Patterns of Mobility and Clock Synchronization in Wireless Networks

Sirajo Abdullahi Bakura

To cite this version:

Sirajo Abdullahi Bakura. Patterns of Mobility and Clock Synchronization in Wireless Networks. Networking and Internet Architecture [cs.NI]. Université Paris-Saclay, 2021. English. NNT : 2021UPASG079 . tel-03712019

HAL Id: tel-03712019
https://theses.hal.science/tel-03712019v1

Submitted on 2 Jul 2022

Patterns of Mobility and Clock Synchronization in Wireless Networks

Modèles de mobilité et synchronisation d'horloge dans les réseaux sans fil

Thèse de doctorat de l'université Paris-Saclay

École doctorale n° 580, Sciences et technologies de l'information et de la communication (STIC)
Spécialité de doctorat: Informatique
Unité de recherche: Université Paris-Saclay, CNRS, Laboratoire interdisciplinaire des sciences du numérique, 91405, Orsay, France
Graduate School: Informatique et science du numérique
Référent: : Faculté des sciences d'Orsay

Thèse présentée et soutenue à Paris-Saclay, le 8 novembre 2021, par

Sirajo ABDULLAHI BAKURA

Composition du jury:

Marc SHAWKY Professeur, Université de Technologie de Compiègne, Heudiasyc	Président & Rapporteur
Anthony BUSSON Professeur, IUT La Doua Université Lyon 1, LIP	Rapporteur & Examinateur
Janna BURMAN Maître de Conférences, Université Paris-Saclay, LISN	Examinatrice
Aminu MOHAMMED Professeur,Usmanu Danfodiyo University Sokoto Nigeria, Computer Science Laboratory	Examinateur

Direction de la thèse:

Alain LAMBERT Maître de Conférences, Université Paris-Saclay, LISN	Directeur de thèse
Thomas NOWAK Maître de Conférences, Université Paris-Saclay, LISN	Co-encadrant

Thèse de doctorat

NNT: 2021UPASG079

Abstract

Mobility in conventional ad-hoc networks is a challenge due to the constant invalidation of end-to-end paths. We deal with mobile ad-hoc networks where humans are the main carriers of mobile devices. A good understanding of human mobility patterns aids the design of a realistic mobility model as a tool for evaluating network protocols.

Conventional models for evaluating network protocols in early ad-hoc networks (e.g., random walks, random waypoints, random directions) fail to properly capture human mobility. In fact, recent studies have shown that human mobility is influenced by personal habits, social relationships, environmental features, and locations preferences. Therefore, a realistic model should be able to include these features.

In this regard, we develop a heuristic to characterize human mobility based on the spatial, temporal, and connectivity features using real traces. Consequently, we uncover temporal dynamic movement clusters associated with individual users. We also study the distribution of the travel distance, pause time, angle of movement, contact duration, and inter-contact duration. Motivated by our findings, we proposed a new synthetic mobility model that mimics realistic features of human mobility. We validate the model by comparing its synthetic traces against real mobility measurements.

Moreover, in a smart-campus environment, networks support applications for environmental monitoring and indoor/outdoor positioning, sometimes with a large deployment of sensors. Considering the limitation of sensors such as battery limitation, dynamicity, and low computing clock rate, sensor clocks need to have a common time to perform information fusion algorithms, implement energy management protocols, or real-time processing for safety applications.

Given this, we proposed a pulse-coupled distributed clock synchronization algorithm for wireless sensor networks to reduce the clock skew due to the ambient conditions, mobility, or manufacturing defects. In our algorithm, sensors measure time differences by only exchanging zero-bit pulse instead of packets. Therefore our algorithm is lightweight and robust to the failure of the sensors in the network. The proposed algorithm is compared to previous work under static and mobile settings, and the results show that it can reduce the clock skew, especially in a dynamic environment with high uncertainty in clock drift and unexpected topological changes like vehicular networks.

Acknowledgements

I would like to extend my sincere gratitude to my thesis director Professor Alain Lambert and my co-supervisor Professor Thomas Nowak for their immense support and counseling throughout this study. I wholeheartedly appreciate their courage, patience, and dedication toward the success of this thesis.

I would also like to appreciate the tremendous efforts of jury members for their great patience and insightful suggestions. I believe that their ideas are inspiring and meaningful in improving the readability and highlighting the results.

Furthermore, I would be glad to appreciate the effort of the Nigerian government in giving me the opportunity to benefit from the scholarship of the Petroleum Technology Development Fund (PTDF). Similarly, I would like to thank the staff and caring friends at the LRI laboratory for their supports and encouragement throughout this program.

Finally, it is with great honor to appreciate the support of my family members without which this academic journey would not have been possible.

Synthèses en Français

Cette thèse traite des réseaux dynamiques dans lesquels les utilisateurs sont des entités mobiles. Dans ce type de réseaux, la mobilité provoque des ruptures de chemin et des changements dans la topologie du réseau. De même, il est nécessaire d'avoir une notion commune du temps parmi les terminaux du réseau pour l'implémentation d'algorithmes de fusion de données, de protocoles de gestion de l'énergie, et la fourniture de services en temps réel.

Pour ces raisons, la thèse caractérise les caractéristiques spatio-temporelles et de connectivité de la mobilité humaine dans les réseaux sans fil en utilisant des traces réelles. En conséquence de la caractérisation, nous découvrons des modèles cachés appelés clusters de mouvement dynamique temporel. Nous avons donc proposé un nouveau modèle de mobilité qui génère des modèles de mobilité humaine similaires aux caractéristiques observées dans les traces de mobilité réelles. Ce modèle sert d'outil d'évaluation pour l'analyse et l'évaluation des protocoles de réseau. Nous proposons également un algorithme de synchronisation d'horloge à couplage d'impulsions qui tolère les impacts de la mobilité des utilisateurs et utilise un minimum de ressources informatiques sans sacrifier la précision de la synchronisation. L'algorithme a considérablement réduit le skew et le décalage de l'horloge par rapport à l'algorithme précédent et peut donc être utilisé pour mettre en œuvre des algorithmes de fusion de données pour la surveillance environnementale, les protocoles de gestion de l'énergie ainsi que les services en temps réel. La thèse est structurée en deux parties : La partie I traite de la caractérisation de la mobilité humaine et de la conception du modèle de mobilité proposé, tandis que la partie II étudie la synchronisation d'horloge dans les réseaux dynamiques et propose un algorithme de synchronisation d'horloge couplé à des impulsions.

Dans la partie I de la thèse, nous menons une étude approfondie sur les caractéristiques spatio-temporelles et de connectivité de la mobilité humaine, qui comprennent : La distance de déplacement, l'angle de déplacement, le temps de pause, la durée d'inter-contact et la durée de contact. Nous analysons deux ensembles de données de localisation et un ensemble de données de contact provenant de trois campus universitaires (KAIST, NCSU et Cambridge) au niveau du grain fin afin de mieux comprendre les propriétés de la mobilité humaine et de découvrir des modèles cachés.

Pour ce faire, nous avons développé une heuristique pour extraire les emplacements de mouvements stationnaires et groupés à partir des ensembles de données étudiés. Nous extrayons les distributions de différentes caractéristiques de mobilité observées à partir des traces de mobilité réelles. Nous adaptons la meilleure distribution à chacune des données empiriques. Ensuite, nous évaluons les paramètres de chaque distribution.

Motivés par les résultats de notre analyse, nous avons proposé un nouveau modèle synthétique de mobilité appelé Escape Path Mobility Model (EPOM). L'EPOM est composé de cinq sous-modèles (à savoir, domicile, études, cafétéria, sport et hors campus) en plus du sous-modèle de transport et d'obstacles. Un nœud se déplace entre les sous-modèles en utilisant le sous-modèle de transport et utilise le sous-modèle d'obstacle pour générer des chemins de fuite lorsqu'il rencontre un obstacle sur sa trajectoire de déplacement. Nous avons démontré que le modèle EPOM pouvait recréer les modèles statistiques observés dans les traces réelles étudiées à travers une série de simulations.

Dans la partie I de la thèse, nous traitons de la synchronisation d'horloge, qui est une pierre angulaire pour l'implémentation d'algorithmes de fusion de données, de protocoles de gestion de l'énergie, et la fourniture de services en temps réel. Nous avons proposé un algorithme de synchronisation d'horloge distribuée par couplage d'impulsions pour les réseaux de capteurs sans fil afin de réduire le décalage d'horloge dû aux conditions ambiantes, à la mobilité ou aux défauts de fabrication.

L'algorithme étend le travail proposé dans Han [2018] en développant une technique d'apprentissage adaptative qui compense le décalage et la dérive de l'horloge. L'algorithme est composé de trois fonctions principales : fonctions d'initialisation, de réception et de transition d'état. L'algorithme fonctionne par cycles, donc à chaque cycle, il estime les compensations de décalage et de dérive. Il accorde plus de poids aux estimations récentes en exploitant l'heuristique de la nouvelle méthode proposée pour effectuer la compensation du décalage et de la dérive. La méthode d'apprentissage adopte une approche adaptative dynamique pour choisir une valeur optimale du facteur de pondération λ. Pour évaluer l'algorithme, nous avons dérivé une limite analytique sur le décalage d'horloge dans deux cas : Premièrement, lorsque seule la correction du décalage est appliquée dans le théorème 1 et deuxièmement lorsque les corrections du décalage et de la dérive sont appliquées dans le théorème 2. Nous évaluons également l'algorithme par le biais d'une série de simulations et le comparons avec les travaux antérieurs de Han [2018], et de Fugger et al. [2015] dans des paramètres statiques et mobiles, et les résultats ont montré qu'il a réduit le skew d'horloge, en particulier dans un environnement dynamique avec une grande incertitude dans la dérive de l'horloge et des changements topologiques inattendus comme le réseau de véhicules. La robustesse de l'algorithme proposé dans des contextes réalistes comme le réseau de véhicules et le scénario de marche aléatoire en intérieur a montré que les techniques de synchronisation couplées par impulsion s'adaptent aux environnements difficiles.

Contents

List of Figures

List of Tables

Chapter 1

General Background

1.1 Introduction

In wireless networks, humans are the main carriers of wireless devices. A better understanding of human mobility patterns in wireless networks aids the design of a realistic mobility model as a tool for evaluating network protocols. Several factors are attributed to the dynamics of human mobility in wireless networks, for instance in Wireless Local Area Networks (WLANs) people make a decision to move between wireless access points to search for better connection, social relationships influence people ability to meet which in turn creates contact opportunities in Delay-Tolerant Networks (DTNs). Similarly, human social attraction and job conditions influence vehicles' movement toward points of interest, impacting traffic flow in Vehicular Ad-Hoc Networks (VANETs).

Most of the existing models for evaluating networking protocols in the early ad-hoc networks were based on the synthetic, conventional mobility models like a random walk Lawler and Limic [2009], random waypoint Johnson and Maltz [1996], and the random direction model Bettstetter [2001], just a few to mention. Several studies have been conducted on their adaptability in the next generation mobile networks like DTNs, VANETs, and Wireless Sensor Networks (WSNs), which show human mobility characterized by intentional mobility as opposed to the random assumptions in the conventional models.

The most widely studied of all conventional models is the random waypoint Broch et al. [1998]. Nevertheless, the movement patterns generated by the model consist of random locations separated by random distances. However, the movement pattern of humans is not completely random; instead, it shows dependencies Munjal et al. [2012]. Similarly, Random walk models generate mobility patterns in which mobile nodes display a completely random behavior that can only be seen in a few wireless networks like sensor network for animal tracking Dominguez-Morales et al. [2016]; Sanchez [1998]).

However, several studies have shown that human mobility is rarely random González et al. [2008]. Therefore, random-based models fail to analyze the protocols in the infrastructure-less-based networks like wireless mobile ad-hoc networks (MANETs), DTNs, and WSNs accurately. Though we cannot completely rule out the statistical similarities of random walks and human movement Rhee et al. [2008]; Lee et al. [2009]. To develop more realistic models that capture human mobility, which is influenced by a person's personal habits, social relationship, environmental features, and locations preferences Aschenbruck et al. [2011], the research communities have made efforts to analyze real mobility traces, and this leads to the design of many synthetic models Rhee et al. [2009]; Munjal et al. [2011].

Recent studies on mobility models have focused on using real mobility traces to develop more realistic synthetic models. These studies became successful with the help of open-source projects like CRAWDAD Henderson [2020] and Microsoft Research Zheng et al. [2008] that provide free access to the archived realistic mobility traces recorded from different experiments at different locations: campuses, conferences, shopping malls, and tourist parks, to mention a few. CRAWDAD alone provides free access to over 125 datasets.

In the previous studies by Kim et al. [2006]; Hui et al. [2005]; Kotz and Essien [2002], synthetic mobility models based on user mobility characteristics extracted from wireless network traces *syslog* was presented. The models use Wi-Fi access points, which have a higher granularity of mobility trajectories. Considering the nature of DTNs, mobility models developed from empirical features with higher granularity (order of 100m) cannot be used to account for small *travel distances* and *angle of travel* in a geo-location at which user spent most of its time. GPS-based data related to human mobility are made available to ease granularity constraints, and their use is growing exponentially Zignani and Gaito [2010]. With this development, analysis, design, testing, and evaluation of wireless networks and their protocols at fine-grain level became possible when an appropriate synthetic model is used.

