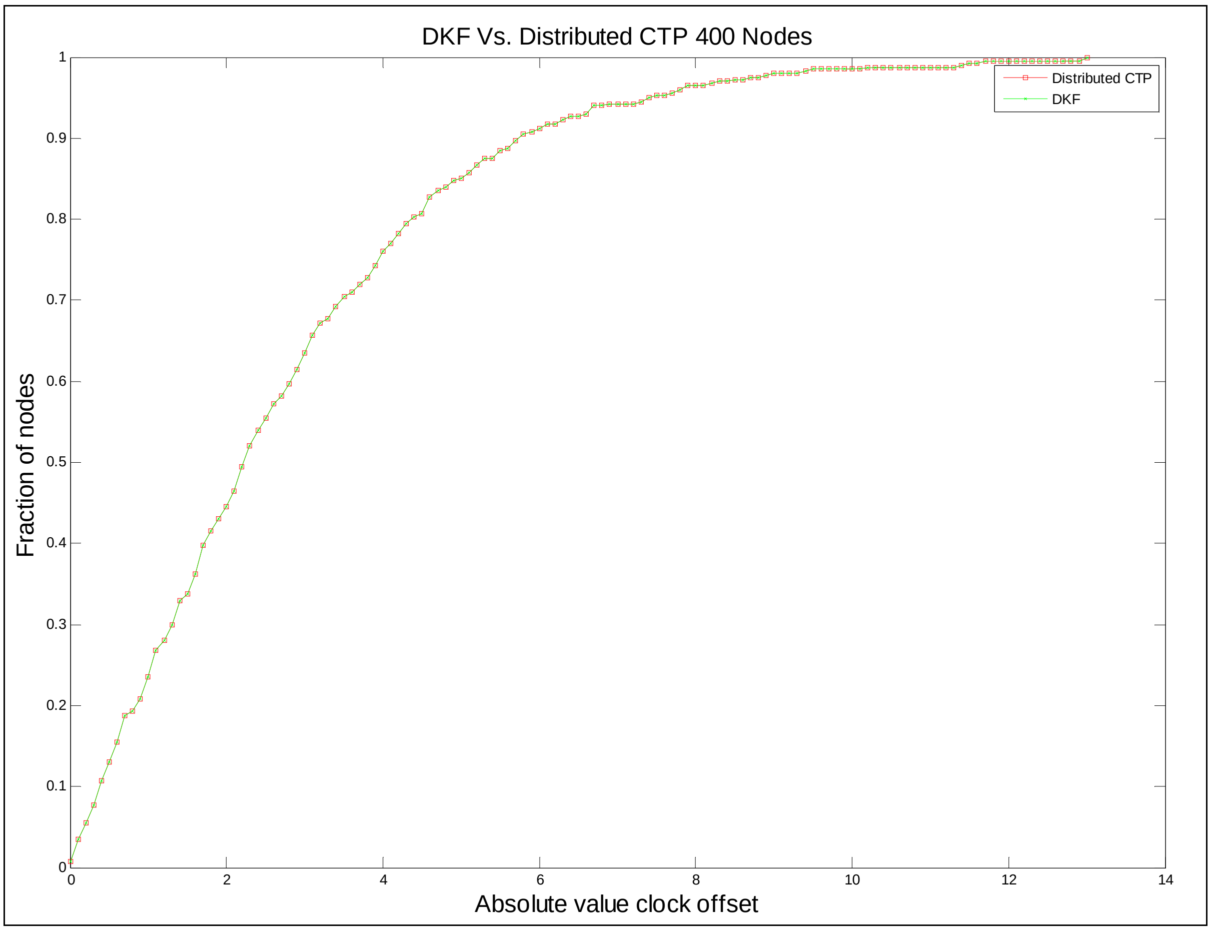


Figure 10.7. Comparison between the decentralized CTP and DKF algorithms (with $P_0^{-1} \to diag(\infty)$ and $R = I$) in Network 2.

The next scenario considered is related to the case where half the network nodes are smart (i.e., small initial variances in the main diagonal of P_0), and the remaining are unintelligent (i.e., bigger initial variances in the main diagonal of P_0). In all the cases, the initial covariance matrix P_0 is assumed to be diagonal and the initial offsets vector $\overline{x}_0$ is supposed to be equal to zero. According to the Kalman Filter requirements, the initial offsets have a Gaussian distribution:

$$\underline{x}(0) \sim N(\overline{x}_0, P_0).$$

We present the results in Figure 10.8 in the case where $R = I$ and P_0 is given by:

$$(P_0)_{ii} = \begin{cases} \sim U(0.01, 0.19)\,; & \textit{half the nodes} \\ \sim U(5,10)\,; & \textit{the remainder} \end{cases}.$$

The case where the matrix R takes different values is straightforward and the results are not presented here because we want to focus on the influence of the a-priori knowledge.

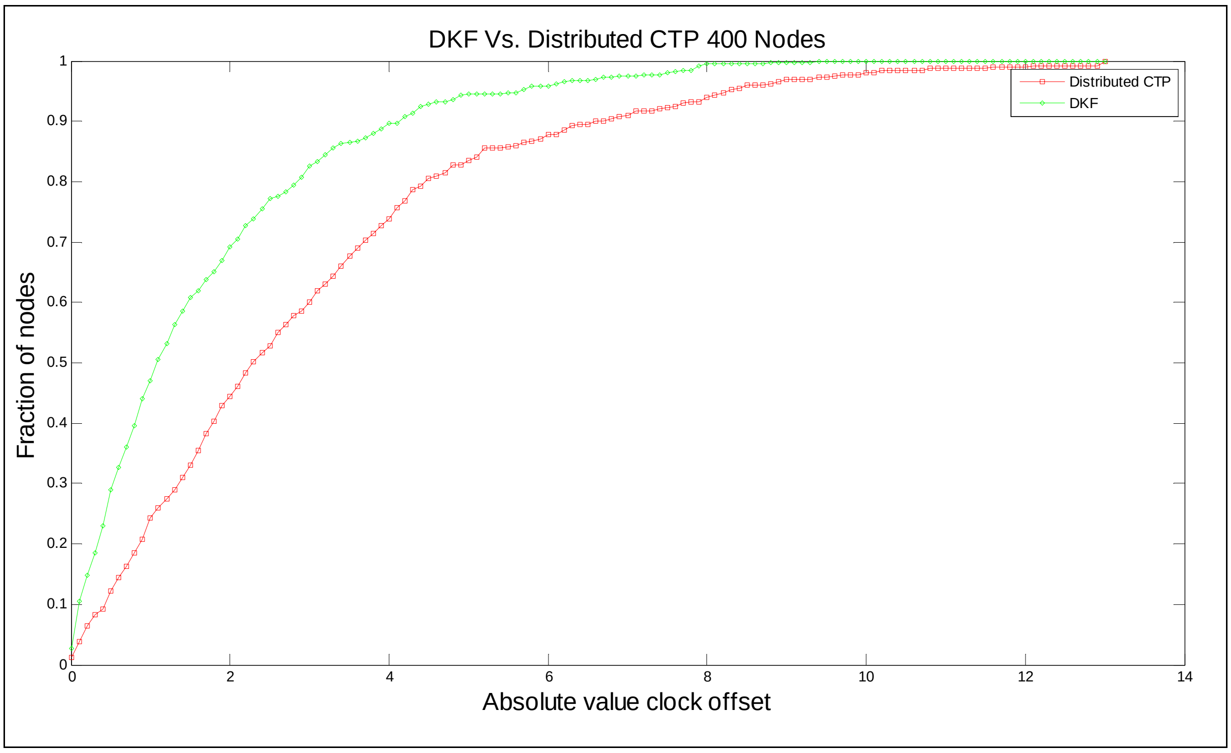


Figure 10.8. Comparison between the decentralized CTP and DKF algorithms (with different $(P_0)_{ii}$ and $R = I$) in Network 2.

The last case we analyze in the DKF context is the one where 10% of the nodes are perfectly synchronized to the UTC (through a GPS for example), and the remainder is not synchronized at all. Namely, for these arbitrary 40 nodes we take the initial variances very small (0.01) and the offsets equal to zero, and for the rest of the nodes, the variances tend to infinity and the offsets are randomly chosen according to a uniform distribution. The graphical comparison between the decentralized CTP algorithm and DKF is presented in Figure 10.9. As expected in this case too, the DKF algorithm outperforms the decentralized CTP method in terms of clock accuracy.