This thesis carried out an in-depth analysis of the spatial, temporal, and connectivity features of human mobility in infrastructure-less wireless networks to develop a synthetic mobility model for smart campus wireless network protocol design and evaluation. We conducted a comprehensive analysis of realistic datasets for Bluetooth encounters, user associations to the mobile wireless networks, and Global Positioning System (GPS) tracklogs traces at the fine-grain level to understand the properties of human mobility better and to uncover hidden patterns. Interestingly, we have uncovered time-varying human mobility patterns associated with a dynamic evolution of movement clusters in which a user undergoes many short walks within his community domain. The empirical statistical features of such dynamic evolving clusters shed light on the realistic feature of human mobility, which can be exploited to predict user location and optimization of localization systems. Motivated by our findings on the Spatio-temporal dynamic clusters, coupled with our in-depth analysis on the connectivity features of realistic traces, we proposed a new synthetic realistic mobility model as a tool for the performance analysis and design of protocols in wireless networks. The proposed model closely mimics the empirical features of the studied real traces Rhee et al. [2009]; Scott et al. [2009].

Moreover, in time-based environmental monitoring and positioning applications, the ability of sensors to accurately estimate a target object's physical location depends on its ability to synchronize the value of its clock to a common global value for efficient execution of fusion algorithms. Some of the existing positioning and localization systems use a reference station node to synchronize the clock of the participating nodes Sidorenko et al. [2019]; Gustafsson and Gunnarsson [2003]. However, when the location of a sensor node is uncertain, the master node could not be able to broadcast synchronization packages to all nodes, so that a universal clock consensus may be unattainable. Additionally, the failure of a reference node leads to the total failure of the network synchronization. There is a need for a better approach to handle the limitations of the centralized systems mentioned above.

In this thesis, we proposed a distributed algorithm for synchronization among sensors without any absolute reference high-accuracy clock. Our work extended the work presented in Han [2018] by proposing a learning technique, which helps to reduce a clock offset and drift. Our algorithm is based on a temporal exponential smoothing of correction terms, which is able to tolerate significant network changes during execution. The offset compensation part of the algorithm ensures a common notion of time among nodes while the drift compensation reduces clock skew. We have evaluated the algorithm under a series of simulations in dynamic and static configurations. The results show a significant improvement over previous algorithms in reducing clock offset and drift.

In a general sense, the first part of the thesis conducted an in-depth analysis of human mobility. It proposed a tool for the evaluation of wireless network protocols. Similarly, in the second part, we proposed a distributed clock synchronization algorithm for dynamic networks, which is the cornerstone for an efficient data fusion in indoor or outdoor time-based localization systems and environmental monitoring for effective learning in a smart campus environment.

1.2 Research Objectives

The general objectives of the thesis are twofold:

1. To develop a synthetic mobility model for the design and analysis of wireless network protocols
2. To develop a distributed clock synchronization algorithm for wireless networks to support efficient data fusion.

The specific objectives of the thesis are:

1. To investigate the spatial, temporal, and connectivity features of human mobility in wireless networks using realistic and synthetic traces to uncover hidden patterns that best characterize human mobility.
2. To integrate the observed mobility features in the design of a new mobility model that could be used to analyze and evaluate protocols in wireless networks.
3. To implement a distributed clock synchronization algorithm that tolerates impacts of node mobility.
4. To design an optimal algorithm that utilizes minimum computational resources without sacrificing the synchronization accuracy.
5. To develop a new technique of reducing short-term fluctuations due to the unpredictable changes in the *clock drift*

1.3 General Outline

The thesis consists of two main parts; the first part presents an analysis of spatial, temporal, and connectivity features in human mobility. It proposed a new mobility model for the design and evaluation of wireless network protocols. In the second part, we present a distributed clock synchronization algorithm, which performs offset and drift compensation to improve precision accuracy.

Part I

Characterization of Human Mobility and Design of Mobility Model.

Chapter 2

Introduction

This chapter presents a brief background on the characterization of human mobility features and the design of mobility models. It describes the objectives of the first part of the thesis, followed by the contributions of the thesis. Finally, the outline of the characterization of human mobility and the design of human mobility is presented.

2.1 Background

Communication in MANETs is performed in a peer-to-peer fashion using short-range mobile devices via technologies like Bluetooth and Wi-Fi. In a smart campus scenario, some static nodes are used as part of the network. In this type of setting, peer-to-peer contact opportunities are the driving factor for information forwarding, which can be fully understood by studying human mobility since humans are the main mobile device carrier. Some models attempt to account for a statistical feature of human mobility but are based on intuition more than reliable information about the mobility features and their respective distributions. We adopted a different approach of characterizing such features by developing a modeling framework that applies formal statistical learning techniques to extract connectivity distributions of contact duration, inter-contact duration, and contact per hour for model calibration. In this regard, we validate our model by comparing our results against real mobility measurements.

Similarly, in addition to the connectivity metrics, we define a formal framework for extracting spatial and temporal features of human mobility from the user traces. The spatial features refer to the trajectory pattern in physical space while the temporal feature is related to the time-varying feature of user mobility Pirozmand et al. [2014]. Our analysis of these features unveils a hidden pattern that evolved with a unique trajectory pattern over time in a community. We observed that users' trajectories are characterized by temporal clusters, which evolve over time. The user spent a long time within such clusters and took many short walks within the vicinity of the community. We develop a technique for the extraction of movement trajectories characterized by the observed pattern from GPS-track logs and extensively analyze the mobility features associated with the clusters and the entire domain. The studied mobility features related to clusters are *travel distance distribution*, and *angle of movement*.

Several models have been developed to characterize human mobility patterns in wireless networks Kim et al. [2006]; Zignani and Gaito [2010]; Hsu et al. [2007]; Nguyen et al. [2011]. However, due to human mobility's complexity and dynamic nature, synthetic models that closely mimic realistic mobility are in tremendous demand for a more accurate evaluation of networking protocols. Our study characterizes mobility features within the movement clusters and whole domain as opposed to the existing studies that characterize features of the whole domain only. To the best of our knowledge, this is the first approach to analyze both connectivity, spatial, and temporal features at a microscopic level with a small granularity in one study.

2.2 Objectives

The specific objectives of this part of the thesis are:

1. To investigate the spatial, temporal, and connectivity features of human mobility in wireless networks using realistic and synthetic traces to uncover hidden patterns that best characterize human mobility.

2. To integrate the observed mobility features in the design of a new mobility model that could be used to analyze and evaluate protocols in wireless networks.

2.3 Contributions

The emergent wireless networks characterized by intermittent connections like human-centric sensing networks, where humans carry nodes, have motivated research on human mobility patterns. However, the design and performance evaluation of such systems' protocols is cost-effective, requiring large-scale test-beds. Hence simulation becomes the most effective tool for the evaluation of network protocols. In this regard, a realistic synthetic mobility model is needed to evaluate the performance of different networking protocols.

In this part of the thesis, we conduct a comprehensive study on the characteristics of human movement based on the spatial, temporal, and connectivity features. Consequently, we uncovered time-varying human mobility patterns associated with dynamic movement clusters in which the user undergoes many short walks within his community domain. Inspired by the uncovered features, we proposed a realistic mobility model based on the realistic human mobility features observed from the studied real traces Rhee et al. [2009]; Scott et al. [2009]. The proposed model is generic enough to be fine-tuned with a few parameters to show matching characteristics with the realistic human traces. Moreover, in this thesis we deal with the application scenario of campus environment, in which the campus users' devices forward messages to each other using peer-to-peer opportunistic communication.

2.4 Outline

Chapter 3 presents the background knowledge on the use of real traces to develop a synthetic mobility models. It gives an overview of different mobility models in the form of related classes. In Chapter 4, we conducted a characterization of the spatial, temporal, and connectivity features of human mobility using real traces. Consequently, we uncover the value(s) of the parameter(s) for the distribution of the inter-cluster travel distance, intra-cluster travel distance, intra-cluster angle of movement, pause time, inter-contact time, and contact time as observed from the studied real traces. A detailed description of the proposed model is presented in Chapter 5. The model domain, initial location selection, trip schedule, displacements, and pause time selection are detailed in the second section of Chapter 5. The third section describes the model implementation. In Chapter 6 we validate our model and show that the model is generic enough to be fine-tuned with a small number of parameters and show matching characteristics with the realistic traces. Finally, we present the conclusion, contributions, and future perspective of this part of the thesis in Chapter 7

Chapter 3

Related Works

3.1 Introduction

In this chapter, we review existing works on human mobility characterization and mobility models. We focus on the works that characterize human mobility features using real-world mobility traces and synthetic mobility models.

3.2 Real-world Mobility Traces

Several research works have been recently committed to characterizing human mobility patterns based on real mobility traces González et al. [2008]; Zignani and Gaito [2010]; Aschenbruck et al. [2011]; Galati et al. [2015]. The availability of wireless network traces on the repositories like CRAWDAD Henderson [2020] and Microsoft Zheng et al. [2008] have ease access to the achieved traces. The two main categories of real traces are the *location-based* and *contact based.* In the *location-based* traces, a simultaneous user location is recorded from the GPS readings, user's association with Wi-Fi access point, or cellular base station. The *contact-based* traces, on the other hand, are generated from the set of pair-wise contacts of short-range wireless technology like Bluetooth and Wi-Fi without recording the location of the contacts.

To be more precise, the *location-based* traces are used in the study of human mobility patterns due to the trajectory records contained in the traces, whereas *contact-based* traces are used to characterize contact patterns between individuals. The *location-based* traces can be used to characterize both mobility and contact patterns in some circumstances. In this regard, the *location-based* traces need to be subjected to pre-processing to extract contact logs starting from the initial location. In our study we use *location-based* traces Rhee et al. [2009] to characterize user mobility patterns and *contact-based* trace Scott et al. [2009] to characterize contact patterns between two nodes to avoid possible approximation errors in pre-processing of *location-based* traces to extract contact details. The use of both the *location-based* and *contact-based* traces aids the characterization of spatio-temporal and connectivity features of human mobility.

Several works have been presented, which use real traces to extract human mobility features. In some studies, the extracted features are used in the development of a synthetic mobility model. Kim et al. [2006] studied a large trace of mobile devices associating with Access Point (AP) in a WLAN. The study extract tracks of users associating with AP. In this type of trace, the mobility feature of WLAN users is difficult to predicts because user's trajectories are not part of the logged data; only users association with the AP is recorded. Therefore, a technique for estimating user trajectories has to be devised. Kim et al. [2006] therefore, suggest three methods for extracting a physical movement trajectory from an AP trace: the *triangle centroid*, *time-based centroid*, and *kalman filter*. Additionally, the work describes a technique for extracting *pause time* at various locations. A *pause time* is the stay time of a user associated with an AP. The model also defines how to extract a *hotspot locations* from the WLAN traces. The main idea is to use the concept of pause time and assign a 2-D Gaussian distribution of popularity weight at each pause location, which creates a small mountain (peak of the Gaussian distribution) equivalent to the pause time duration. Later, the mountains are aggregated for all users, the hotspot regions are identified as those regions with popularity weight higher than a fixed threshold. Finally, a mobility model is proposed that reproduces the physical movement trajectory, pause time distribution, and hotspot localization similar to the empirical features in the studied trace. However, their user's transition probabilities computation does not consider location distances.

The trajectories of over 100,000 anonymized mobile phone users were analyzed by González et al. [2008]. In their effort to explore the statistical properties of the population's mobility patterns, a distance between user's positions at consecutive calls were measured. It follows that the distribution of the displacement over all users follows a truncated power-law. Nevertheless, modeling mobility using calls record has the problem of spatial and temporal bias measurement due to the variation in the placement of cellular towers. Additionally, two users with similar mobility dynamics might have different mobility ratings due to the variation in their call frequency.

Zignani and Gaito [2010] analyze few GPS-based traces to infer human mobility patterns. The use of GPS-based traces in the study has an impact on the granularity of the trajectories. The work proposed a clustering algorithm, which extracts geo-location from individual user traces. The clustering algorithm considers two points to be connected when the distance between them is less than 25 m as against 100 m set in Lee et al. [2008]. When distance that connect points within a cluster is small, more fine-grain clusters that enable characterization of movement features within a small location like library and conference room are generated.

A study on human mobility patterns from the traces of environments with a definite and highly organized structure, such as shopping malls, is proposed in Galati et al. [2015]. The traces consist of contacts between devices carried by people in the shopping mall. Finally, a routing protocol for DTNs is designed based on the statistical feature observed from the real traces. Additionally, they validate the proposed mobility model based on the user mobility characterization of the real traces. However, the model needs to be validated using location-based traces for a better mobility pattern prediction.

3.3 Synthetic Mobility Models

In this section, a brief review of the related works on synthetic models is presented. We grouped the related works into the following classes: The conventional models, pedestrian walk models, Pedestrian Mixed with vehicular models, and Obstacle-based models. The classification is made in such a way that each class shared similar features in terms of its purpose and the target mobility metrics. Suppose a model has multiple features such that it can belong to multiple classes. In that case, we focus on the model's main characteristics to place it in a more appropriate related class.

3.3.1 Conventional models

This class of models is targeted at modeling mobility in short-range, multi-hop wireless networks, i.e., MANETs. These models are generic, synthetic, and can be described using a set of mathematical rules.

One of the most widely used conventional models is the random walk mobility model Lawler and Limic [2009]; Clementi et al. [2011], which is characterized by the lack of intentional mobility that affects the notion of movement trajectory. Random walk models generate mobility patterns in which mobile nodes display a completely random behavior. With this regard, only a few wireless networks (e.g., sensor network for animal tracking Dominguez-Morales et al. [2016]; Sanchez [1998]) can display such kind of randomness. In contrast, the majority of wireless networks strictly obey certain mobility rules. The random walk is classified into discrete and continuous. In discrete random walks Clementi et al. [2011], the simulation environment is composed of points in n-dimensional space for all possible positions of a node in the environment. In continuous random walks, the simulation environment is a continuous sub-region of n-dimensional spaces Santi [2012]. Levy flight is an example of a continuous random walk. In the levy flight model, the length of steps is distributed according to a heavy-tailed probability distribution, specifically to a power law of the form $y = x^{-\alpha}$ with $1 < \alpha < 3$.