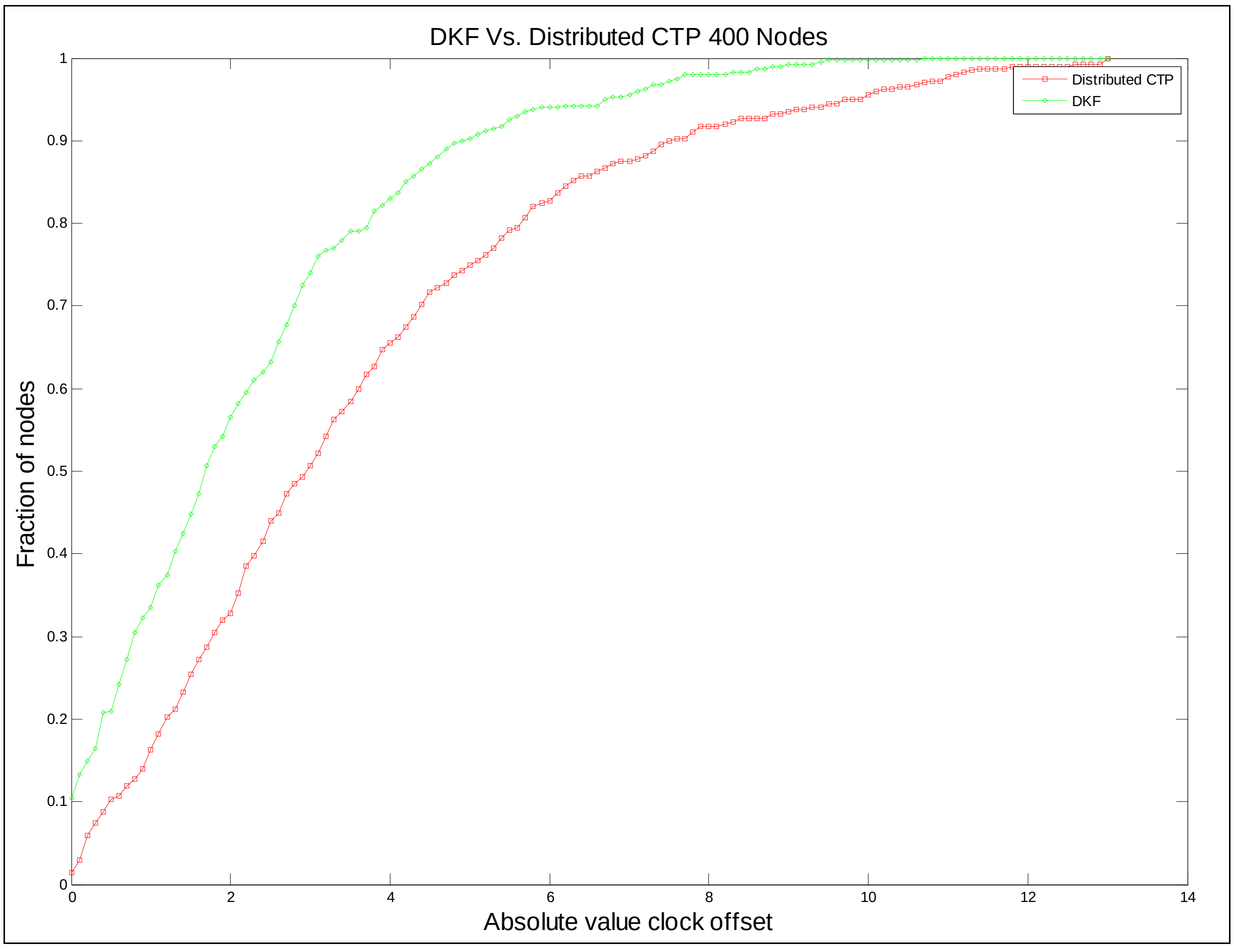


Figure 10.9. Comparison between the decentralized CTP and DKF algorithms (with 10% of nodes synchronized via GPS) in Network 2.

In all the previous simulations, the results we obtained agree with our expectations. Namely, the CTP algorithm is more accurate than all the NTP schemes and the corresponding decentralized version converges to the optimal centralized solution (see the proof in [14]). Moreover, the WLS algorithm is more accurate than the basic CTP (LS) method and the DKF outperforms the CTP algorithm in some suitable scenarios. These results are expected from the theory because in the WLS algorithm, we provide the best weighting factors to the measurements and in the DKF method, some a-priori knowledge is included in the right manner. Hence, in several appropriate scenarios (if some a-priori knowledge is available or the variance of the measurements can be approximated), the DKF and the WLS algorithms are preferable. As expected from the theory, this typical application shows that all the algorithms give satisfactory results. In many cases, the Kalman Filter is the best algorithm, followed by the Weighted Least-Squares and the regular Least-Squares methods and the NTP based protocols.

In this section, we are not comparing the convergence rate of the different algorithms and this can be also an important factor in the choice of the appropriate protocol.

The last part of this section is devoted to the comparison of the recursive Centralized Kalman Filter (CKF) algorithm to the Sub-Optimal Algorithm (SOA) that neglects the off-diagonal terms of the inverse covariance matrix (for more details, see Section 7.4). The latter is only a sub-optimal procedure and there is no need to know the parameter n. We check several values of n (the number of measurement sets) and P_0 (the initial covariance matrix). In this part, we consider the topology of Network 3 (170 nodes with 1200 edges). Our objective is to compare the accuracy of the estimated offsets obtained by both the optimal and sub-optimal algorithms and to compare the variances.

In this case, the queuing delay is randomized in accordance with the Kalman Filter assumptions, namely normally distributed with zero mean and covariance matrix R. The first situation we considered is the basic case where $P_0 = R = I$. The following figure presents the results obtained by applying both the optimal CKF method and the SOA for different values of n. As expected, in all the cases the optimal algorithm gives the best results. The sub-optimal algorithm gives relatively poor results but reduces the complexity and is not diverging. If the important criterion is the accuracy of the clock synchronization, it is obvious that the optimal CKF is preferable, but if the accuracy is less important than the complexity and the computation time, the sub-optimal algorithm can be valuable. We are also interested in comparing the variances of the different algorithms. We compute the following expression:

$$\frac{\left(P_{CKF}\right)_{ii} - \left(P_{SOA}\right)_{ii}}{\left(P_{CKF}\right)_{ii}} \qquad (10.3.1)$$

at each node over the network and present the results in a graphical form in Figure 10.11. As we can note from the figure, the variances of the CKF method are always bigger than the variances of the SOA algorithm and the normalized deviation is quite elevated (from 45% to 75 %). In other words, the sub-optimal algorithm underestimates the variances and consequently is not a good method. Since the variances are small, it will give a wrong estimation and will not correct the results. In brief, we can infer that the simple sub-optimal algorithm does not solve the problem efficiently and it is preferable to employ the Kalman Filter algorithm.

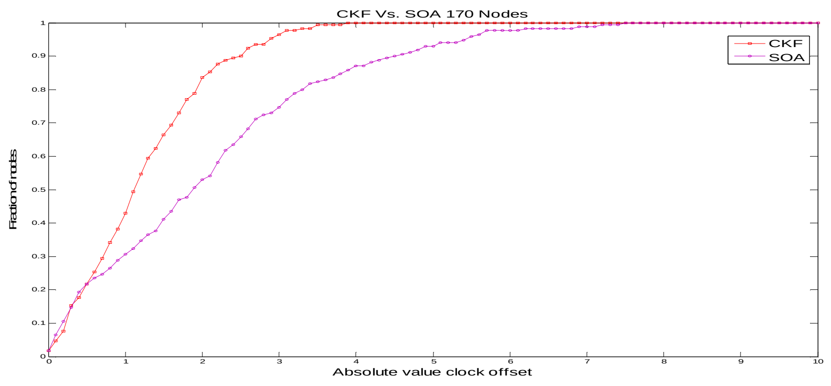
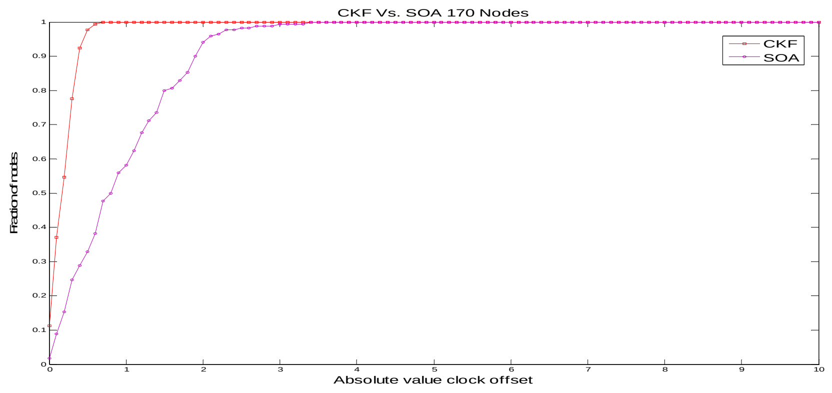
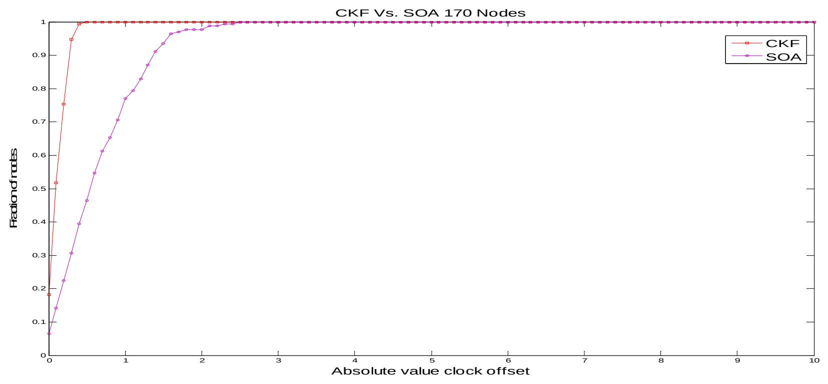
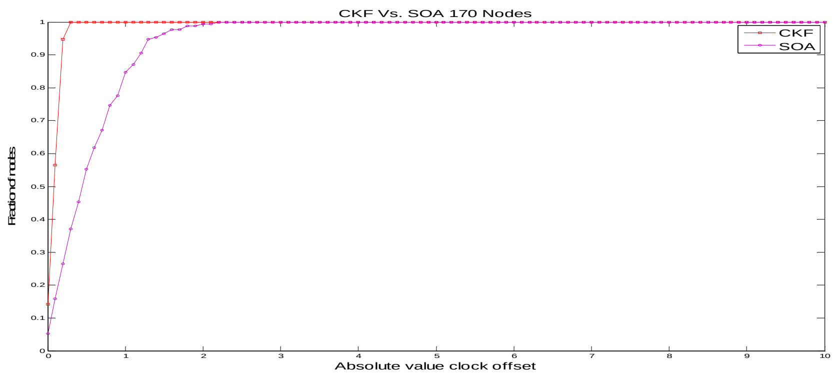
$n = 1$	
$n = 10$	
$n = 30$	
$n = 50$	

Figure 10.10. Comparison between CKF and SOA (with $P_0 = R = I$) in Network 3 for different values of n.

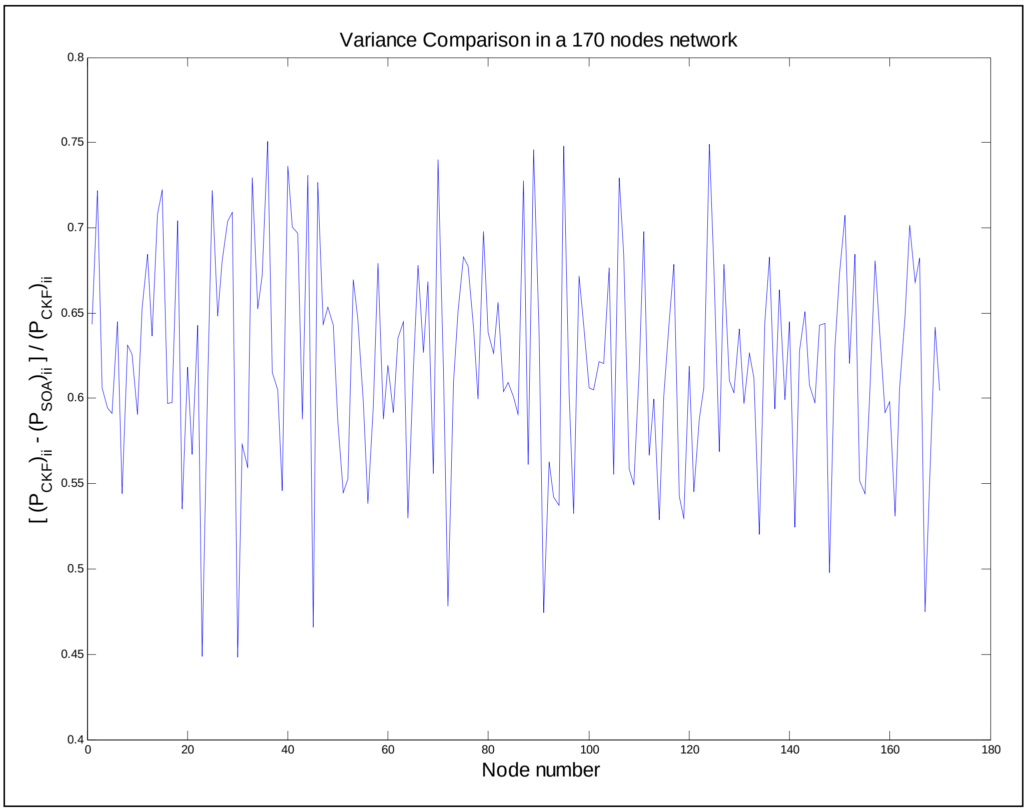


Figure 10.11. Variance comparison between CKF and SOA (with $P_0 = R = I$) in Network 3.