The random waypoint, Johnson and Maltz [1996]; Broch et al. [1998]; Chiang and Gerla [1998] on the other hand, is considered the first synthetic model that attempts to model the intentional human movement, which is not captured in the random walk models. It is implemented as the default mobility model in the most widely used network simulators: the Network Simulator (Ns2), Global Mobile Information System Simulator (GloMoSim), and Opportunistic Network Simulator (ONE) Tkk/Comnet [2013] as part of its movement models. Nevertheless, the movement patterns generated by the model consist of random locations separated by random distances. However, the movement pattern of humans is not completely random; instead, it shows dependencies Munjal et al. [2012]. Some simple fixes and modifications to the random waypoint presented in Yoon et al. [2003] still fails to capture a realistic behavior for intentional human mobility to some locations due to the strength of a social relationship or connection. For instance, a student going to the class for the lecture, going to the cafeteria to eat or visiting a friend at the nearby dormitory.

In the random walk and random waypoint models, the node's movement is not restricted to a pathway. The Manhattan mobility model Bai et al. [2003] on the other hand, restricts the movement of a node to the pathway in the simulation area.

We can easily integrate these models into wireless network simulator, but on the other hand, they do not capture an application-specific aspect of mobility (e.g., presence of geographical constraints like obstacles and road constraint in vehicular networks), and hence lacks realism.

3.3.2 Pedestrian Mobility Models

Lee et al. [2009] proposed Self-Similar Least-Action Walk (SLAW) that employs the concept of fractal waypoints, Least Action Trip Plan (LATP), and a walker model to generate regular patterns of daily human mobility. In the SLAW model, the movement of an individual is independent of any other person's movement. The model generates the fractal waypoints separated by gaps that follow a power-law distribution. Then, LATP is used to order travels between the fractal waypoints. The community (cluster) idea is addressed using a walker model that assigned some destinations to a node and restricts its movement within the destinations unless necessitated by a higher priority event. The SLAW model is based on daily routines like going to the office or attending a lecture. However, the model did not capture the time of occurrence of an event and the repetitiveness observed in the people's realistic daily activities. A variant of SLAW called Map-based SLAW is presented in Schwamborn and Aschenbruck [2013] which proposed geographic restriction to the SLAW model.

Munjal et al. [2011] presents the Simple way to model human mobility (SMOOTH). SMOOTH mimics real patterns of human mobility by relaxing an assumption of random mobility with a notion of a *mobility influence*, (i.e., node's mobility is influenced by factors like cluster size). The model studies seven Mobility statistical features: the flights (trip from one location to another without a pause or change of direction), inter-contact time, pause time, long flight due to popularity, closest mobile node visits, community interaction, and mobile node distribution. The simulation space in Munjal et al. [2011] is a free space without restricting obstacles that are not always realistic in an environment like a campus setting characterized by buildings of different shapes and sizes.

Small World In Motion (SWIM) Mei and Stefa [2009] is a simple mobility model designed to reproduce contact patterns between individuals. In SWIM, each node is assigned a home location chosen randomly and uniformly within the simulation area. In the beginning, the node does not know any location unless it visited it; once it visited the location, the location is added among the most popular location known by the node. The possibility of revisiting the location is higher than unvisited locations. *speed selection* is simple and straightforward such that once a node has chosen its next destination, it starts moving towards it following a straight line and with a speed that is proportional to the distance between the starting point and the destination. The *destination selection* for node i is described as follows: let h_i be the initial home of a node i. When i visited location C (i.e. a cell not a point), it meets other nodes at C represented by *seen(C)*. node i would assign a weight to location C as follows:

$$w(C) = \alpha.distance(h_i, C) + (1 - \alpha).seen(C) \tag{3.1}$$

where *distance*(h_i,C) is a function that decays as a power law as the distance between node i and cell C increases, α is a constant in [0:1], the larger is α, the more the node will tend to go to places near its home. The smaller is α, the more the node will tend to go to *popular* places. SWIM is evaluated theoretically and through simulation by testing its ability to reproduce human contact patterns. In Mei et al. [2011] a variation of SWIM is proposed aimed at capturing the user interest when associating home location to the node.

The Weighted random waypoint mobility model presented in Hsu et al. [2005] captures preferences in choosing a destination to characterize pedestrian mobility patterns in a campus environment. The main ideas behind the model are the observation that different buildings in a university campus have different degrees of popularity (*uneven waypoint popularity*). Similarly, such popularity evolve with time (*time-varying popularity)* and also *location dependent pause time distribution.* The model was calibrated with the data generated from a mobility survey of a random set of 268 students. In this model, the initial node position is computed based on the popularity using the Markov chain model; after spending the location-dependent pause time, the node chooses another destination based on the five-step Markov chain transition matrix continuous in this order for the time of the simulation. Though some of the assumptions in the model are realistic about the campus environment, the fact that all students behave identically is unrealistic since, in the real world, people display different stochastic mobility behavior.

Hsu et al. [2007] proposed the Time-Variant Community (TVC) mobility model for wireless mobile networks. The model is tuned to display empirical features observed from WLAN traces (i.e., *skewed location visiting* and *periodic re-appearance*) of nodes at the same location. The model addressed the concept of *skewed location visiting* by creating a community for each node; each node visits its community more often than other locations outside its community. The concept of community helps to model node heterogeneous movement patterns. It achieves *Period re-appearance* by defining two-movement periods called the *concentration movement period* and *normal movement periods*. At each time instant, a node is either in the *concentration movement period* or *normal movement period*. A node can only visit its community concentrated area in the *concentration movement period* while it can visit other places outside its community when in the *normal movement period*. The model was evaluated by comparing the theoretical results and the simulation results to the real traces of WLAN. They show that the model achieved its purpose of reproducing the observed features in the real traces (i.e., *skewed location visiting* and *periodic re-appearance*). The TVC model was evaluated using a custom discrete-time simulator with an unobstructed and unconstrained movement path, which does not reflect a more realistic movement domain.

Social mobility models were presented in Musolesi et al. [2004]; Herrmann [2003]. Hrabčák et al. [2017] presents a Students Social Based Mobility Model (SSBMM). The daily routine of student life inspired their work. The model distinguishes between the student's free time and the mandatory time upon which social and school activities are simulated. They compare their model with the classical random walk model, even though the random walk model cannot capture repetitiveness and heterogeneity of time and space.

The model under this category refers to as human walk models designed to models pedestrian movement in various environments. The main focus of walker models is the macro-mobility decisions, which include deciding the next destination to visit. Mobility is influenced by location popularity in the SLAW, SMOOTH, and SWIM models. Similarly, the TVC model mimics community using skewed location visiting and periodic re-appearance. All of the models assume that the movement area is free space without obstacles and the nodes are not restricted to defined movement routes.

3.3.3 Pedestrian Mixed with Vehicular Mobility Models

Ekman et al. [2008] presents a Working Day Model (WDM) for DTNs that mimics the workers' daily activities like going to the office, going for evening activities, or returning home. The model uses *map-based movement* on the concept of *sources-destination.* It also uses a timescale to switch between different *submodels.* The WDM model has four main submodels (i.e., *home activity, office activity, evening activity, and transport*) submodels. The *transport* submodel has three submodels (i.e., *bus, car, and walk*) submodels that a node chose to move between the home, office, and evening activity places. The choice of a mode of transport depends on the probability of owning a car. Depending on the preferred mode of transport, a node move from its current location toward the destination with a constant speed through the shortest route calculated using the Dijkstra algorithm. The WDM was implemented as an extension to the Opportunistic Network Environment (ONE) simulator Tkk/Comnet [2013] as a combination of many mobility models controlling the movement of the nodes going to work, their homes, and meeting their friends. It was validated by showing that the distribution of the *contact duration, inter-contact duration, and contacts per hour* of the synthetic traces generated by the model resembles that of iMote traces from the Cambridge university experiment. The model does not cover the impacts of a floor, walls, and other constraints which affect nodes mobility.

An extension to WMD is presented in the real-life mobility model (RLMM) Gorawski and Grochla [2013]. The model differentiates between the working days and the weekend day. The weekly periodic repetitive features were achieved by defining multiple states and their influence on users' mobility and behavior. The steps described in the model are at-home, at-work, at-popular-place, regular-travel, and alternative-travel. Every node has a fixed initial location called home, at which it starts its transition to visit other fixed locations like a workplace. The node often takes a longer travel distance during the weekend, which gives it power law-like characteristics. The travel speed is selected according to the commute distance with the pedestrian movement (1.25m/s), the fast pedestrian or cyclist movement (5m/s), the car/bus slow movement (8.3m/s), and the car/bus fast movement (11.1m/s). The model was evaluated in BonnMotion tool for generating different mobile scenarios Aschenbruck et al. [2010] using inter-contact times and some metrics defined in the BonnMotion project like the average degree of temporal dependence, the average degree of spatial dependence, total links, and etc. However, the model does not consider movement restrictions on maps and does not involve social interactions.

Zheng et al. [2010] introduced the Agenda Driven Mobility Model (ADM) that is used to simulate a mobile ad hoc network in an urban scenario. In ADM, the node's movement solely depends on its agenda, which describes its activities and locations of the activities. The individual agenda is generated from various distributions derived from the data obtained from the National Household Travel Survey (NHTS) (U.S. Department of Transportation, 2007). The geographic information is inserted into the model in the form of maps, roads, locations, and addresses extracted from WLAN traces, or real maps from GIS database like TIGER (the U. S. Census Bureau's Topologically Integrated Geographic Encoding and Referencing database). Each node moves at speed decided by the road traffics. The model was compared to the Random waypoint in terms of routing performance metrics like delivery ratio and average path. More evaluation with the realistic traces would make a great contribution.

These types of models integrate human walk models and vehicle movement models. They aimed at mimicking the decision of a pedestrian to move using alternative methods of transport such as walking, cycling, cars, or bus. The models assume restriction of movement on roads to improve realism. The models do not assume obstacles like standing pedestrian or road diversion and therefore lack the mechanism for obstacle avoidance.

3.3.4 Obstacle-based Models

The Obstacle Mobility (OM) model proposed by Jardosh et al. [2005] models the environmental obstructions which affect both movement and signal propagation. In this model, the node's paths and destinations are constructed from a Voronoi diagram based on the obstacle position on a campus-like simulation area.

As an extension to Jardosh et al. [2005] is proposed in Papageorgiou et al. [2009], the model allows nodes to move around the obstacle but not limited to a defined path. The model only considers rectangular obstacles that limited its ability to capture the realistic feature of an environment with obstacles of different shapes and sizes.

A random obstacle-based mobility model for DTN is proposed in Wu et al. [2011]. In this model, the node moves from the initial location to the destination via the shortest path if there is no obstacle along the path; otherwise, the node recursively selects the other node's location close to the obstacle and moves forward. This operation is repeated until the node reaches its destination. This model considers obstacles with a rectangular shape. Similarly, in the absence of a node close to the obstacle, an unnecessary trip would be made, especially when the destination is just behind the obstacle.

Wang et al. [2017] proposed an obstacle-based mobility model that generates a smooth trajectory of a Bezier curve for escaping obstacles. Human mobility trajectories for escaping obstacles like building or road diversion are not always smooth curves in real scenarios. In addition to that, the model did not capture movement to attraction factors such as points of interest which represents human social behavior.

These models' movement area is characterized by obstructing bodies of different shapes and sizes. The mechanisms for escaping obstacles are integrated as part of the model. However, most of these models do not assume restriction of movement on the road.

3.4 Conclusion

We present background knowledge on the use of real traces to develop synthetic mobility models. We complement our background study with a comprehensive overview of the related works on synthetic mobility models in the form of classes. The classification of synthetic mobility models with related features would ease access to the existing models for research purposes.

Moreover, trace-based models exploit real mobility traces while synthetic models can be run independently from real mobility traces. Nevertheless, we can have a more realistic model by calibrating the parameters of some synthetic models based on real-world mobility traces. However, a characterization of the mobility features model is the first step toward achieving this goal. In the next chapter, we characterize mobility features from the location and contact-based real traces.

Chapter 4

Characterization of Human Mobility

4.1 Introduction

Due to the dynamic nature of human mobility, a thorough investigation of its characteristics features is in tremendous need. Despite the fact that various aspects of human mobility patterns have been characterized in recent studies, yet due to the difficulty to acquire large-scale human mobility data, several features do not get the required consideration. The features that require a thorough investigation include some of the fundamental features like travel distance, pause time, and intra-cluster features such as the intra-cluster travel distance, and direction of movement in the cluster. People move around their physical environment (whole domain), but the movement concentrates around their community or point of interest. Consequently, one or more movement clusters are formed over time.

However, our primary focus is on the characteristics features within the movement cluster (*intra-cluster features*), though we explicitly analyze some of the fundamental features for the whole domain (*inter-cluster features*), which include: the connectivity features (i.e., contact and inter-contact time), the spatial feature (i.e., travel distance), and the temporal feature (i.e., pause time). The intra-cluster feature like *direction of movement* has not been given much attention by the researcher community despite its impacts on the human mobility patterns. To the best of our knowledge there is a lack of research on this feature, all of the previous works either investigate the direction of movement for the whole domain Kim et al. [2006]; Nunes [2012] or assume the movement to be random within the cluster Hsu et al. [2007]. The concept of community in our study is the location at which a user spent much of his time exploring neighboring locations. Therefore, the concept should not be confused with that in a social community, which mean a group of people sharing physical location, ideas, or common goals.

We use the daily GPS track log collected from two different university campuses (NCSU and KAIST) for the location-based trace Rhee et al. [2009]. Garmin GPS 60CSx handheld receivers are used for data collection which are WAAS (Wide Area Augmentation System) capable with a position accuracy of better than three meters 95 percent of the time, in North America. The GPS receivers take reading of their current positions at every 10 seconds and record them into a daily track log. The data is available at Henderson [2020]. We are interested in the stationary locations at which users stay.

For the contact-based trace, we use the Bluetooth encounters between mobile nodes from the Cambridge-city-students iMote experiment Scott et al. [2009]. The data consist of 10641 contacts between iMote devices carried by students for the duration of about 11.43 days. The data is available at Henderson [2020] repository. We are interested in the duration at which two devices are in contact with each other (contact duration) and the time between two consecutive contacts between two devices (inter-contact time).

4.2 Extracting Dynamic Clusters from the GPS Tracklogs

In this section, we explain the procedures for extracting clusters from an individual trace at which the user spent a long time exploring neighboring locations. Generally, human trajectories can be viewed as a mixture of *stationary* and *movement locations*. A *stationary location* is a fixed position at which a user stay with zero velocity. A node spends more time at the *stationary location* while only spend a small amount of time at the *movement or transit location*. We focused on the *stationary locations* because it is in the stationary location that the user decides to visit the next location closed to it or far away from it. Even in the set of *stationary locations*, we are more interested in those connected by a distance less than a certain threshold, which we refer to as dynamic clusters.

The GPS coordinates that belong to the transit locations at which a user pauses briefly on its way to the destination do not belong to the stationary locations and therefore need to be removed from the trace. Section 4.2.1 explains the method used to remove the transit locations from the trace.

4.2.1 Removing Transit locations

Some of the coordinates from the GPS traces do not belong to *stationary locations* rather they belong to the transit locations at which user stays briefly on its way to its destination. So we use a simple rule to identify locations at which a node is in transit. We set a speed threshold Δ_s such that if a node speed between two GPS positions is greater than the threshold Δ_s, we remove the next position from the original trace since the position is part of the transit points. We set $\Delta_s = 0.5$ m/s to remove positions at which a user is on transit. This value is reasonable considering the time granularity of 30 seconds in the studied trace and the assumed average human velocity of 1.3 m/s in the literature. We further extract the actual waypoints by deleting all transit locations at which a node pause time is less than the time threshold Δ_t. These locations are also considered as part of the transit locations at which a node stays on its way to the destinations. After this processing, the output trace consist of only sequences of node stationary locations. In the next subsection, we extract locations that belong to a common geographic cluster where the user stay in the same coarse location.

4.2.2 Clustering

we need to extract movement clusters from the stationary locations associated with each user. Several methods of clustering exist for grouping similar elements into a common cluster Xu and Wunsch [2008]. We use the hierarchical method in our study . The two hierarchical clustering methods are divisive and agglomerative clustering; the divisive clustering is good for identifying large clusters while the agglomerative clustering is for identifying small clusters. We opted for the agglomerative method since we aim to get insights into the evolution of small clustered locations at which a user takes intentional (not random) steps close enough to be considered its new evolved cluster over time. Small clusters enable us to understand the movement patterns within small locations like lecture rooms or restaurants. The hierarchical algorithm is explain as follows:
Given a set of N items to be clustered and an NxN distance (or similarity) matrix, the basic processes of hierarchical clustering are Johnson [1967]:

1. Assign each item to its own cluster so that if you have N items, you now have N clusters, each containing just one item. Let the distances (similarities) between the clusters equal the distances (similarities) between the items they contain.

2. Find the closest (most similar) pair of clusters and merge them into a single cluster so that now you have one less cluster.

3. Compute distances (similarities) between the new cluster and each of the old clusters.

4. Repeat steps 2 and 3 until all items are clustered into a single cluster of size N.

We run the above algorithm using a single linkage method, sometimes called connectedness or minimum method, and created location clusters based on the similarity of the closest pair of locations. A trajectory point is connected to another point if the distance between them is less than 20 m. This enable us to generate clusters at fine grain to better analyse movement patterns at smaller locations like a lecture room or restaurant.

4.3 Estimation of Pause time from Real Traces

Pause Time is the stay time of a user associated with a location. To extract a *Pause time* from the GPS mobility traces, we follow the following procedure:

1. Pre-process GPS tracklogs to extract the real stationary locations.
2. Estimating the *elapse time* (i.e., the sum of the pause time and the duration of travel)
3. Estimate *min* and *max* speed
4. Estimate node speed
5. Evaluate the pause time using Equation (4.6)

The pre-processing technique of extracting stationary locations is described in section 4.2.1. Next, estimating the elapse time, e_i between two consecutive points p_i and p_{i+1} is very straight forward, since our trace has timestamp at each location. The elapse time between two points is the difference of the time t_{i+1} at position p_{i+1} and t_i at position p_i respectively:

$$e_{i+1} = t_{i+1} - t_i \tag{4.1}$$

The speed between two consecutive encounter with time t_i and time t_{i+1} is.

$$speed_i = \frac{dist_i}{time_i} \tag{4.2}$$

Moreover, the average speed of pedestrians in the educational environment was evaluated in Rastogi et al. [2011], which depends on the walking area like sidewalks, wide sidewalks, and precincts. We assume the students at the KAIST and NCSU campuses walk using sidewalks, which has an average speed of approximately 1.20m/s, So we set the min_s and max_s speed at 0.2m/s and 2.2m/s respectively.

$$min_s \leq speed \leq max_s \tag{4.3}$$

At each interval, we test the speed value, and if the return speed is within the range, we assume that the node does not pause at the observed location. If the return speed is less than the *min* speed, we assume that the node paused at that location. If the computed speed is greater than the *max* speed, we ignore the interval as it does not reflect reality, or the node uses alternative means of transport like a bike.

We evaluate node speed at the position p_i to p_{i+1} as the difference between the Euclidean distance between the two locations and time.

$$\begin{aligned} speed_i &= \frac{distance_i}{time_i} && (4.4) \\ &= \frac{\parallel p_{i+1} - p_i \parallel}{\parallel t_{i+1} - t_i \parallel} && (4.5) \end{aligned}$$

Finally, the pause time is evaluated as the difference between the elapsed time and travel duration:

$$pauseTime_i = e_i - travelDuration_i \tag{4.6}$$

$$= e_i - \frac{dist_i}{\overline{speed}_i} \tag{4.7}$$

Where $\overline{speed}_i$ represents an average speed evaluated by exponential smoothing on the current speed $speed_i$ and the previous estimates $\overline{speed}_{i-1}$:

$$\overline{speed}_i = \lambda * speed_i + (1 - \lambda)\overline{speed}_{i-1} \tag{4.8}$$

4.4 Fitting Distribution to the Empirical data

In this section, we explain the procedures follows to identify the distribution that best fits the empirical data of the *contact duration*, *inter-contact time*, *travel distance*, *pause time*, and *intra-cluster angle of movement*. We aim to determine the appropriate distribution that best fits our empirical data to estimate the value(s) of parameter(s) for the distribution and calibrate them in the proposed mobility model. We used Kolmogorov-Smirnov (KS) Goodness-of-Fit Test in *poweRlaw* R package Clauset et al. [2009] to decide if a sample comes from a population with a specific distribution. The *poweRlaw* R package implements maximum likelihood estimators for a variety of heavy tailed distributions. It uses goodness-of-fit based approach to estimate the lower cut-off for the scaling region. The Kolmogorov-Smirnov test is based on the empirical distribution function (ECDF). Given N ordered data points x1, x2, ..., xN, the ECDF is defined as

$$EN = \frac{n(i)}{N} \tag{4.9}$$

Where $n_{(i)}$ is the number of points less than X_i and the X_i is ordered from the smallest to the largest value. It is a step function that increases by $\frac{1}{N}$ at the value of each ordered data point.

Generally, we start with the choice of distributions family, then we estimate the parameter(s) of the best fitted distribution using goodness of fit statistical tests.

4.4.1 Choice of distributions family

Depending on the nature of empirical data, an appropriate distribution family that fit the uncensored data needs to be determined at the initial stage. Several methods can be used as a guide, including graphical techniques or analytical expressions such as Pearson's K criterion. At the first step, we applied the graphical techniques to select the distribution that fits the empirical data, using four continuous distributions: the *weibull, log normal, exponential, and power law* in our evaluation. Seldom the graphical view may not clearly determine the most fitted distribution, so we proceed to the *KS Goodness-of-Fit Test* to determine the most fitted distribution.

4.4.2 Estimating Distribution Parameter(s)

After choosing the parametric distribution that can mathematically represent our empirical data, we need to estimate the parameters of such distribution. Several methods exist in the literature, but in this study, we deployed the method of *Maximum Likelihood Estimation* and *KS Goodness-of-Fit Test.* The first method begins with the mathematical expression known as a likelihood function of the empirical data; the mathematical expression contains the unknown parameters. We need to evaluate the parameters' values that maximize the sample likelihood, otherwise called Maximum Likelihood Estimation (MLE). In the second approach, we match empirical frequencies with the parametric model that fits the data (e.g., the power law, log-normal, Weibull, and exponential) distribution.

We used the Kolmogorov-Smirnov test as follows:

H_0: The empirical data follow a specified distribution

H_1: The empirical data do not follow the specified distribution.

$$C = \max_{1 \le i \le N} \left(F(X_i) - \frac{i-1}{N}, \frac{i}{N} - F(X_i) \right) \tag{4.10}$$

Where F is the theoretical cumulative distribution of the continuous distribution such as power law, log-normal, Weibull, and exponential distribution.

We reject the null hypothesis regarding the distribution only if the test statistic C is greater than the critical value obtained from the KS table in Appendix F. We use 0.05 significant level throughout the estimation. The critical value is evaluated for each distribution using the *ntrails* (number of observations) from the KS table and compared with the respective test statistics C derived using a bootstrap_p() function in *poweRlaw* package of R software Gillespie [2015].

4.4.3 Inter-Cluster Travel Distance Distribution

The *travel distance* is one of the fundamental spatial features of human mobility. Our goal is to extract sequences of geographic displacements users travel during a time period. These sequences of displacements are helpful to the understanding of the travel schedules of an individual. We explain how we generate sequences of destination waypoints in Section 4.2.1. To extract the distances between these locations for the whole domain, we measure the Euclidean distance between consecutive locations for each individual trace.

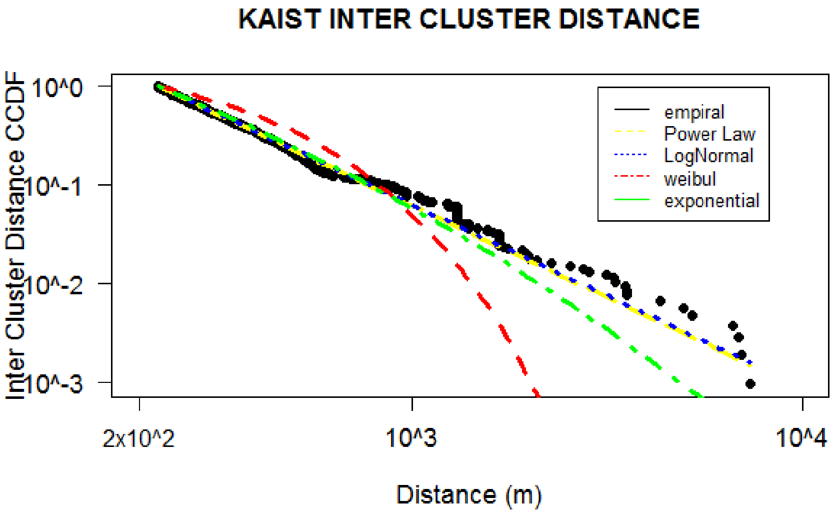
We start with the KAIST trace, its aggregate distribution of the inter-cluster travel distance, and the fitted parametric distributions are shown in Figure 4.1a. After fitting, we evaluate how best the parametric models for the class of Log-normal, Power-law, Exponential, and Weibull fit our empirical data. This is done by examining the graphical view of the distributions, and when the graphical method fails to distinguish the best distribution that fits the empirical data clearly, we proceed to other techniques like Maximum Likelihood Estimation or KS Goodness-of-Fit Test (see Section 4.4 for more details). Following this procedure, the KAIST inter-cluster travel distance distribution, which reflects the whole domain, follows a power-law distribution with the exponent α = 1.735462. The Ks test for the inter-cluster distance is shown on Table 4.1

Figure 4.1b shows the power-law distribution with an exponential cut-off fits to the KAIST inter-cluster distance. The power-law feature of travel distance indicates that people usually visit close locations more often than far away locations. This feature also gives more insight into the students' mobility patterns; we can understand that many students visit closer popular locations like cafeterias, sports fields, shopping malls, and friends that are within similar geographical locations, while only few student frequently visit far away locations.