The next step extends the preceding analysis to a more general framework where the measurement covariance matrix R is uniformly distributed and the queuing delays are normally distributed with zero mean and covariance matrix R. In other words:

$$R \sim U\left[0.01,12\right]$$
$$Q_{delay} \sim N\left(0,R\right)$$

In addition, we consider that 10% of the nodes are perfectly synchronized to the UTC (through a GPS for example), and the remainder is not synchronized at all. Namely, for these arbitrary 17 nodes we will take the initial variances very small (0.01) and the offsets equal to zero, and for the rest of the nodes, the variances tend to infinity and the offsets are randomly chosen according to a uniform distribution. In this analysis, we also compare the results to the Centralized Least-Squares (CLS) algorithm.

In fact, as we previously mentioned, the SOA gives relatively poor results in comparison to the CKF algorithm. We thus want to determine if the basic LS algorithm is more accurate than SOA. In other words, is it preferable to totally neglect the a-priori knowledge (i.e., take $P_0^{-1} = 0$) or to consider this a-priori knowledge and then to neglect a part of its influence?

Figure 10.12 presents the results for the offsets obtained by applying the optimal CKF method, the SOA and the CLS algorithms for the same different values of n as in the previous case. Figure 10.13 compares the variances between CKF and SOA. We can draw the same conclusions as in the previous case, with the exception of a normalized variance dispersion between 4% to 78%.

Moreover, we obtained that the sub-optimal algorithm is even worse (in terms of clock accuracy) than the basic centralized Least-Squares method (that does not take into account the initial covariance matrix). From the two subsequent figures, we may conclude that despite the fact that the sub-optimal algorithm is valuable for complexity and computation time reasons, the results are relatively far from the optimal ones. Hence, we do not consider it as an efficient algorithm to solve the time network synchronization problem considered in this thesis. In order to solve this problem in an efficient way, we propose several alternatives. The first option is to apply the recursive version of the DKF algorithm that we developed in this report because it leads to the optimal solution and requires only local communication. A second alternative is to investigate another sub-optimal solution, such as the method proposed in [17]. This recent work presents a distributed Kalman Filter that estimates sparsely connected, large scale systems (L-banded matrix algorithm). Actually, a smart approximation of the covariance matrix is employed and the authors prove that the solution converges to the global Kalman Filter as the number of bands increases.

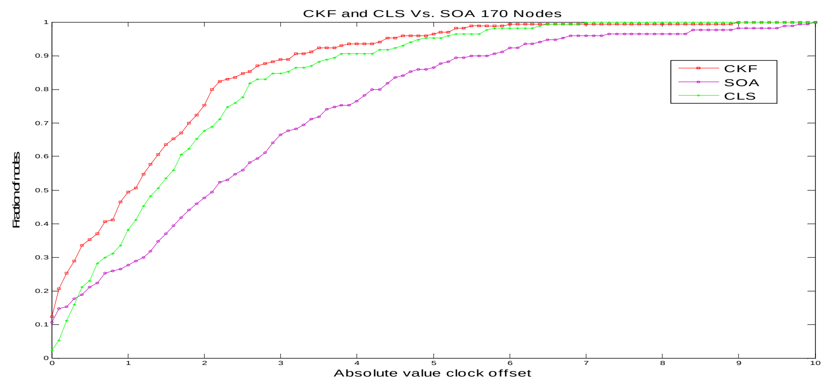
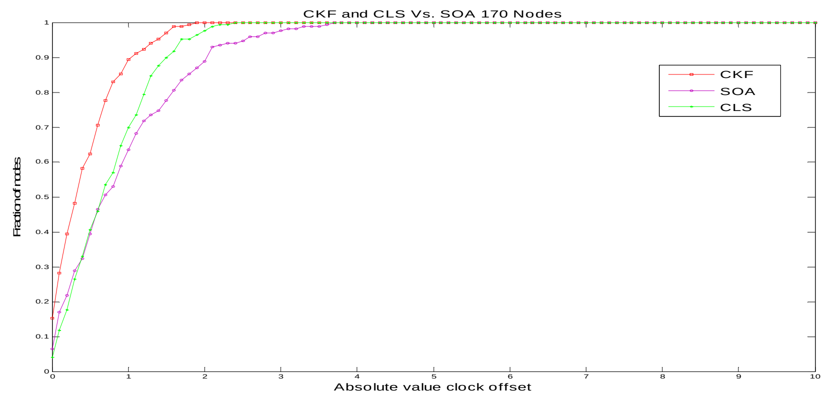
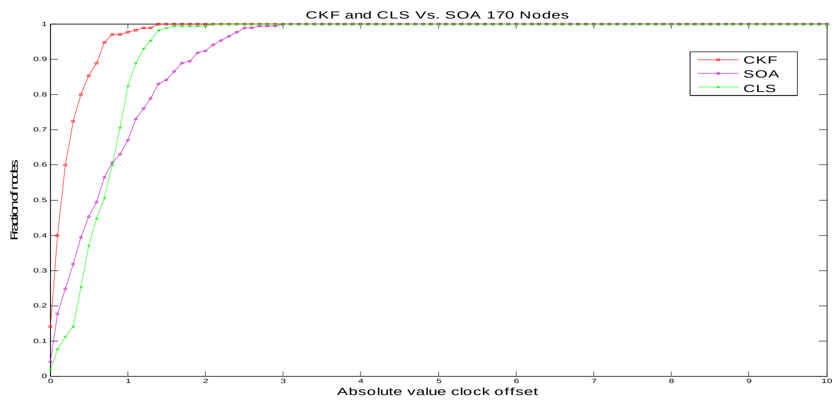
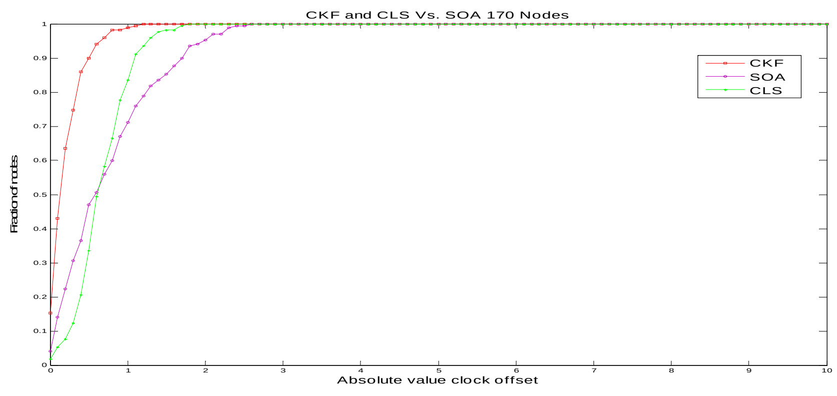
$n = 1$	
$n = 10$	
$n = 30$	
$n = 50$	

Figure 10.12. Comparison between CKF, SOA and CLS (with $R \sim U[0.01,12]$ and $P_0 \neq I$) in Network 3 for different values of n.

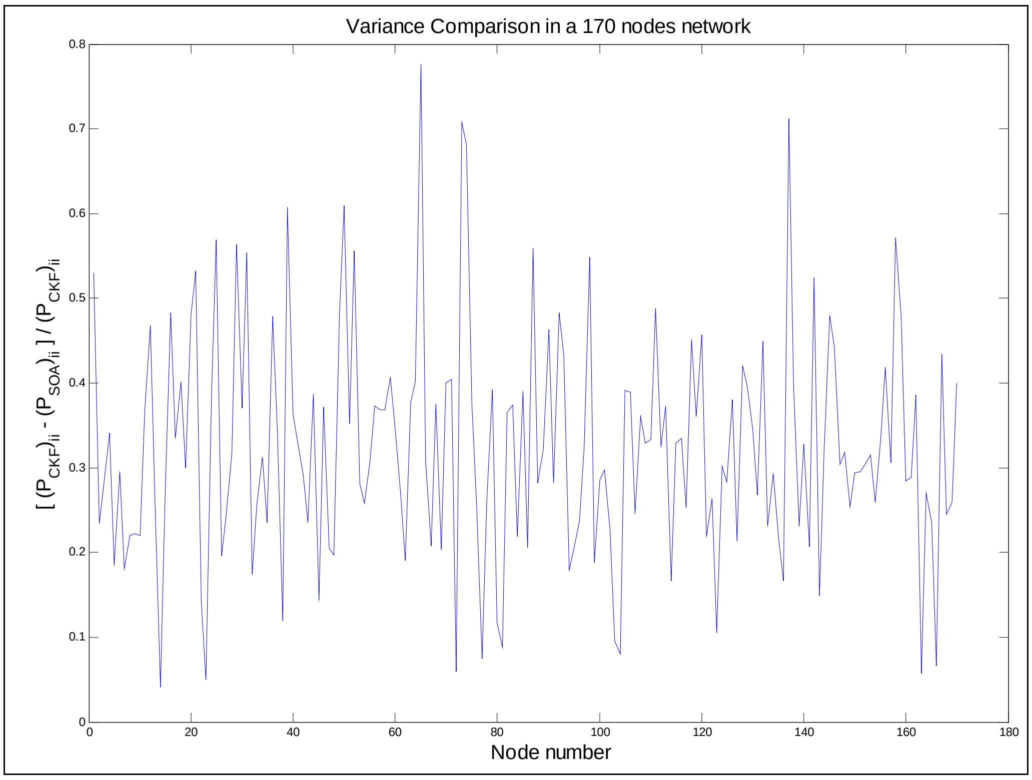


Figure 10.13. Variance comparison between CKF and SOA (with $R \sim U\left[0.01, 12\right]$ and $P_0 \neq I$) in Network 3.