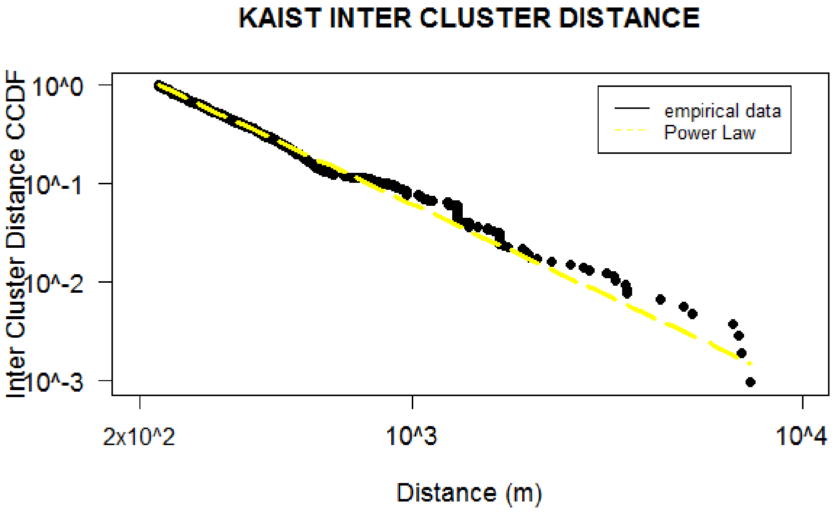
Similarly, for the NCSU trace, the distributions of inter-cluster travel distance and the parametric models are plotted in Figure 4.2a. We determine that its inter-cluster distance follows exponential distribution with the parameter $\lambda = 0.01227271$ as shown in Figure 4.2b. We observed that the limited number of traces in NCSU traces influenced the exponential decay of the inter-cluster distribution.

$Dist$	gof	$ntails$	$crit.Val.$	$Remark$
lognorm	0.02613176	6693	0.0166	Rejected
power law	0.03166393	1062	0.0417	Accepted

Table 4.1: KAIST Inter-Cluster Distance gof Table

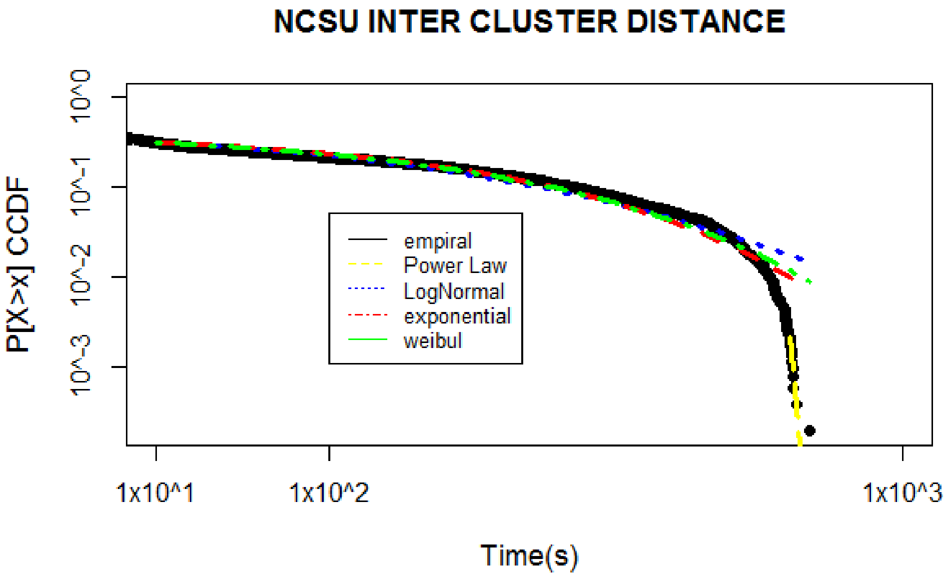


(a) Four (4) distributions fit to the KAIST inter-cluster distance trace

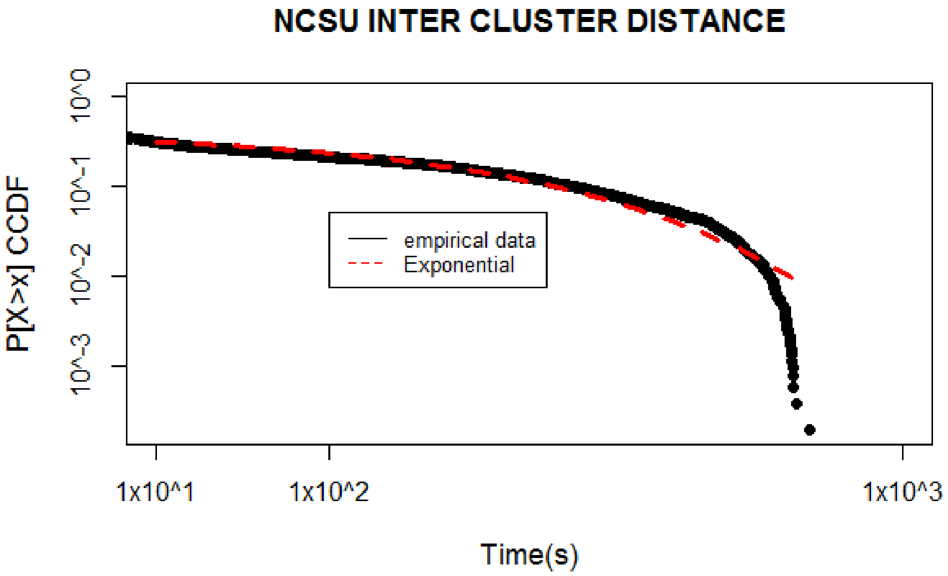


(b) Power-law fits to the KAIST inter-cluster distance trace

Figure 4.1: Figure 4.1a shows four (4) different distributions that fit the KAIST trace inter-cluster distance, the power-law, and log-normal distributions closely fit the empirical data more than the Weibull and exponential distributions. Finally, the distribution for the inter-cluster distance of the KAIST trace is found to be power-law distribution with an exponential cut-off as shown in Figure 4.1b.



(a) Four(4) parametric models fitted to the NCSU Trace



(b) The exponential distribution fits to the NCSU Trace

Figure 4.2: Figure 4.2a shows four (4) different distributions that fit to the NCSU inter-cluster distance. From the graphical view, it appears that the empirical data matches with all distributions except power-law but after KS-test, the data follows an exponential distribution except at the tail as shown in Figure 4.2b.

4.4.4 Intra-Cluster Travel Distance Distribution

In this section, we focused on the travel distances between consecutive locations within a cluster at which a node spent a long time exploring neighboring locations. We observed that at least two or more of such clusters evolve overtime on each individual daily trace.

We emphasize that a clear understanding of the Spatio-temporal features of the dynamic clusters would enormously help in characterizing human mobility and would be a great tool for the design of realistic mobility models, which serve as a tool for the network design and protocols performance evaluation.

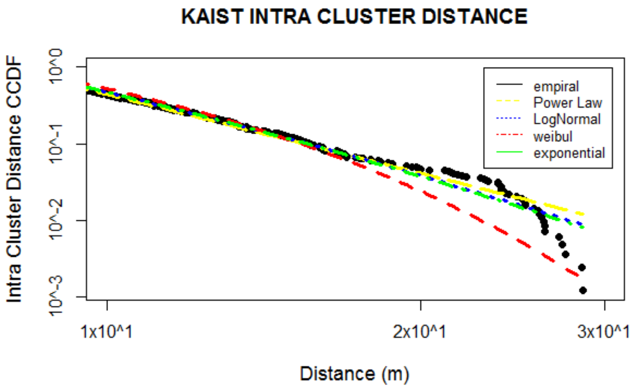
$Dist$	gof	$ntails$	$crit.Val.$	$Remark$
lognorm	0.01564494	815	0.04764	Accept
power law	0.03181678	1900	0.03120	Reject

Table 4.2: KAIST Intra-Cluster Distance gof Table

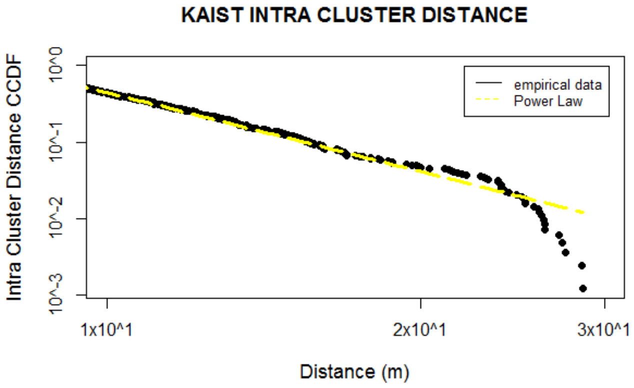
After fitting the parametric models with the observed KAIST intra-cluster travel distance data, the distribution that best fits the data turns out to be log-normal distribution with the parameters 2.29989493, 0.8685148 for the log mean and log standard deviation, respectively, as shown in Figure 4.3c and KS test on Table 4.2. This shows students take repeated short walks around some popular locations like classes, libraries, dormitories.

Moreover, for the NCSU trace, a similar distribution that follows a log-normal distribution with the parameters 3.1651969 and 0.5044682 for the log mean and log standard deviation, respectively, were observed. Figure 4.4a shows the four parametric models fitted on the empirical data, and Figure 4.4b visualized the best-fitted distribution after the KS Test.

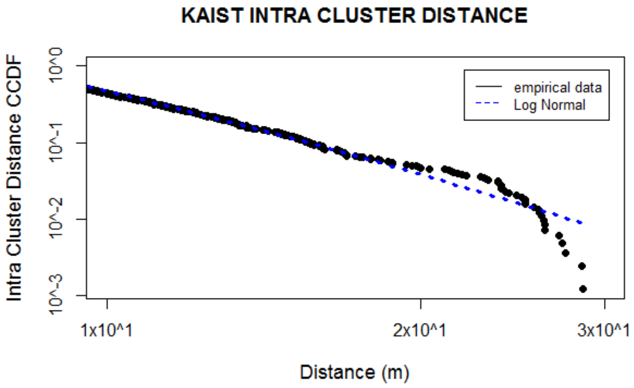
To get more insight into the direction of the cluster short walks, we study the angle of movement in each dynamic cluster in the next section.



(a) Four (4) distributions fit to the KAIST intra-cluster distance data

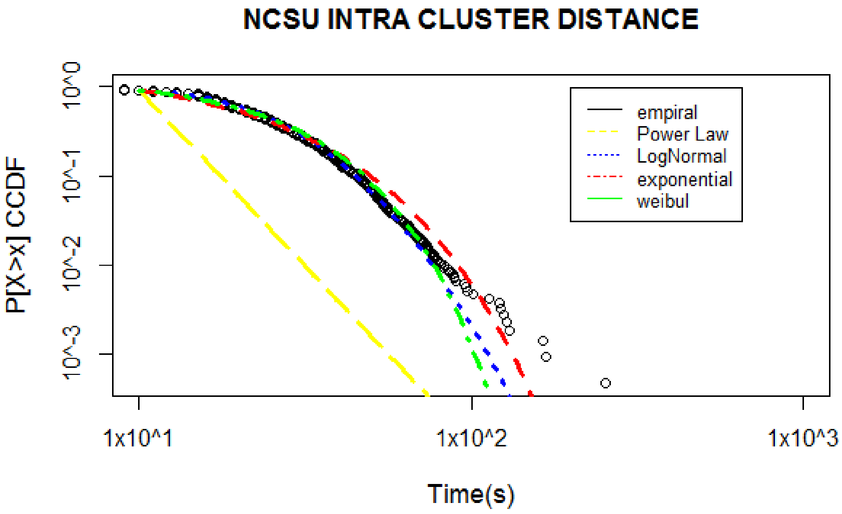


(b) Power-law distribution fits to the KAIST intra-cluster distance data

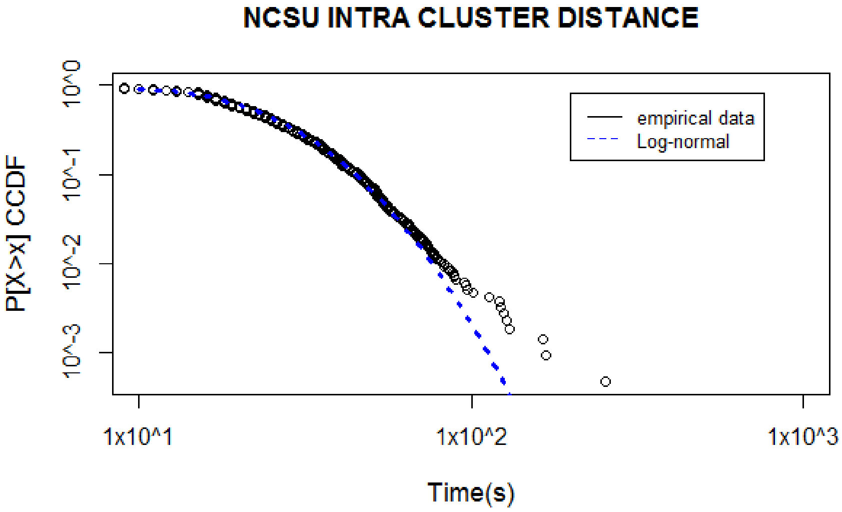


(c) Log-normal distribution fits to the KAIST intra-cluster distance data

Figure 4.3: Figure 4.3a shows four (4) different distributions fitted to the KAIST intra-cluster distance trace; all distribution fits the empirical data, but power-law and log-normal show better matching features. Figure 4.3b and Figure 4.3c visualize the two related distributions. Finally, the distribution for the intra-cluster distance of the KAIST trace is found to be log-normal distribution as shown in Figure 4.3b.



(a) Four (4) distributions fit to the NCSU intra-cluster distance data



(b) Log-normal distribution fits to the NCSU intra-cluster distance data

Figure 4.4: Figure 4.4a shows four (4) different distributions fitted to the NCSU intra-cluster distance. We can see that the empirical data closely follows the log-normal and Weibull distribution. Still, after a goodness-of-fit test, we found that log-normal distribution is more plausible than the Weibull distribution. Figure 4.4b shows the log-normal model fitted to the empirical data.