In summary, we performed several clock synchronization algorithms over different network topologies and compare the results. We obtained that the algorithms based on the Kalman Filter framework give the best results in terms of clock accuracy. The basic Least-Squares algorithm (equivalent to CTP in [14]) outperforms the three hierarchical NTP schemes considered and we have extended the results to some more general situations. We can provide different weights to the measurements according to their accuracy and incorporate a-priori knowledge of the problem. As seen in the simulation results, the decentralized Kalman Filter algorithm constitutes the most appropriate and the most general method for clock synchronization among the proposed algorithms. The clock accuracy is the most precise, it requires only local communication between neighbors and the complexity is not increased. In the last part of this section, we compared the centralized Kalman Filter solution to a simple sub-optimal algorithm that neglects the off-diagonal terms of the inverse covariance matrix P_0^{-1}. As expected, the optimal algorithm gives improved results with respect to the sub-optimal method, and this is why we needed to develop a decentralized version of the Kalman Filter.

Simulation results on the skew estimation problem are not considered in this thesis. As a future work, one may envisage the comparison of the different algorithms of Section 9.
In the next section, we present the conclusions and several directions for future research.

11. Conclusion and Future Work

We developed several decentralized algorithms for estimating the offset at each node in a network with respect to the reference time, using a Kalman Filter framework. These algorithms can be either synchronous or asynchronous, some of which being recursive. In addition, we showed that these decentralized filtering algorithms converge to the optimal centralized solution. The essential characteristic of these algorithms is their decentralized nature; each node can estimate its clock offset only by exchanging packets with its one-hop neighbors. We considered the case where all the clocks run at the same speed, i.e., $\alpha_i = \alpha_j = 1$ (there is no skew) as well as the case where $\alpha_i \neq \alpha_j$. In the latter case, we developed several different estimation algorithms: one for estimating the clock skews (without knowledge of the offsets) and one for estimating the clock offsets. In practice, we will treat skew synchronization and offset synchronization on different time scales. We obtained that the offset and the skew estimation problems reduce to the same mathematical setup. Hence, we developed several decentralized algorithms for clock synchronization that can treat general clocks with both offsets and skews. We investigate two different approaches. In the first, time offsets and rate offsets are estimated simultaneously, whereas the second is based on a separate estimation of both parameters.

In summary, we extended the existing Least-Squares results using the Kalman Filter framework. Namely, we can give different weights to the measurements according to their accuracy and include a-priori knowledge. The main algorithm is both decentralized (requires only local broadcasts), recursive (works in real-time applications) and converges to the optimal centralized solution. As expected, under some appropriate assumptions, we showed that our optimal estimated offsets correspond to the Maximum A-Posteriori estimator (or to the Maximum-Likelihood estimator in the statistical Least-Squares case).

We also considered several extensions to the basic case, like the incorporation of a discount factor and the exposure to temporary communication failures. We tested the different algorithms on typical networks and compared the results with several versions of the Network Time Protocol. In most of the cases, the proposed algorithms outperform the NTP schemes and the LS method. In addition, we compared the recursive CKF algorithm to a simple sub-optimal algorithm that neglects the off-diagonal terms of the inverse covariance matrix.

Several directions may continue the work performed in this research thesis. Among these, we suggest the following:

- The general clock problem that includes both offset and skew was not solved in an optimal way. It can be interesting to find a decentralized optimal algorithm that converges to the optimal solution and makes use of all the measurements.

- There exist several approaches to performing heuristics that approximate the Kalman Filter solution. For example, [17] presents a distributed Kalman Filter to estimate sparsely connected, large scale systems (L-banded matrix algorithm). It can be worthwhile to use the same approach to solve the clock synchronization problem.

- In this thesis, we gave more importance to theoretical analysis than to numerical simulations. Extending the simulation results to the case where a general clock is considered (including skew estimation), and implementing the algorithm with temporary communication failures could provide fruitful. Moreover, it can be valuable to compare the convergence rate of the different algorithms in a numerical analysis.

- In the general state space problem that includes process noise, we did not achieve an optimal decentralized algorithm. A further possible direction for future research may be to solve this problem optimally.

- So far, the offsets were considered to be time-invariant and the network topology static. The next step involves solving the same problem when these assumptions are relaxed, that is, dynamic topologies with time-varying offsets. In the time-varying case, the advantageous properties of the Kalman Filter structure can be exploited.

- As we previously noted, the problem that we considered in this thesis can be viewed as a general problem related to distributed estimation based on relative measurements in sensor networks. The time synchronization and the sensor localization problems are only special cases. An additional question of interest is to consider a general framework in order to estimate the distances between several cooperative agents. An interesting example is that of a group of aircraft flying in formation, where we seek to estimate the distance to the leader. This problem seems to be very interesting albeit more difficult due to its non-linear nature.

References

[1] C. Aitken (1935), "On Least-Squares and linear combinations of observations", *Proceedings of the Royal Society of Edinburgh*, 55, 42-48.

[2] P. Alriksson and A. Rantzer, "Distributed Kalman filtering using weighted averaging", *Proceedings of the 17th International Symposium on Mathematical Theory of Networks and Systems*, Kyoto, Japan, 2006.

[3] P. Barooah, and J. P. Hespanha, "Distributed estimation from relative measurements in sensor networks," *Proceedings of the 3rd Int. Conf. on Intelligent Sensing and Information Processing*, 2005.

[4] T. M. Berg and H. F. Durrant-Whyte, "Distributed and decentralized estimation", *Proceedings of the Singapore International Conference on Intelligent Control and Instrumentation*, vol. 2, 1992.

[5] D. P. Bertsekas and J. N. Tsitsiklis, Parallel and distributed computations, *Prentice Hall, Englewood Cliffs*, NJ, 1989.

[6] R. Carli, A. Chiuso, L. Schenato and S. Zampieri, "Distributed Kalman filtering based on consensus strategies", *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 4, May 2008.

[7] W. H. Chung and J. L. Speyer, "A general framework for decentralized estimation", *Proceedings of the American Control Conference*, Seattle, Washington, June 1995.

[8] Mark Coates, "Distributed particle filters for sensor networks", *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks*, Berkeley, California, USA, 2004.

[9] J. Elson, L. Girod and D. Estrin, "Fine grained network time synchronization using reference broadcast", *Proceedings of the 5th Symposium on Operating Systems Design and Implementation*, pp. 147-163, Boston, Dec. 2002.

[10] J. Elson and K. Romer, "Wireless sensor networks: A new regime for time synchronization", Tech. Rep., UCLA, July 2002.

[11] P. Ferguson and J. How, "Decentralized estimation algorithms for formation flying spacecraft", *AIAA Guidance, Navigation and Control Conference*, Austin, Texas, Aug. 2003.

[12] A. Giridhar and P. R. Kumar, "Distributed clock synchronization over wireless networks: algorithms and analysis", *Proceeding of the 45th IEEE Conference on Decision and Control*, San Diego, Dec. 2006.

[13] G. C. Goodwin and K. S. Sin, "Adaptive filtering prediction and control", *Prentice Hall, Englewood Cliffs*, NJ, 1984 (chapter 3).

[14] O. Gurewitz, I. Cidon and M. Sidi, “Network time synchronization using clock offset optimization”, *IEEE International Conference on Network Protocols (ICNP)*, pp. 212-221, Atlanta, GA, November 2003.

[15] H. R. Hashemipour, S. Roy and A. J. Laub, "Decentralized structures for parallel Kalman filtering", *IEEE Trans. on Automatic Control*, vol. 33, no. 1, pp. 88-94, Jan. 1988.

[16] R. A. Horn and C. R. Johnson, Matrix analysis, Cambridge, UK: *Cambridge University Press*, 1991 (pp.344-347).

[17] U. A. Khan and J. M. F. Moura, "Distributing the Kalman Filter for large-scale systems", *IEEE Trans. on Signal Processing*, 2008.

[18] R. E. Kalman, "A New approach to linear filtering and prediction problems", *Transactions of the ASME*, vol. 82 (Series D), pp. 35-45, 1960.

[19] E. D. Kaplan, editor, "Understanding GPS: principles and applications", *Artech House*, 1996.

[20] E. D. Kaplan, editor, "Adaptive Kalman Filter for time synchronization over packet-switched networks", *2nd International Conference on Communication Systems Software and Middleware (COMSWARE)*, Jan. 2007.

[21] R. Karp, J. Elson, D. Estrin and S. Shenker, “Optimal and global time synchronization in sensornets”, *Center for Embedded Networked Sensing,* University of California, Los Angeles, Tech. Rep., April 2003.

[22] C. T. Kelley, Iterative methods for linear and nonlinear equations, *SIAM*, Philadelphia, PA, 1995.

[23] L. Lamport, "Time, clocks and the ordering of events in a distributed system", *Communications of the ACM*, 21(4):558-565, July 1978.

[24] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk and J. Anderson, "Wireless sensor networks for habitat monitoring", *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, ACM press, pp. 88-97, 2002.

[25] J. Mannermaa, K. Kalliomaki, T. Mansten and S. Turunen, "Timing performance of various GPS receivers", *Proceedings of the 1999 Joint Meeting of the European Frequency and Time Forum and the IEEE International Frequency Control Symposium*, pp. 287-290, April 1999.

[26] M. Maroti, G. Simon, B. Kusy and A. Ledeczi, "The flooding time synchronization protocol", *Proceedings of the 2ndInternational Conference on Embedded Networked Sensor Systems*, pp. 39-49, Baltimore, MD, USA, Nov. 2004.

[27] P. S. Maybeck, Stochastic models, estimation, and control, New York: *Academic Press, Inc*, 1979. Republished, Arlington, VA: Navtech, 1994.

[28] D. L. Mills, “Internet time synchronization: the network time protocol”, *IEEE Trans. Commun*., vol. COM-39, no. 10, pp. 1482-1493, Oct. 1991.

[29] D. L. Mills, “Improved algorithms for synchronizing computer network clocks”, *IEEE/ACM Trans. Netw.*, vol. 3, no. 3, pp. 245-254, June 1995.

[30] D. L. Mills, Network Time Protocol (version 3) specification, implementation and analysis, *Network Working Group Report*. Univ. Delaware, RFC-1305, pp. 113, 1992.

[31] M. Mock, R. Frings, E. Nett and S. Trikaliotis, "Continuous clock synchronization in wireless real-time applications", *The 19th IEEE Symposium on Reliable Distributed Systems (SRDS)*, pp. 125-133, Washington – Brussels – Tokyo, Oct. 2000.