4.4.5 Intra-Cluster Direction of Movement

The direction of movement within clusters has not been given attention by the research community despite its impacts on the mobility patterns within the community. Nevertheless, some related works reported an aggregate distribution of movement directions for the whole domain instead of movement clusters Hsu et al. [2007]; Nunes [2012]. This study takes a different approach by investigating the direction of movement in dynamic clusters to understand the realistic movement directions within the clusters.

Figure 4.6 shows the distribution of the movement direction within clusters from NCSU trace with a bin size of 1°. We measure the direction of each movement by its movement duration. We can see that the direction of movement is biased symmetric toward some preferred directions. It shows that the movement within a dynamic cluster favors some direction of the popular locations. The symmetry in the distribution was expected due to the possible return of nodes to their main locations after exploring neighboring close-by points of interest. We can also deduce that students visit common locations around their homes and classes, which resulted in the similar aggregated distribution of angles with bias symmetry to angles between 90°-150° and 240°-330° respectively. We can see that nodes move to other sides as well but with smaller frequencies than the direction of the point of interest. It implies that geographical restrictions like constraint movement on roads are not the driving factor for the bias symmetry of the movement angle distribution. Similarly, the aggregate distribution for the whole domain is shown in Figure 4.5. The figure shows the distribution is similar to that of the normal distribution; though it has a small symmetry shape, but the direction of movement is almost uniformly distributed within the domain.

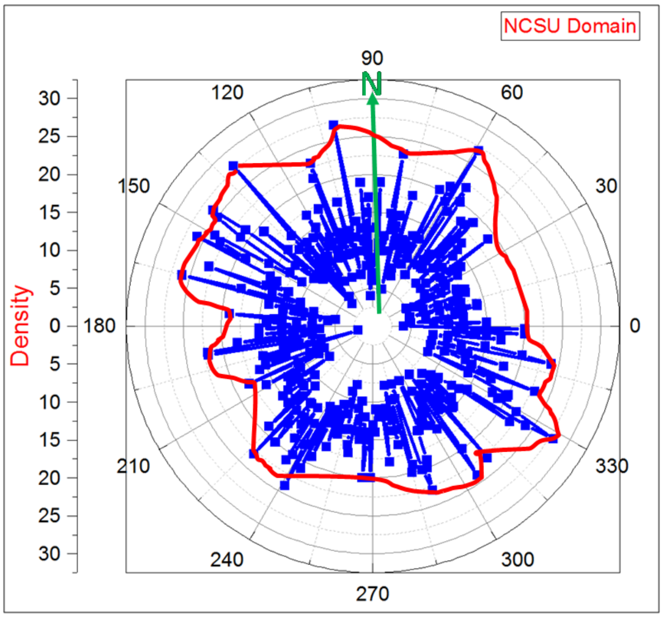


Figure 4.5: The uniform distribution of Direction Angle for the whole domain (NCSU trace). The x-axis represents the Angular (Units are in degrees) and y-axis is the density of movement toward a given direction. The bin size is 1°. Each direction is weighted by the duration of its movement.

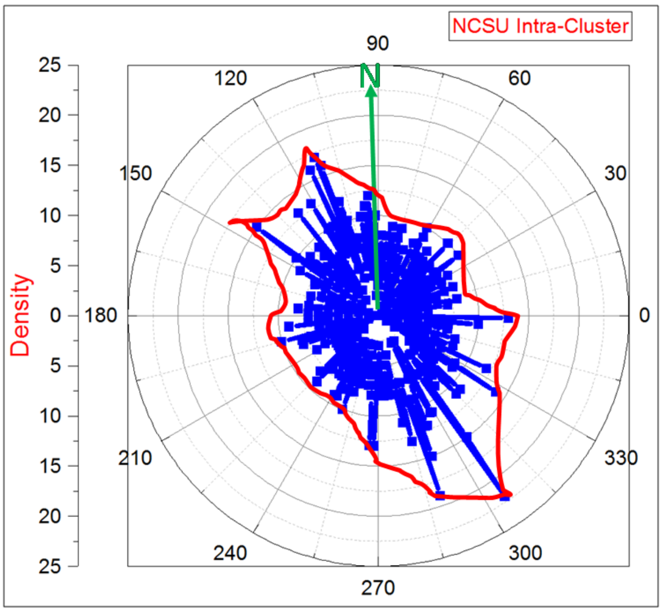


Figure 4.6: The bias symmetry distribution of Direction Angle for the dynamic clusters (NCSU trace). The x-axis represents the Angular (Units are in degrees) and y-axis is the density of movement toward a given direction. The bin size is 1°. Each direction is weighted by the duration of its movement.

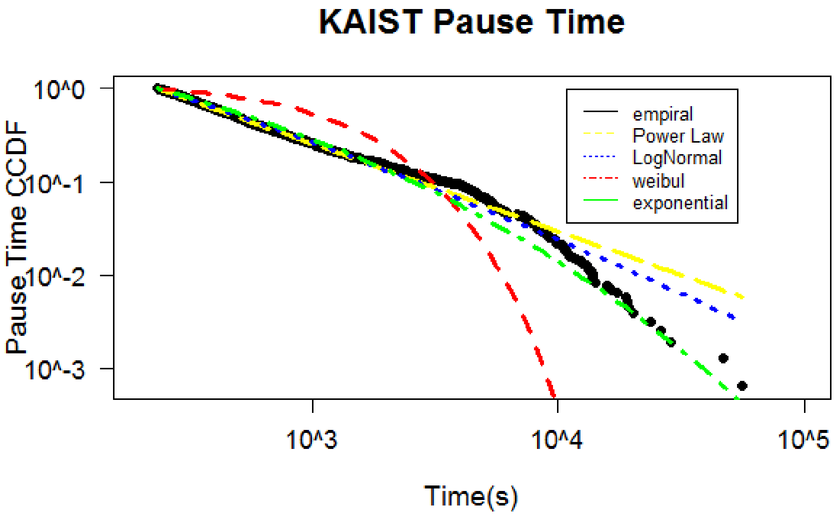
4.4.6 Pause Time Distribution

Pause time distribution is one of the temporal features of human mobility, which plays a vital role in the diffusive nature of human mobility. It dictates the amount of time a node spent at a location with zero or close to zero velocity. Though it is assumed to be a random value chosen from a uniform distribution in some conventional models like Random walk, recent studies have presented different results suggesting a fat-tail distribution González et al. [2008] [Song, SLAW]. Figure 4.7a shows the pause time distribution for the KAIST campus empirical data and the four parametric models. After the KS test we found that power-law distribution is plausible, and hence there is no enough evidence to support its rejection, and hence we accepted the hypothesis that the empirical data follows a power-law distribution. It is shown graphically in Figure 4.7b. The KS test is also shown in Table 4.3. We can see that the power-law has a threshold value (xmin) of four minutes(240s), cut-off value of P(Δt) = 16hrs, and exponent value $\alpha = 1.94$. The power-law pause time distribution indicates a scale-free characteristic.

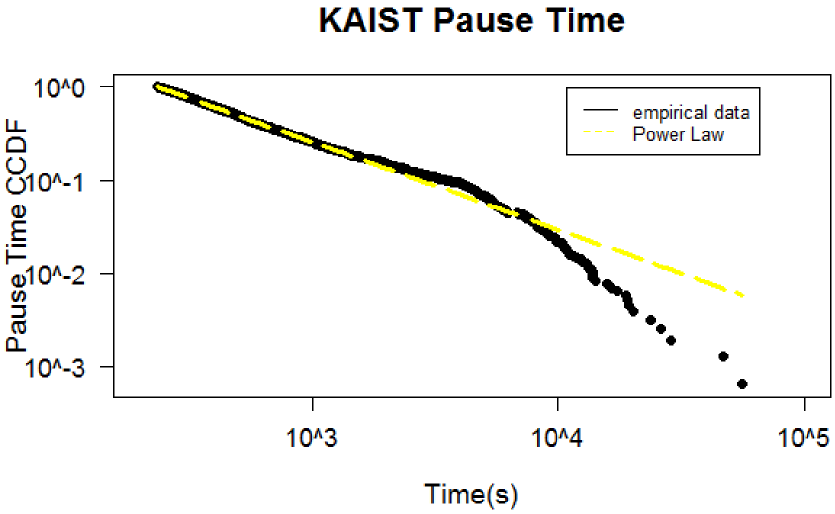
However, the NCSU trace has a slightly different distribution with the same power-law but with an exponential cut-off. The value of the exponent $\alpha = 1.97$. We can see from Figure 4.8b how an exponential decay out-weight the power-law decay toward the end of the distribution. We suspect the limited number of traces in the NCSU dataset to have contributed to the exponential decay. This power-law feature in the NCSU data distribution is interesting as it implies the scale-free characteristic, just like in the KAIST trace. In Table 4.4, the results of KS test is shown.

Dist	*gof*	*ntails*	*crit.Val.*	*Remark*
power law	0.02367702	850	0.04665	Accept
exponential	0.1315407	386	0.06922	Reject

Table 4.3: KAIST Pause Time gof Table

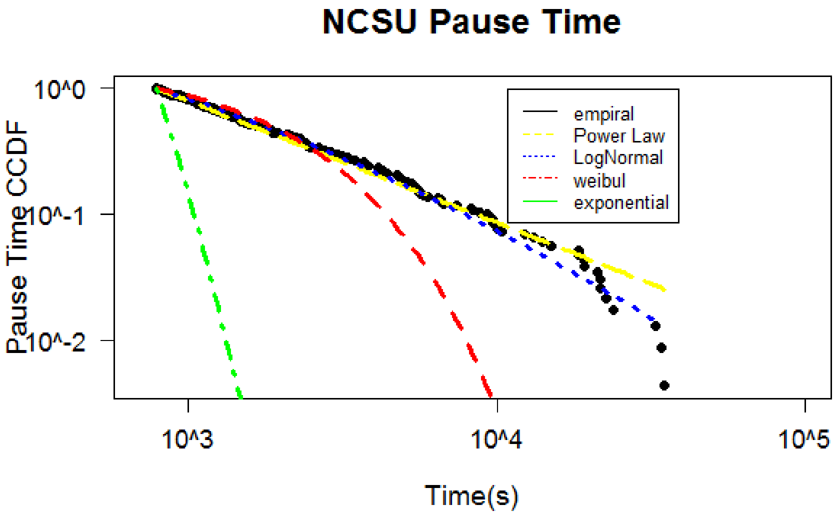


(a) 4 Distribution fit to KAIST Trace

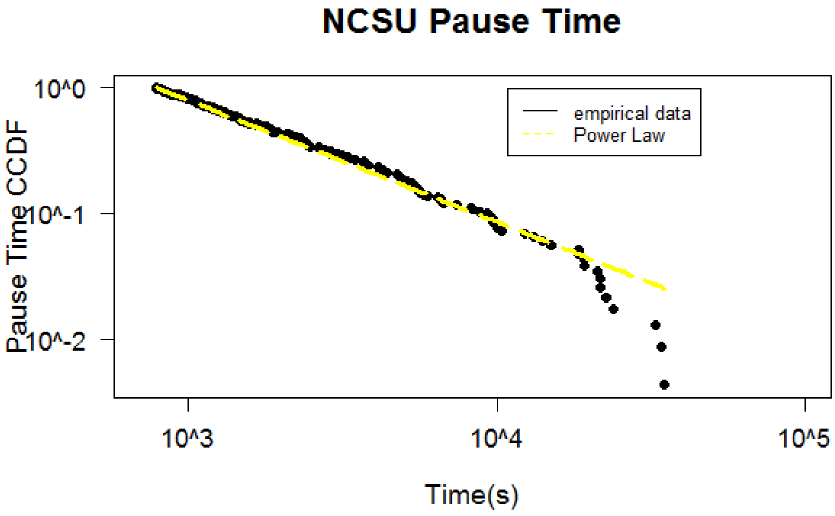


(b) Power Law fit to KAIST Trace

Figure 4.7: This figure shows the Pause time distribution for the KAIST trace. All parametric models are fitted to the empirical data in 4.7a. The power law distribution that is more plausible than other distributions is shown on 4.7b



(a) 4 Distribution fit to NCSU Trace



(b) Power Law fit to NCSU Trace

Figure 4.8: This figure shows the Pause time distribution for the NCSU trace. All parametric models are fitted to the empirical data in 4.8a. The power law distribution is more plausible than other distributions as shown on 4.8b

$Dist$	gof	$ntails$	$crit.Val.$	$Remark$
Power law	0.04570027	850	0.04665	Accept
exponential	0.14460256	189	0.09893	Reject

Table 4.4: NCSU Pause Time gof Table

4.4.7 Analysis of the Connectivity Features

This section characterize the connectivity features observed in the empirical data from iMotes experiments at the Cambridge university Scott et al. [2009]. The data contain traces of Bluetooth sightings by groups of users carrying small device (iMote) for five days.Our goal is to extract the distribution of *contact duration* and *inter-contact time* from the dataset for further analysis. It follows that in Opportunistic Networks, each contact between the network devices is assumed to increase the possibility of forwarding a bundle closer to its destination. In this regard, an in-depth understanding of the statistical properties of contacts and inter-contacts time is a great tool for designing and analyzing networks with intermittent connections.