[32] E. M. Nebot, M. Bozorg and H. Durrant-Whyte, "Decentralized architecture for asynchronous sensors", *Autonomous Robots 6*, pp. 147-164, April 1999.

[33] K. Plarre and P. R. Kumar, "Object tracking by scattered directional sensors", *Proceedings of the 44th IEEE Conference on Decision and Control*, pp. 3123-3128, Seville, Spain, Dec. 2005.

[34] R. Olfati-Saber, "Distributed Kalman Filter with embedded consensus filters", *Proceedings of the 44th IEEE Conference on Decision and Control*, and *European Control Conference*, Dec. 2005.

[35] R. Olfati-Saber, J. A. Fax and R. M. Murray, "Consensus and cooperation in networked multi-agent systems", *Proceedings of the IEEE*, vol. 95, pp. 215-233, Jan. 2007.

[36] B. S. Rao and H. F. Durant-Whyte, "Fully decentralized algorithm for multisensor Kalman filtering", *IEEE Proceedings-D,* Vol. 138, No. 5, September 1991.

[37] K. Romer, "Time synchronization in ad hoc networks", *ACM Symposium on Mobile Ad Hoc Networking and Computing (Mobihoc)*, Long Beach, CA, Oct. 2001.

[38] S. I. Roumeliotis and G. A. Bekey, "Collective localization: A distributed Kalman Filter approach to localization of groups of mobile robots", *Proceedings of the IEEE International Conference on Robotics and Automation*, San Francisco, CA, April 2000.

[39] S. Roy and R. A. Iltis, "Decentralized linear estimation in correlated measurement noise*", IEEE Transactions on Aerospace and Electronic Systems*, July 1991.

[40] A. H. Sayed and T. Kailath", A state-space approach to adaptive RLS filtering", *IEEE Signal Processing Magazine*, July 1994.

[41] R. Solis, V. Borkar, and P. R. Kumar, “A new distributed time synchronization protocol for multihop wireless networks”, *Proceeding of the 45th IEEE Conference on Decision and Control*, pp. 2734-2739, San Diego, Dec. 13-15 2006.

[42] D. P. Spanos and R. M. Murray, "Distributed sensor fusion using dynamic consensus", *Proceedings of the 16th IFAC World Congress*, July 2005.

[43] J. Stoer and R. Burlirsch, Introduction to numerical analysis, *3rd Ed. Springer Verlag*, New York, 2002.

[44] J. D. Wolfe and J. L. Speyer, "A low-power filtering scheme for distributed sensor networks", *Proceedings of the 42nd IEEE Conference on Decision and Control*, Maui, Hawaii USA, Dec. 2003.

[45] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging", *Systems and Control Letters*, vol. 53, pp. 65-78, 2004.

[46] D. Yu, Y. Kim and S. Hwang, "A stable estimation model for time synchronization on the Internet using Kalman filtering", *Embedded and Ubiquitous Computing*, pp. 931-941, July 2004.

[47] E. W. Zegura, K. Calvert, and S. Bhattacharjee, “How to model an internetwork”, *Proceedings IEEE INFOCOM*, pp.594-602, San Francisco, CA, 1996.

[48] RFC1769 "Simple Network Time Protocol", RFC- Editor, www.rfc-editor.org.

Appendix A – Equivalence between KF and LS

We intend to prove the following general theorem about the equivalence between the Kalman Filter and the Least-Squares problem.

Assuming the Gaussian linear (time varying) state space model given by:

$$\begin{cases} \underline{x}(n+1) = \Phi(n)\underline{x}(n) + G(n)\underline{w}(n) & \underline{x}(0) \sim N(\overline{x}_0, P_o) \\ \underline{y}(n) = H(n)\underline{x}(n) + \underline{v}(n) & \underline{w}(n) \sim N(0, Q(n)); \quad \underline{v}(n) \sim N(0, R(n)) \end{cases}$$

Let us assume that the measurement noise $\underline{v}(n)$ and the system noise $\underline{w}(n)$ are white, uncorrelated and statistically independent of x_0.

We denote the state vector that maximizes the Maximum A-Posteriori (MAP) probability by $\underline{x}_{MAP} = \left(\underline{x}_0^{(n)}, \ldots, \underline{x}_k^{(n)}\right)^T$. Let us recall that for the Gaussian case, this is equivalent to the MMSE estimator, i.e., the state vector obtained by applying the Kalman Filter ($\hat{\underline{x}}(n \mid n)$).

Additionally, consider the following deterministic optimization problem:

$$\min_{\{\underline{x}_k\},\{\underline{w}_k\}} \left\{ \begin{array}{l} J_n = \frac{1}{2}\left(\underline{x}(0) - \overline{x}_0\right)^T P_0^{-1}\left(\underline{x}(0) - \overline{x}_0\right) + \frac{1}{2}\sum_{k=0}^{n-1}(\underline{w}_k)^T Q_k^{-1}(\underline{w}_k) + \\ + \frac{1}{2}\sum_{k=0}^{n}(\underline{y}_k - H_k\underline{x}_k)^T R_k^{-1}(\underline{y}_k - H_k\underline{x}_k) \end{array} \right\} \tag{A.1}$$
$$s.t. \quad \underline{x}_{k+1} = \Phi_k\underline{x}_k + G_k\underline{w}_k\,, \; k = 0, \ldots, n-1$$

Here, $\overline{x}_0$ and $\left\{\underline{y}_k\right\}_{k=0}^{n}$ are given vectors, and P_0, R_k^{-1}, Q_k^{-1} are symmetric positive-definite matrices.

The previous constrained optimization problem is by definition a Least-Squares problem. Let us denote its optimal solution by $\underline{x}_{LS}$.

Theorem

The minimizing solution of the above LS problem is equal to the MAP estimator (and to the MMSE solution), i.e., under the above conditions:

$$\boxed{\underline{x}_{LS} = \underline{x}_{MAP} = \underline{x}_{MMSE}} \tag{A.2}$$

Proof

Let us write the expression of $P(\underline{x}_0,...,\underline{x}_n,\underline{y}_0,...,\underline{y}_n)$:

$$P(\underline{x}_0,...,\underline{x}_n,\underline{y}_0,...,\underline{y}_n) = P(\underline{x}_0 = \overline{x}_0, \underline{x}_{i+1} - \Phi_i \underline{x}_i = G_i w_i, \underline{y}_j - H_j \underline{x}_j = \underline{v}_j, i = 0,...,n-1, j = 0,...,n)$$

Without loss of generality, we can assume that $G_k = I$. Recalling that each term is Gaussian and that all the terms are independent, we can obtain:

$$P(\underline{x}_0,...,\underline{x}_n,\underline{y}_0,...,\underline{y}_n) = const \cdot \exp\left(-\frac{1}{2}\left(\underline{x}_0 - \overline{x}_0\right)^T P_0^{-1}\left(\underline{x}_0 - \overline{x}_0\right)\right) \cdot$$
$$\cdot \prod_{i=0}^{n-1} \exp\left(-\frac{1}{2}\left(\underline{x}_i - \Phi_i \underline{x}_i\right)^T Q_i^{-1}\left(\underline{x}_i - \Phi_i \underline{x}_i\right)\right) \cdot \prod_{j=0}^{n} \exp\left(-\frac{1}{2}(\underline{y}_j - H_j \underline{x}_j)^T R_j^{-1}(\underline{y}_j - H_j \underline{x}_j)\right)$$

On the other hand, we can compute the MAP estimator:

$$\underline{x}_{MAP} = \arg\max\left\{P\left(\left(\underline{x}_0,...,\underline{x}_n\right)\middle|\left(\underline{y}_0,...,\underline{y}_n\right)\right)\right\} = \arg\max\left\{\frac{P\left(\underline{x}_0,...,\underline{x}_n,\underline{y}_0,...,\underline{y}_n\right)}{P\left(\underline{y}_0,...,\underline{y}_n\right)}\right\} =$$
$$= \arg\max\left\{P\left(\underline{x}_0,...,\underline{x}_n,\underline{y}_0,...,\underline{y}_n\right)\right\} = \arg\max\left\{\log P\left(\underline{x}_0,...,\underline{x}_n,\underline{y}_0,...,\underline{y}_n\right)\right\} =$$
$$= \arg\min\left\{J_n\right\} = \underline{x}_{LS}$$

■

Appendix B - Maximum-Likelihood and Maximum A-Posteriori Estimators

Consider the Weighted Least-Squares case, i.e., the objective function is given by:

$$J = (y - A^T \underline{x})^T R^{-1} (y - A^T \underline{x}) = \sum_{\substack{i,j \\ j \in N_i}} \frac{1}{r_{ji}} \left(\hat{O}_{ji} - \tau_i + \tau_j \right)^2$$

As we have shown previously, the optimal decentralized solution for estimating the offset at each network node according to a single set of measurements is given by:

$$\tau_i = \frac{1}{\sum_{j \in N_i} \frac{1}{r_{ji}}} \cdot \sum_{j \in N_i} \frac{1}{r_{ji}} \left(\hat{O}_{ji} + \tau_j \right) \qquad (B.1)$$

We will now compute the Maximum-Likelihood Estimator of the offsets and under some basic assumptions; we will obtain the same expression as in (B.1).

As usual, the measurements model we used is:

$$\hat{O}_{ji} = \frac{1}{2}\left(\Delta T_{ij} - \Delta T_{ji} \right) = \underbrace{\left(\tau_i - \tau_j \right)}_{O_{ij}} + \varepsilon_{ij}$$

Here, ε_{ij} is the estimation error of the relative offset between node Λ_i and node Λ_j (by definition $\varepsilon_{ij} = \varepsilon_{ji}$) .We will further assume that these random variables are independent for any $i, j = 1, ...N \ (i \neq j)$.

Let us write the expression for the probability of the estimated offset $\hat{O}_{ij}$ given the values of the vector $\underline{\tau}$ under the assumption that the errors ε_{ij} are Gaussian random variables with mean zero and variance r_{ij}:

$$P\left(\hat{O}_{ij} \mid \underline{\tau} \right) = \frac{1}{\sqrt{2\pi r_{ij}}} e^{\frac{-\left(\hat{O}_{ij} - \tau_i + \tau_j \right)^2}{2 r_{ij}}} \qquad \text{(B.2)}$$

The joint probability of the estimated offsets $\hat{O}_{ij}$ given the values of the vector $\underline{\tau}$ ($\forall i, j$ such that $j \in N_i$) is given by (assuming that ε_{ij} are independent):

$$P = \prod_{i, j \in N_i} \frac{1}{\sqrt{2\pi r_{ij}}} e^{\frac{-\left(\hat{O}_{ij} - \tau_i + \tau_j \right)^2}{2 r_{ij}}}$$

Taking the logarithm function on P, differentiating it with respect to τ_i and setting it to zero will lead to the same solution as in the Weighted-Least-Squares case. Under the assumptions that the estimation errors are both Gaussian (with zero mean) and independent random variables, our previous optimal estimated offsets (for the WLS case) also correspond to the maximally likely set of offset assignments. Moreover, it was proved in [21] that the Maximum-Likelihood estimated offsets are similar to the minimum-variance unbiased estimated offsets.

The next step is to compute the Maximum A-Posteriori estimator for the general problem where the objective function is composed of two distinct terms:

$$J = (\underline{x} - \overline{x}_0)^T P_0^{-1} (\underline{x} - \overline{x}_0) + (y - A^T \underline{x})^T R^{-1} (y - A^T \underline{x})$$

We notice that this case is a Bayesian case and thus the Maximum-Likelihood estimator is irrelevant, but we may compute the MAP estimator instead.

We consider the general case, where the matrix P_0 is not assumed to be diagonal. The joint probability of the estimated vector $\underline{\tau}$ given the values of the estimated offsets $\hat{O}_{ij}$ ($\forall i, j$ such that $j \in N_i$) is given by (assuming that ε_{ij} are independent):

$$P = P\left(\underline{\tau} \mid \hat{O}_{ij}\right) \underset{\substack{\uparrow \\ Bayes}}{=} \frac{P\left(\hat{O}_{ij} \mid \underline{\tau}\right) \cdot P\left(\underline{\tau}\right)}{P\left(\hat{O}_{ij}\right)}$$

$$P\left(\hat{O}_{ij} \mid \underline{\tau}\right) = \prod_{i, j \in N_i} \frac{1}{\sqrt{2\pi r_{ji}}} e^{\frac{-\left(\hat{O}_{ij} - \tau_i + \tau_j\right)^2}{2 r_{ji}}}$$

$$\underline{\tau} \sim N\left(\overline{x}_0, P_0\right)$$

$$P\left(\underline{\tau}\right) \propto e^{-\frac{1}{2}\left\{(\underline{\tau} - \overline{x}_0)^T P_0^{-1} (\underline{\tau} - \overline{x}_0)\right\}}$$

This implies:

$$\left(\tau_i\right)_{MAP} = \frac{1}{\left(\sum_{j \in N_i} \frac{1}{r_{ji}} + \left(P_0^{-1}\right)_{ii}\right)} \left[\sum_{j \in N_i} \frac{1}{r_{ji}} \left(\hat{O}_{ji} + \tau_j\right) + \left(P_0^{-1}\right)_{ii} \tau_i(0) - \sum_{\substack{k=1 \\ k \neq i}}^{N} \left(P_0^{-1}\right)_{ik} \left(\tau_k - \tau_k(0)\right) \right] \quad (B.3)$$

In this case, we assumed that each offset is a Gaussian random variable that is independent of the other offsets. Moreover, we used the fact that $P\left(\hat{O}_{ij}\right)$ is independent of $\overline{\tau}$ (it comes directly from the measurements: $\hat{O}_{ij} = \frac{1}{2}\left(\Delta T_{ij} - \Delta T_{ji}\right)$) and then, does not affect the differentiation. Under these assumptions, we obtain that the Maximum A-Posteriori estimator in (B.3) is the same as the optimal estimator computed previously.

Appendix C – CTP Numerical Results

We compare the CTP algorithm to three different hierarchical versions of NTP and then analyze the convergence rate of the decentralized CTP algorithm.

First, we consider Network 1 (400 nodes with relatively high connectivity of 1798 edges). Figure C.1 shows the fraction of nodes with clock offset with respect to the reference time node that is not grater than *t* for the 4 different algorithms. In other words, the y-axis represents the fraction of nodes with clock offset, with respect to the UTC, not greater than the value described by the x-axis. Figure C.1 clearly demonstrates the significant improvement of CTP (Least-Squares approach) over all the hierarchical NTP schemes considered here, in terms of clock accuracy.

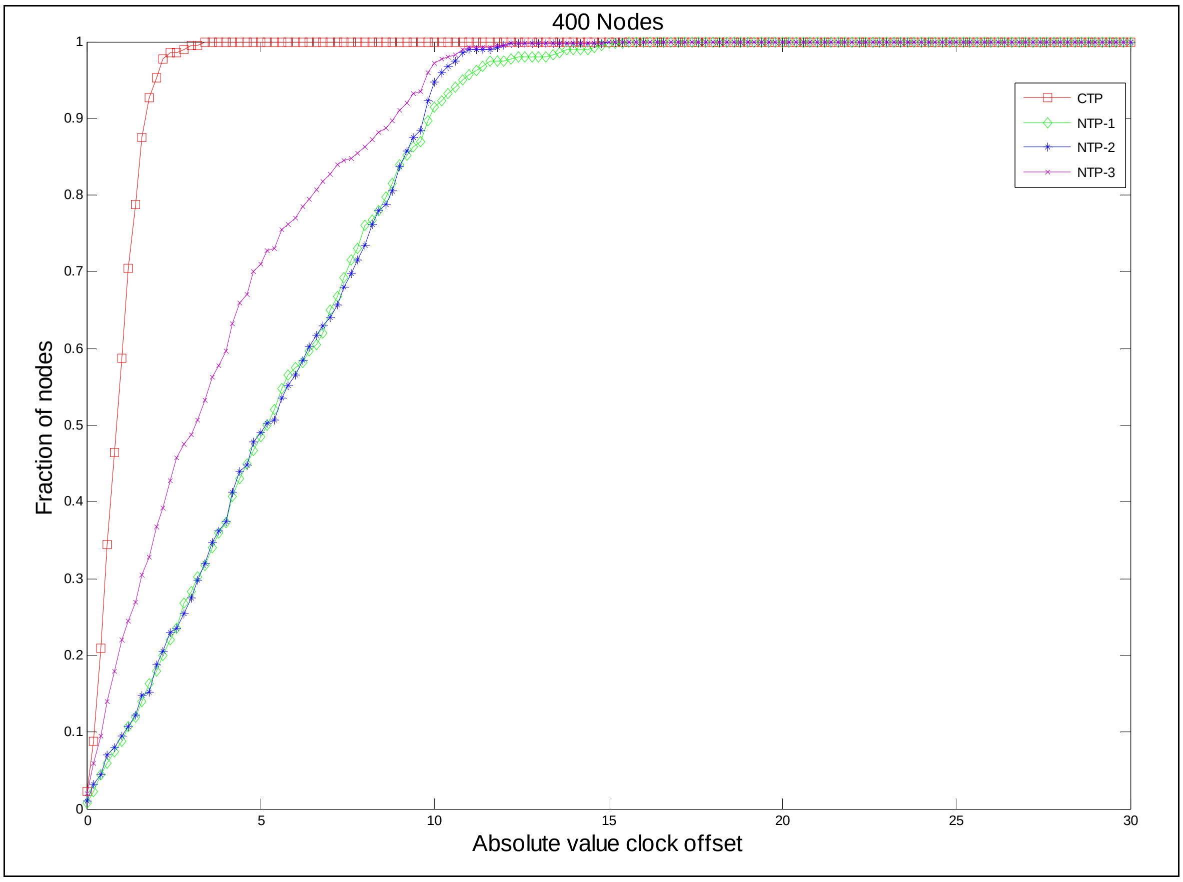


Figure C.1. Distribution of the clock offsets for each algorithm in Network 1.

Figure C.2 presents the results for the same network topology, but in a different way. The y axis shows the Probability Density Function (PDF) with clock offsets described by the x-axis.

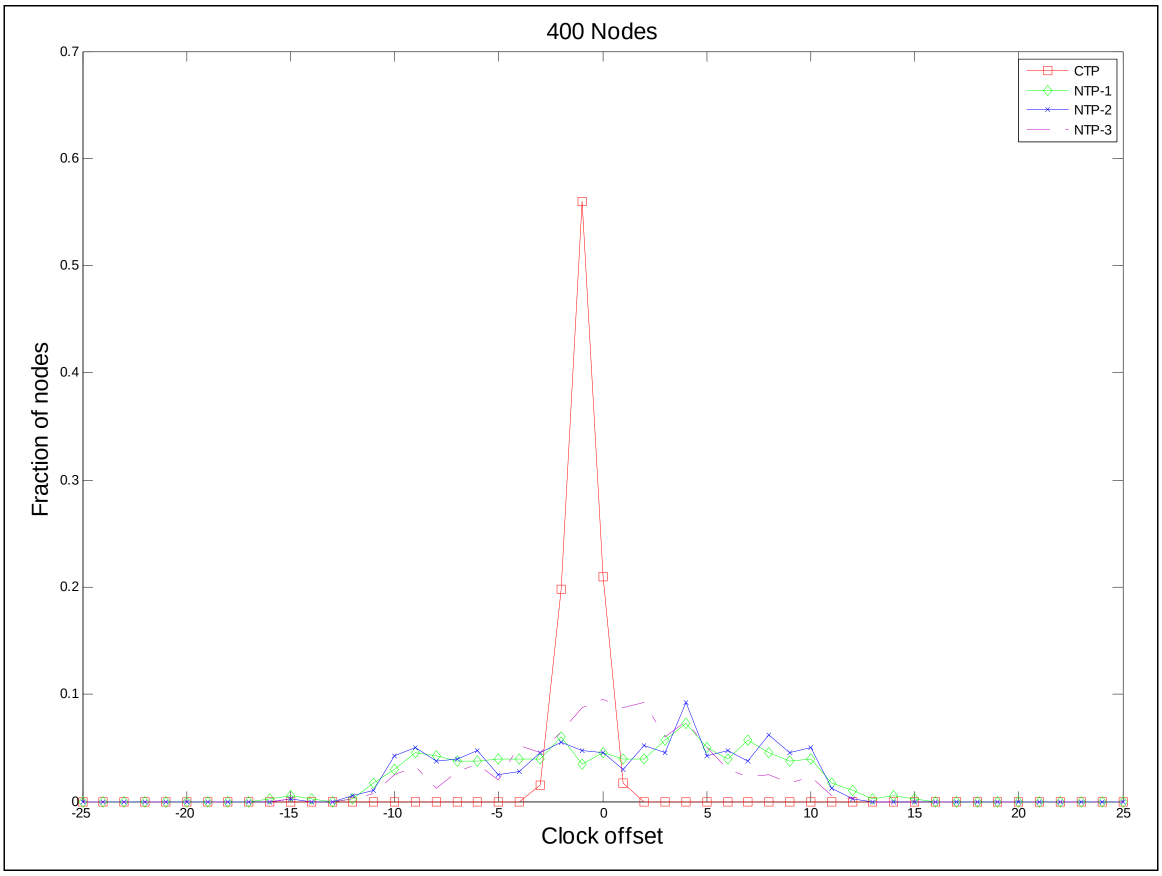


Figure C.2. Probability Density Function of the clock offsets for each algorithm in Network 1.