The contact duration measures the time elapsed between the start and end of contact between two devices, while the inter-contact time represents the time between two contacts between two devices. Interestingly, the data is processed in a format that makes it easier to extract the features using simple heuristics. Figure 4.9 shows the aggregate CCDF distribution for the *inter-contact duration* of the empirical data. The distribution follows a power-law distribution with the exponent $\alpha = 1.4$, but the power-law decay is overweight by an exponential decay toward the end of the distribution. The distribution is called a truncated power law, which is similar to the results presented in Lee et al. [2009]. The power feature of the inter-contact time distribution is interesting because it dictates the scale-free properties of an opportunistic network. This power-law condition has been proved in Chaintreau et al. [2007] to impact a finite delay for all oblivious forwarding algorithms as long as the exponent is greater than 1, which is the case in Figure 4.9

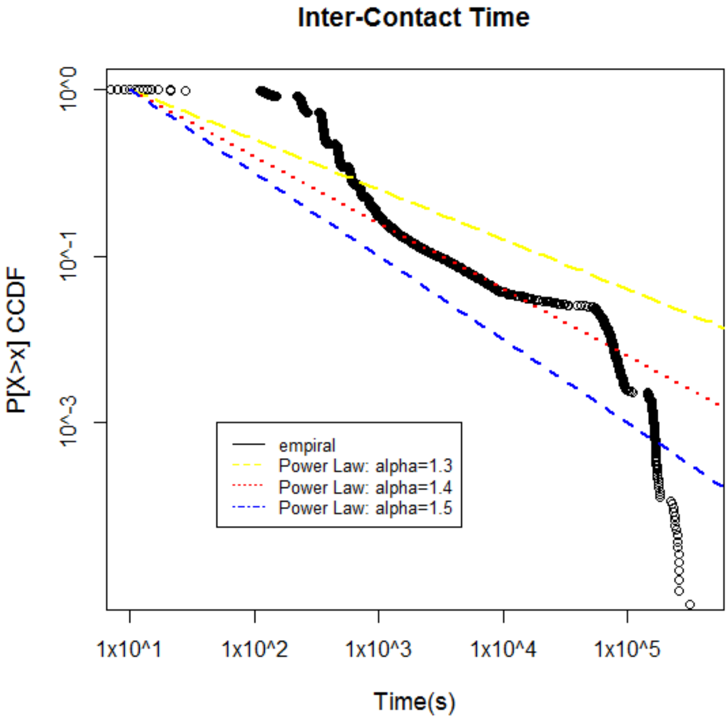


Figure 4.9: shows power-law distribution with different values of lambda fitted on the Cambridge iMotes trace Inter-contact time distribution.

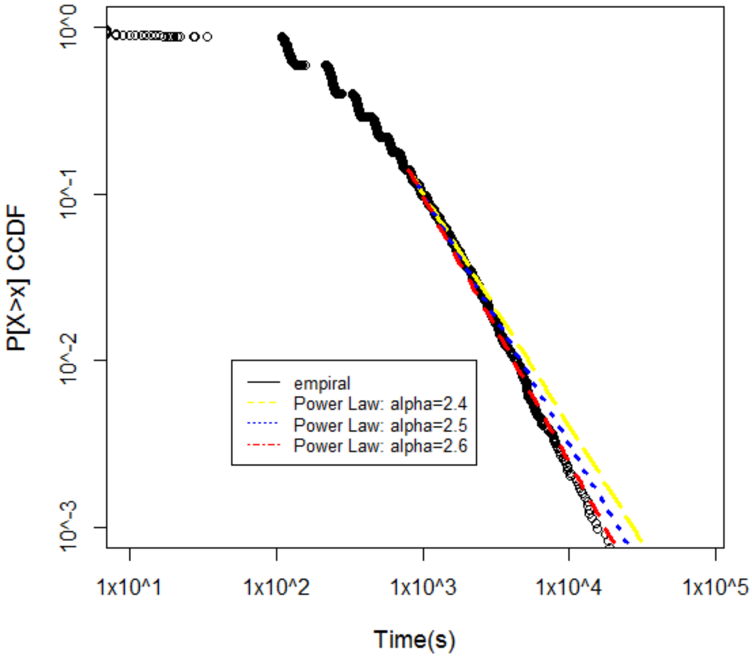


Figure 4.10: shows power-law distribution with different values of lambda fitted on the Cambridge iMotes trace Contact time distribution

Figure 4.10 shows the aggregate CCDF distribution for the *contact duration* of the empirical data. We can see that the distribution of the contact time follows power-law distribution after almost 13 minutes. The distribution for the shorter contacts follows an exponential distribution, but after few minutes, the distribution follows a power-law distribution with the exponent $\alpha = 2.5$.

4.5 Conclusion

We conduct a characterization of human mobility's spatial, temporal, and connectivity features using real traces. To achieved this, we proposed heuristics for extracting stationary locations and dynamic clusters from individual traces. The heuristic extensively explains how to extract movement and stationary locations from the individual traces. We have uncovered a new characteristic feature in the human mobility patterns referred to as dynamic movement clusters. In our analysis, we employed statistical techniques for fitting the empirical data to the best fit distribution. Consequently, we determine the value(s) of the parameter(s) for the distribution of the inter-cluster travel distance, intra-cluster travel distance, intra-cluster angle of movement, pause time, inter-contact time, and contact duration as observed in the studied real traces.

Chapter 5

EPOM: Design and Implementation

5.1 Introduction

In this chapter, we introduce the Escape Path Mobility Model (EPOM). We start with a detailed description of the model, which explains the model's spatial domain, initial location, trip schedule, inter-cluster movement, intra-cluster movement, and pause time. Then, we present how a user switches between intra-cluster and inter-cluster movements to explore cluster locations and the whole domain. Finally, we explain the implementation details of the model on the ONE simulator, different submodels, including the obstacle submodel algorithm.

5.2 Model Specifications

5.2.1 Spatial Domain

The Spatial domain is a two dimensional bounded region R. Given any node i, the location of node i in the spatial domain R at time t denoted by $L_{i,t}$ is the position of i at time t in the spatial domain R, expressed in two-dimensional coordinates as $L_{(x,y)i,t}$. The spatial domain is divided into a number of communities denoted by c. c$\in\{1..N_c\}$, where N_c is the total number of communities in the domain. A community is a rectangular sub-region within the main simulation domain with a fixed size. The concept of a dynamic cluster in our model reflects the most frequently visited locations where a node spent a long time with zero velocity and performs many short walks within the community. It means a node can belong to more than one cluster within its community over time. Similarly, a node can visit other communities and generate clusters at some locations over time. In any of the two cases, we observed that a node takes short walks to explores its neighbors with more chances of visiting close neighbors than distant neighbors. This type of movement is mostly influenced by necessity and, therefore, its direction is not completely random in nature.

5.2.2 Initial location

We define the initial position of nodes by associating each node with a fixed location on the simulation domain using uniform node distribution. We refer to the initial location as *home*, denoted by H. At the start of the simulation, each node belongs to the community of its home location.

5.2.3 Trip Schedules

A node starts its trip from the initial location by generating trip schedules for the day. This trip schedule can be within its community (*i.e. intra-cluster movement mode*) or outside its community (i.e. *inter-cluster movement mode*) or both. The node uses a two-state Markov model in Figure 5.3 to switch between modes, which models node heterogeneity in movement area. It means a node's movement is not confined within a bounded community but can move out of the community.

Depending on the movement mode, a node chooses its next waypoint from the list

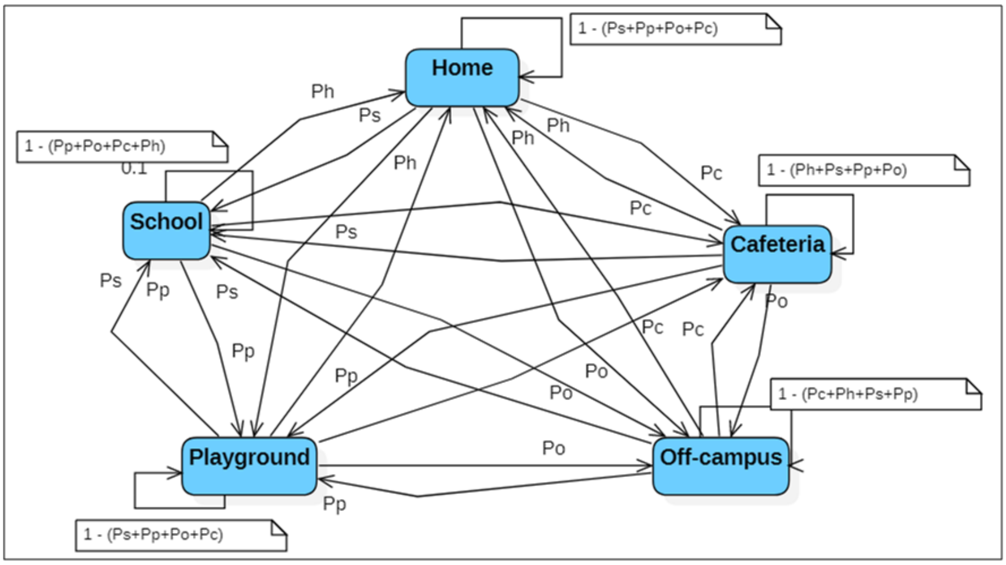


Figure 5.1: The figure shows the five-step Markov Model for location switching. The colored rectangles represent different locations a user may visits. The lines represent transition probabilities between locations, and the self-loop indicates the probability of staying in the same location.

of defined popular locations (i.e., home, school, cafeteria, playground, and off-campus) using a transition matrix defined by the five-step Markov model. Figure 5.1 helps to model the concept of *location preference*. Each step in the Markov model represents one *submodel*. Each submodel consists of different locations in the simulation domain; as shown in Figure 5.2. The transport submodel serves the function of switching between submodels, and the obstacle submodels ensure a collision-free transition in case of obstruction along the movement path.

After choosing the destination waypoint coordinates using the location switching Markov model in Figure 5.1, a node generates the shortest path using the Dijkstra algorithm and move to the destination with speed chosen uniformly at random in an interval [V_{min} - V_{max}]. When it reaches the destination, it pauses for time uniformly chosen at random from the pause time distribution of that location. Note that pause time is location dependant, i.e., pause time in the classroom is different from the pause time at the cafeteria. After the pause time, it decides to explore its neighborhood places

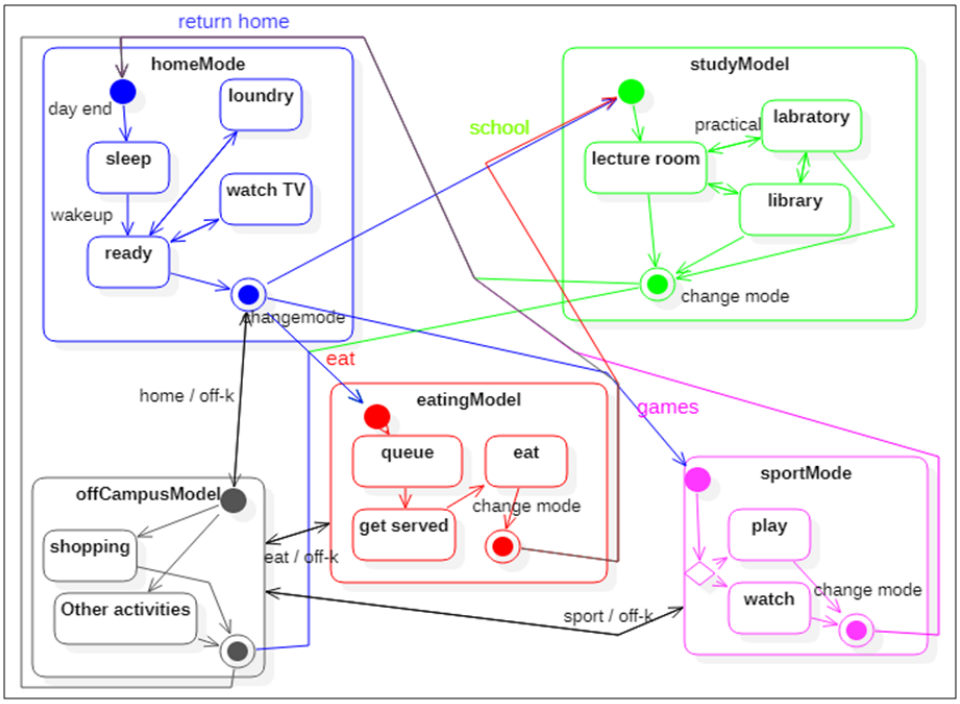


Figure 5.2: Submodels: The large rectangles represent the submodels (i.e. home, study, cafeteria, sport and off campus). The line connecting them is a transport submodel. The red shapes are obstacles which affect both mobility and signal propagation.

within the community or switch to inter-cluster movement. The probability of intra-cluster movement is higher than inter-cluster movement.

5.2.4 Travel Distance

The travel distance or jump size is the length of movement displacement, which depends on the movement mode. There are two different movement modes. The inter-cluster and intra-cluster movement modes are described below.

5.2.4.1 Intra-cluster Movement

We use an intra-cluster movement to capture the movement within the community. When a node stays in its community and undertakes many short walks to explores its neighboring locations, it creates a cluster within its community. It can create more than one cluster depending on the proximity of the locations visited and the time spent within the community. This phenomenon is called *neighborhood exploration* similar to the observed phenomenon in the empirical data. Generally, when a node is in intra-cluster mode, it either visits a point of interest within its community with higher probability or generates a travel distance chosen from a log-normal distribution bounded by the community size. Next, it walks with an angle biased toward popular locations within the interval $[0, 2\pi[$ (*intra-cluster location preferences*). Our choice of log-normal distribution was motivated by the fact that the empirical distribution of the *intra-cluster travel distance* is found to be log-normally distributed. The log-normal jump size means many short walks and fewer longer walks, and the bias symmetry direction of movement directs the movement of nodes toward the location of popular waypoints in the community.