The next figure depicts the clock offset dispersion around the UTC clock in the same 400 node network. The x-axis corresponds to the node ID, whereas the y-axis corresponds to the clock offset with respect to the UTC clock after performing each one of the different algorithms. It can be seen that as expected, the CTP algorithm keeps all the estimated offsets in a very narrow region which means small errors and small dispersions in the clocks. The other schemes are characterized by a much wider domain. Furthermore, the thickness of the region in CTP is the same for all the nodes over the network, whereas in the NTP schemes it slightly depends on the ID. This can be explained by the fact that the NTP schemes are hierarchical and, as a consequence, the closest the nodes are from the reference (small ID), the most accurate are the offsets.

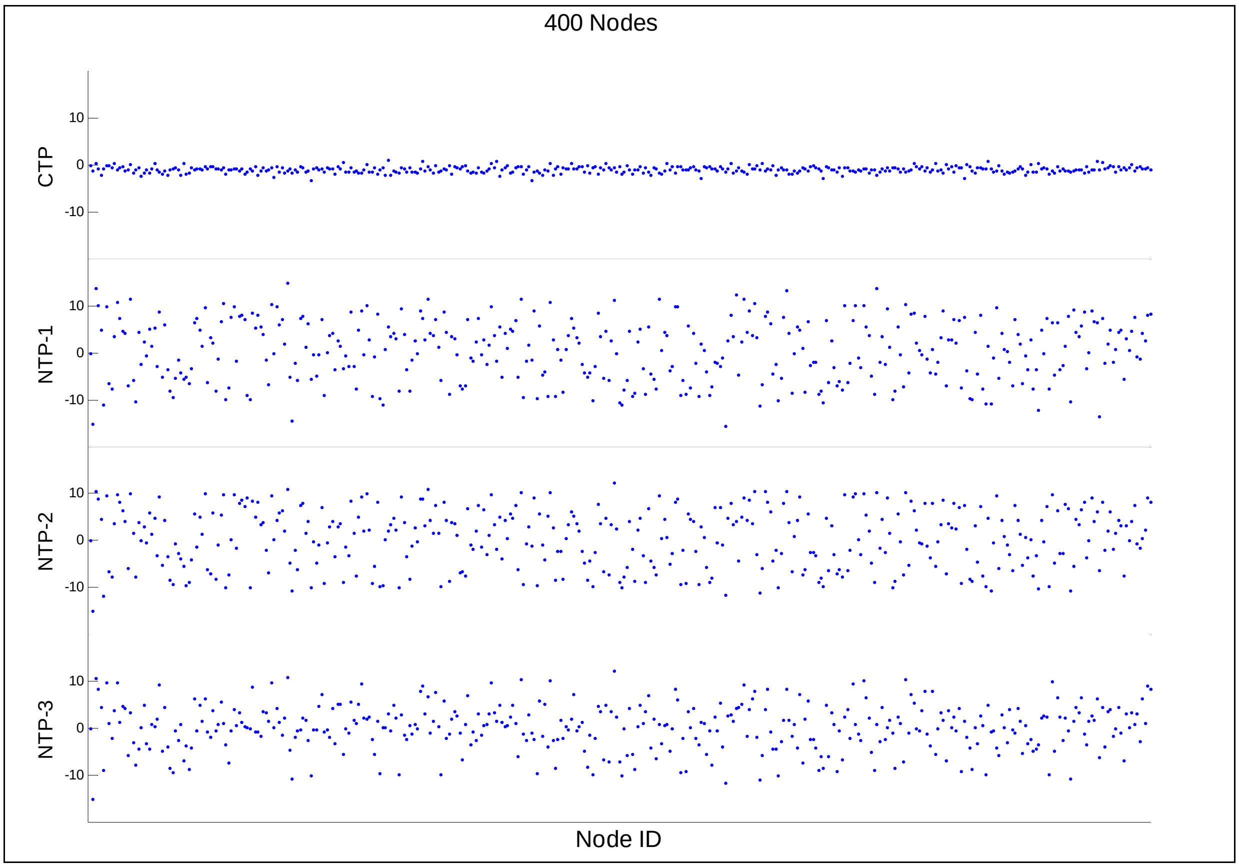


Figure C.3. The clock offsets dispersion in Network 1.

The next part of this section is dedicated to the convergence analysis of the distributed CTP algorithm. From now, we consider Network 2 (400 nodes with 997 edges).

We examined the clock offsets after 0, 1, 3, 5 and 10 iterations with respect to the optimal centralized solution. Figure C.4 describes the fraction of nodes with clock offset relating to the set of optimal values not greater than t in Network 2. We start with clock offsets that are uniformly distributed (0 iteration). It can be seen in the graph that before we start, we have very few nodes (less than 10%) that are synchronized, but 32%, 55%, 77% and 84% of the nodes are within one time unit of the optimal solution after the first, third, fifth and tenth iteration respectively.

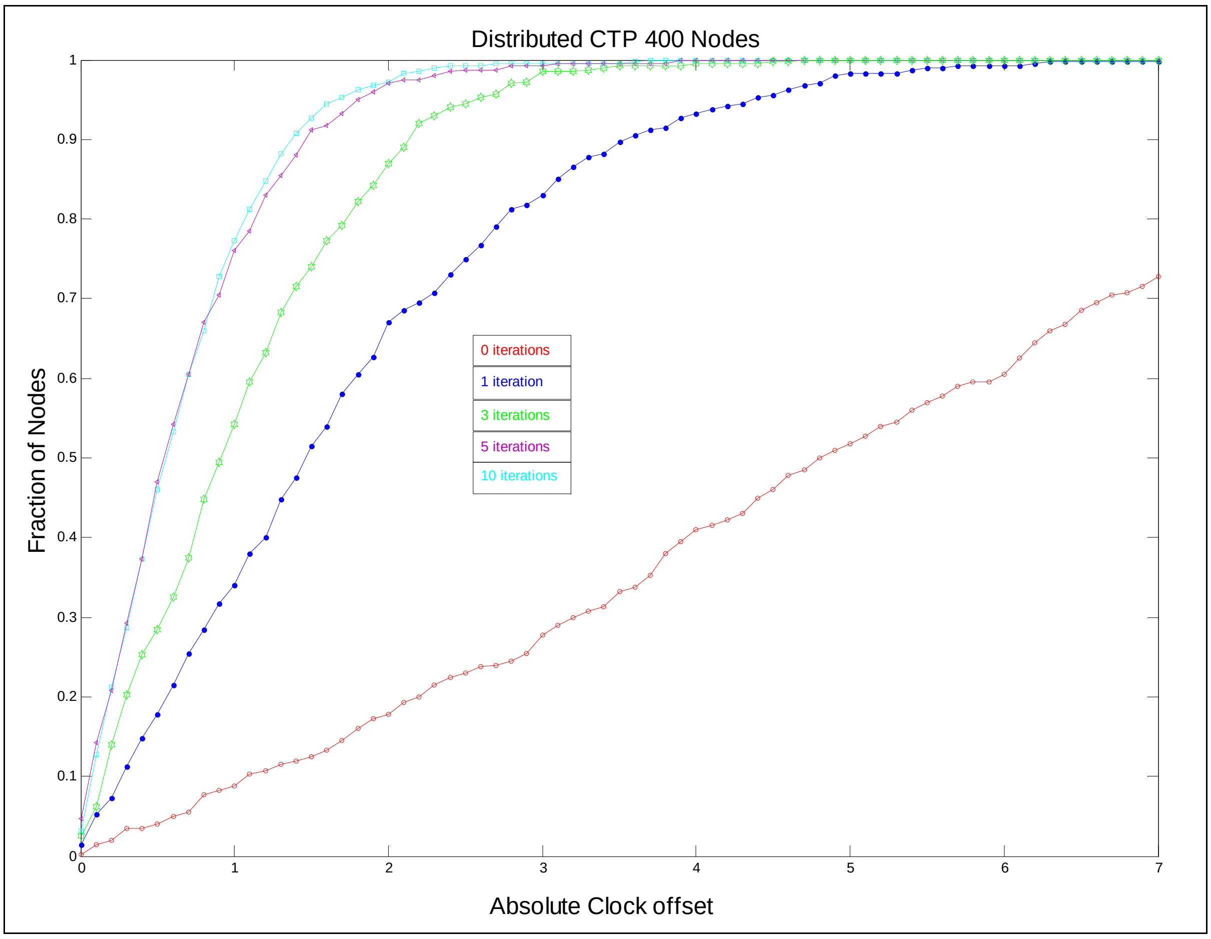


Figure C.4. Rate convergence analysis of the decentralized CTP algorithm in Network 2.

תקציר

עבודה זו דנה באלגוריתמי שערוך על מנת להציע פתרונות יעילים לבעיית סנכרון שעונים ברשת מחשבים. סנכרון שעונים מדויק מהווה דרישה בסיסית עבור רוב מערכות המחשבים המבוזרות. רמת הביצועים של יישומים רבים המבוססים על תאום בין ישויות שונות תלויה במידה רבה בדיוק הסנכרון בין השעונים השונים הנמצאים במערכת. דוגמא לכך היא בעיית עקיבה בעזרת רשת חיישנים אלחוטיים. המטרה של פרוטוקול שמסנכרן שעונים היא להבטיח ששעונים ברשת התקשורת, המרוחקים האחד מהשני, ימדדו את השעה בצורה אחידה ככל הניתן. הדרך המקובלת לסנכרן שעונים ברשת מחשבים הינה באמצעות החלפת הודעות סטנדרטיות (Probe Packets) בין הישויות המבוזרות על מנת לתאם בין הזמנים שלהם. בעבודה זו, נניח לשם פשטות שהקשרים ברשת סימטריים, שהטופולוגית קבועה בזמן, ושכל צומת ברשת מסוגל לשלוח ולקבל הודעות מהצמתים השכנים. קיימות גישות רבות בספרות המציעות פתרונות יעילים לבעיית סנכרון שעונים ברשתות מסורתיות. הסטנדרט המקובל היום לסנכרון שעונים באינטרנט למשל, הוא ה-NTP (Network Time Protocol) [Mills, 1991, 92 and 1995].

לאחרונה, הוצעה גישה חדשה: אלגוריתם ה-CTP (Classless Time Protocol) [Gurewitz, Cidon and Sidi, 2003]. הפרוטוקול פועל בצורה לא היררכית ומפחית את שגיאות השעונים מבלי להגדיל את הסבוכיות ביחס ל-NTP. גישה זו מנצלת את תורת האופטימיזציה הקמורה על מנת להעריך את ההשפעה של כל התזוזות בזמן (Clock Offsets) על פונקצית המטרה הכללית. גישה נוספת מבוססת על משערך הריבועים הפחותים (Least-Squares Estimator) [Solis, Borkar and Kumar, 2006]. דיוק משופר של סנכרון השעונים מושג ע"י התחשבות אילוצים הקשורים לטופולוגית הרשת (דהיינו העובדה שסכום תזוזות השעונים לאורך מעגלים סגורים מתאפס) ושימוש באלגוריתם מבוזר א-סינכרוני הדורש תקשורת מקומית בלבד. המאפיין המרכזי של השיטות הנ"ל הינו המבנה המבוזר הדורש תקשורת מקומית עם השכנים בלבד. ניתן להראות שאלגוריתם ה-CTP והאלגוריתם שמבוסס על שיטת הריבועים הפחותים שקולים לחלוטין.

בתורת השערוך, עבור מערכת דינמית ליניארית תחת ההנחה הגאוסית, מסנן קלמן הינו משערך המצב האופטימאלי במובן שגיאה ריבועית ממוצעת מינימאלית (MMSE). ללא ההנחה הגאוסית, הפתרון יוביל למשערך מצב הליניארי האופטימאלי במובן MMSE. היישום של מסנן קלמן בצורה מבוזרת נחקר באופן נרחב בספרות כפי שנראה בפרק 3. מטרתנו הינה לפתח אלגוריתמי שערוך יעילים על מנת לסנכרן את כל השעונים ברשת ביחס לשעון הייחוס תוך שיפור הביצועים בהשוואה לאלגוריתמים הקיימים. ללא הגבלת הכלליות, נוכל להניח שצומת מספר 1 מסונכרן עם השעון האוניברסאלי ולכן מספיק לסנכרן את כל יתר הצמתים ברשת ביחס אליו. בשלב ראשון, אנו נניח שכל השעונים רצים בקצב זהה, כלומר אין הבדלי קצב (Clock Skew) ואז המטרה שלנו היא לשערך את תזוזות הזמן של כל השעונים ברשת ביחס לשעון הייחוס.

במחקר זה, אנו נרחיב את גישת הריבועים הפחותים בעזרת פיתוח אלגוריתמי שערוך של תזוזת השעון של כל צומת ברשת המבוססים על סינון קלמן. השלב הראשון יהיה לייצג את המערכת במודל מישור המצב כאשר וקטור המצב נתון ע"י תזוזות שעון של כל צומת ברשת. בנוסף, אנו נראה שניתן לבצע עדכון מדידה בודדת בעזרת סכימה מבוזרת איטרטיבית, המתכנסת למשערך האופטימאלי המרוכז. המסגרת של מסנן קלמן מאפשרת לנצל את הידע האפריורי על המערכת, ולהעניק משקלים שונים למדידות ביחס לאיכותן ולדיוקן. נניח באופן טבעי שמטריצת הקווריאנס ההתחלתית הינה אלכסונית. אולם, נשים לב כי אחרי השלב של עדכון המדידה הראשונה של המסנן קלמן, מטריצת הקווריאנס מאבדת את המבנה האלכסוני שלה. במילים אחרות, החל משלב זה המשוואות של המסנן קלמן לא יהיו מבוזרות ואז כל צומת ברשת מוכרח לתקשר עם כל צומת אחר. זהו כמובן מצב לא רצוי, כי עבור רשתות גדולות זמן התקשורת יהיה בלתי נסבל. אנו נפתור בעיה זו בעזרת אלגוריתם רקורסיבי מבוזר המבוסס על מניפולציה של המשוואות הסטנדרטיות. נסתמך על משפט שטוען שהפתרון של המסנן קלמן שקול לפתרון של מינימיזציה מאולצת על פונקצית מטרה דטרמיניסטית ואז נוכל לקבל את המקרה הקיים בספרות (LS) כמקרה פרטי. על מנת למצוא את הפתרון האופטימאלי, יש לגזור את פונקצית המטרה לפי כל קוארדינאטה במקרה המבוזר ולהשוות לאפס (במקרה המרכזי, נשווה את הגרדיאנט לאפס). כדי לממש

את המשוואה האופטימאלית המתקבלת, אנו נבחר באלגוריתם איטרטיבי סינכרוני ונראה התכנסותו לפתרון האופטימאלי המרוכז בעזרת כלים מתורת המטריצות האי שליליות. המשמעות של אלגוריתם מבוזר הינה שכל צומת ברשת מחשב את תזוזת השעון שלו באמצעות תקשורת עם השכנים בלבד. עד כה, התייחסנו למקרה שרק מדידה אחת זמינה. בשלב הבא נרחיב את הטיפול למקרה שבו קיימות מספר קבוצות של מדידות בזמנים שונים, ונציג גרסה רקורסיבית של האלגוריתם השומר כמובן על המבנה המבוזר שלו. אלגוריתם זה מהווה את הסכימה המרכזית של העבודה מכיוון שהוא מבוזר ומתכנס לפתרון האופטימאלי המרוכז במובן MMSE. חשוב לציין שהפתרון זהה לפתרון של המסנן קלמן בצורת אינפורמציה, אך המבנים של שתי השיטות שונים בצורה מהותית. בנוסף לחישוב של תזוזות הזמן המשוערכות לכל צומת ברשת, האלגוריתם מניב את השונויות של שגיאות השערוך, דבר הנותן לנו אינדיקציה על טיב השערוך. בנוסף, אנו נתייחס לאלגוריתם התת-אופטימאלי הפשוט שמזניח את האיברים מחוץ לאלכסון של המטריצה ההפוכה למטריצת הקווריאנס. שיטה זו מורידה בצורה ניכרת את הסיבוכיות אך מאבדת את תכונת האופטימאליות. אנו נראה בפרק שדן בתוצאות הסימולציה שהאלגוריתם האחרון משיג ביצועים חלשים ולא מהווה פתרון יעיל לבעיה.

נדון במספר הרחבות של האלגוריתם הבסיסי, הכוללות הוספה של מקדם היוון לפונקציית המחיר הריבועית והוספה של רעש מערכת במשוואת הדינאמיקה. כמו כן, נתאים את האלגוריתם כך שיהיה עמיד בפני שגיאות זמניות בתקשורת. בנוסף, נתייחס בקצרה לבעיית השערוך של התזוזה בזמן ביחד עם התזוזה בקצב (Clock Skew). אנו נראה ששתי הבעיות מצטמצמות לאותה המסגרת המתמטית תחת ההצבות המתאימות. עבור בעיית שערוך תזוזת הקצב, נציע גישות שונות. בגישה אחת, שערוך הקצב מתבצע בנפרד לשערוך התזוזה. בגישה השנייה, נציע שערוך אופטימלי משולב של תזוזות הקצב והזמן בעזרת מסנן קלמן.

לאחר מכן, נציג מספר תוצאות סימולציה עבור טופולוגיות רשת שונות על מנת להעריך ולהשוות את הדיוק של הסכימות השונות. כצפוי, נקבל כי הגישה של מסנן קלמן משפרת את הביצועים מבחינת דיוק סנכרון השעונים. במילים אחרות, האלגוריתם שמבוסס על מסנן קלמן מניב תוצאות משופרות בהשוואה לשיטות האחרות הקיימות בספרות, כגון: גרסאות שונות של NTP ואלגוריתם ה-CTP. נבצע השוואות שונות הכוללות: הוספת מטריצת משקול למדידות, הכללה של ידע התחלתי ומימוש של האלגוריתם הרקורסיבי עבור מספר קבוצות של מדידות. אנו נסיק שהשיטה המוצעת הינה הרחבה של האלגוריתם הקיים, ותחת התנאים המתאימים נותנת את הביצועים הטובים ביותר. נציין כי תוצאות הסימולציה מתבצעות תחת ההנחה שכל השעונים ברשת רצים במהירות זהה.

נציין כי בעיית סנכרון שעונים ברשת הינה שקולה מבחינה מתמטית לבעיה כללית של שערוך מבוזר בעזרת מדידות יחסיות אדיטיביות ברשתות סנסורים. לדוגמה, ניתן ליישם את אותם האלגוריתמים לבעיית איכון סנסורים (Sensor Localization Problem) במישור או במרחב, בעזרת הרחבה וקטורית פשוטה. אנו נפרט על כך בקצרה בפרק 7.

מבנה העבודה הזו הוא כדלקמן. בפרקים 2 ו-3, נבחן את הרקע המדעי הנחוץ ונביא סקירה מקיפה של הספרות בהתאמה. בפרק 4, נתאר את המודל וננסח את הבעיה. בפרק 5, נציג את האלגוריתמים השונים עבור המקרה של מדידה בודדת (הן בגרסה המרוכזת והן בגרסה המבוזרת). פרק 6 מוקדש להוכחת ההתכנסות של האלגוריתם המבוזר לפתרון האופטימאלי המרוכז. בפרק 7, אנו נספק את הגרסה הרקורסיבית של האלגוריתם עבור מדידות מרובות ואלגוריתם נוסף לא רקורסיבי. פרקים 8 ו-9 דנים במספר הרחבות של המקרה הבסיסי, כגון הוספת מקדם היוון והשערוך של תזוזת הקצב. תוצאות סימולציה מוצגות בפרק 10. לבסוף, מסקנות ומספר הערות אודות מחקר עתידי מצוינות בפרק 11.

תוכן עניינים

המחקר נעשה בהנחיית פרופ' נחום שימקין
בפקולטה להנדסת חשמל

הכרת תודה

ברצוני להודות לפרופ' נחום שימקין על הנחייתו, על התמיכה ועל האווירה המצויינת בכל שלבי המחקר. בנוסף, ברצוני להודות לבוחנים עבור התובנות רבות הערך שלהם על התיזה הזאת. תודה מיוחדת להוריי על האהבה שלהם ועל החינוך שהעניקו לי, בפרט על הדגש העצום על מצוינות אקדמית.

אני מודה לטכניון על התמיכה הכספית הנדיבה בהשתלמותי

סנכרון שעונים ברשת בעזרת מסנן קלמן מבוזר

חיבור על מחקר

לשם מילוי חלקי של הדרישות לקבלת התואר מגיסטר
למדעים

מקסים כהן

הוגש לסנט של הטכניון – מכון טכנולוגי לישראל.

חשוון תשס"ט חיפה אוקטובר 2009

סנכרון שעונים ברשת בעזרת מסנן קלמן מבוזר

מקסים כהן