5.2.4.2 Inter-Cluster Movement

When a node changes its mode to inter-cluster movement, it can move to any location in the domain with a higher probability of visiting popular locations. The travel distance is chosen from the truncated power-law distribution bounded by the main domain R size instead of the log-normal distribution in the *intra-cluster movement*. Note that whenever a node visits any location, it is associated with that community and guided by conditions defined in Section 5.2.4.1. That means a node can generate a dynamic cluster in the community or otherwise.

5.3 Model Implementation

The Escape Path Obstacle-based Movement model was implemented on the Opportunistic Network Environment (ONE) simulator Keränen et al. [2009]; Tkk/Comnet [2013] as a collection of different submodels. The ONE supported different movement models

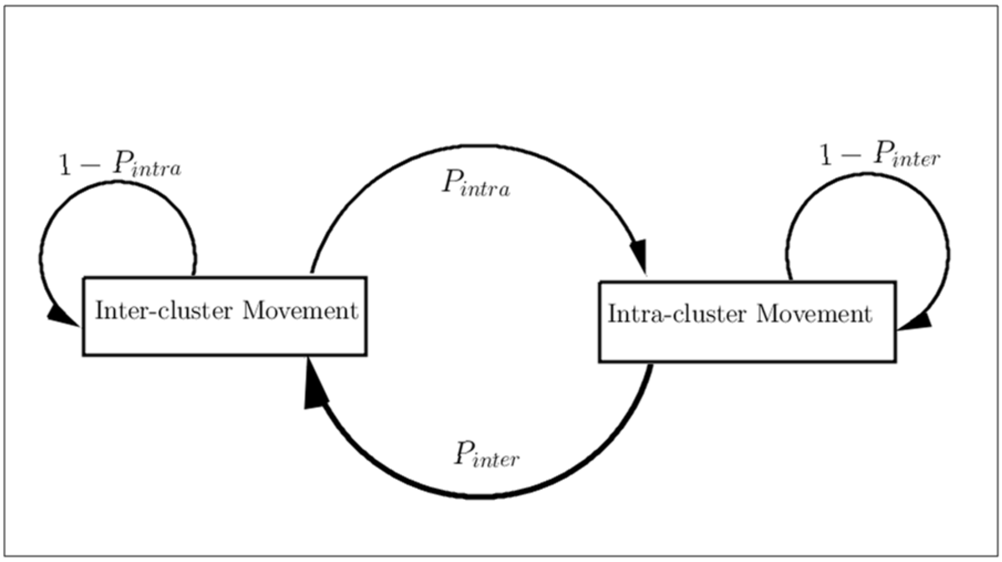


Figure 5.3: Two steps Markov Model for switching between the inter-cluster and intra-cluster movements

like the Random Waypoint Movement (RWP), Map-Based Movement (MBM), Shortest Path Map-Based Movement model (SPMBM), and Route-Based Movement model (RBM). The MBM is a special type of RW in which nodes move along the map paths defined in Well-Known Text (WKT) files. We use OpenJUMP Geographic Information System (GIS) program Project [2008] to define the location of obstacles, homes, classes, cafeterias, playgrounds, shops, and points of interest for off-campus activities in form of WKT files. We create a *main* movement model that inherited the *Extended movement model* of ONE and controlled the movement of nodes going to school, going to the cafeteria, going to sport, going shopping, or similar activities outside the campus and finally returns home to sleep. The main model orders and switches between submodels pass the control to the submodels responsible for different activities, facilitates the movement to the destination by giving information about the destination to the *transport* submodels, and decides on the probability to walk or use a bus based on the setting configuration.

5.3.1 Sub Models

In this section, we describe different submodels implemented in the proposed model.

5.3.1.1 Home Submodel

Home is the starting point of the simulation. Initially, a predefined location is assigned to each node in the home location file. These locations are used for *sleeping* or nodes' free time. Daily routine activities of a node start in the morning when it *wakes up* from the *sleeping state*. Each node is assigned a *wakeup time*, which determines when the node should wake up from the *sleeping*. The wakeup time obeys a normal distribution with the mean seven o'clock and configurable standard deviation. After waking up, a node would check its lecture schedules and decide whether to go for a lecture or do some *in-home* activities like cocking, watching morning headlines, laundry services, and visiting a friend nearby dormitory, etc. These short walks account for the possible evolution of the first dynamic cluster. Some nodes leave their home without doing any internal activities. Depending on the time of the day and a node lecture schedules, a node can switch to other *submodels* from *home*. For example, a node may switch to the *sport submodel* in the evening to play games, and it can switch to *eating submodel* for dinner or switch to *off campus submodel* for shopping or visiting a friend in another location. This flexibility of EPOM captures *social influence* and *heterogeneity* in time and space.

5.3.1.2 Study Submodel

We assigned specific locations on the map as a *lecture rooms*. If the node is in the lecture room, it walks into the lecture room and pauses for the lecture duration. The pause time distribution is location-dependent in our model. The pause time for the lecture is different from the pause time at the cafeteria. However, the pause time at non popular locations is derived from the truncated power-law distribution observed in the empirical data. We turn off the pause time completely during the lecture period for 80 percent of the nodes; only 20 percent can make some movement within the lecture room; this is to capture realistic behavior of students for changing desks or forming discussion groups. After the lecture, a node decides to walk to the laboratory for a practical or library to read. This *internal movement* is modeled as an intra-cluster walks within the vicinity of the study area with the libraries, laboratories, and other study-related locations as points of interest.

5.3.1.3 Eating Submodel

Some strategic locations on the map are defined as cafeterias. When it's time for lunch or dinner, a node may switch to *eating submodel* and move to the cafeteria to eat. While in the cafeteria, a node waits, makes some intra-cluster walks, and gets served, then eats and switches to another *submodel.* During the eating activities in a large cafeteria, we observed a large crowd of students within a confined location; hence, the need for collision avoidance to allow smooth flow of students. Our obstacle submodel ensures collision-free movement within the simulation domain.

5.3.1.4 Sport Submodel

We define some points on the map as playgrounds; the time for sport is also defined. A node in the *sport submodel* spends some time at the playground watching or doing some random intra-cluster movements around the vicinity of the playground.

5.3.1.5 Off-Campus Submodel

The off-campus submodel models all activities not included in the above four submodels. These activities include shopping, evening walk, or visiting friends. We define some Points of Interest (PoI) on the map edges as meeting points. We have two types of PoIs; location preferences PoI and Bus Normal PoI with uniform preferences. Mobile nodes visit such locations in a group to capture *group mobility* characteristics and *social* *influences* of human mobility and individually capture *independent mobility freedom.*

5.3.1.6 Transport Submodel

This submodel is used to move between different submodels when a node switched mode. We define two means of transport in our model: *walking and bus riding.* Most nodes *walk* while a *bus* is mostly used for an off-campus activities. The probability of moving with a bus is configured in the setting. The heterogeneity in the transport submodel has a great impact on the performance of routing protocols; high-speed nodes can deliver messages to longer-distance destinations quickly.

Bus service is accessible by the node at predefined *bus stops.* Initially, the node walks to the nearest bus station, wait for the bus; when the bus arrives, the node enters the bus and drop at the bus stop nearest to its destination, the node switches to a walking submodel to complete its journey to the final destination.

The nodes in our model move on the map; this is another aspect of realism. The maps contain the location information of homes, classes, cafeterias, playgrounds, shops, PoIs, and bus stops. The map data is essential for restricting the movement of the nodes to specific areas, which helps to increase node *localization.* It is used to distribute nodes in the simulation area uniformly.

5.3.1.7 Obstacle submodel

The obstacle submodel describes how the EPOM model handles collision avoidance between nodes and other obstructing objects along their movement path. In the case of a static obstacle with zero speed like a pedestrian standing on the road, at the middle of the corridors, or any other static object, we define the location of different obstacles on the map using OpenJUMP geographic information system program as in Figure 5.4.

The transport submodel moves the node from the current location (e.g., home) to the destination (e.g., class). Dijkstra shortest path algorithm is used to calculate the

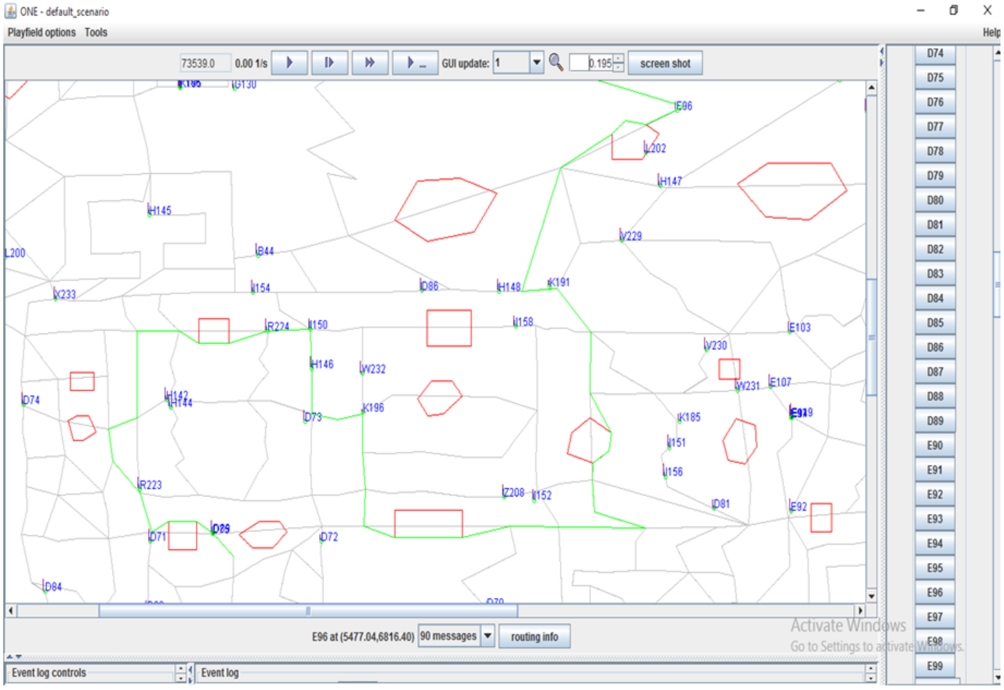


Figure 5.4: An example of visualization of a mobility scenario. The red irregular polygons mimic random obstacles. The blue numbered icons stands for nodes. The gray line stands for normal trajectories without obstacles and the green line represents a new trajectory created by node E96 using Escape path mobility model.

shortest path from the current location to the destination. We have two scenarios here: in the first scenario, there is no obstacle(s) on the path, while in the second scenario, an obstacle exists along the shortest path. In the first scenario, a node would follow the shortest path to its destination without obstruction, but in the second scenario, a node would explore the logic in Algorithm 1 to generate an escape path using the following transitions:

1. Move along the shortest path trajectories until an obstacle is reached, keeping a minimal distance to the obstacle
2. Generate escape path using Algorithm 1
3. Complete the movement to the next obstacle (in case of more than one obstacles) or destination

4. Repeat 2 and 3 until the final destination is reached.

Algorithm 1 Escape Path Movement for node *i*

```
   Initially $escapeVertex = \emptyset$, $neighbors = \emptyset$, $distToDest = \emptyset$.
1: get obstacle's vertices
2: $escapeVertex$ := nearest vertex
3: repeat
4:     move to the $escapeVertex$
5:     $neighbors$ := neighbor vertices
6:     $escapeVertex$ := nearest neighbor
7: until $distToDest(escapeVertex) \leq distToDest(\text{all neighbors})$
8: move to the destination
```

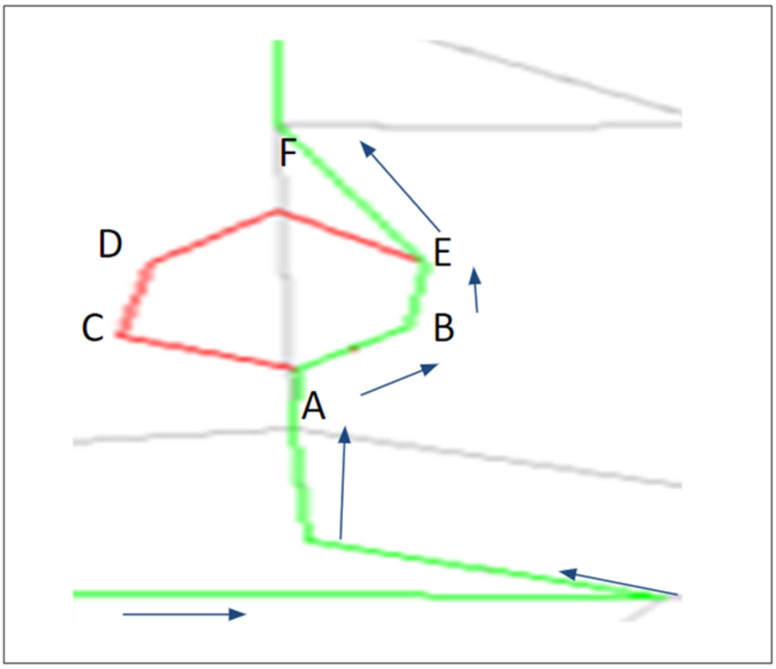


Figure 5.5: A snapshot of the Escape Path generated with Algorithm 1 from the ONE simulator

Algorithm 1 avoids collision with an obstacle by generating an escape path as shown in Figure 5.5. In Algorithm 1, line 1 gets the coordinates of an obstacle's vertices V =(A,C,D,F,E,B). Note that the shape of an obstacle determines the number of vertices. In line 2, a user finds the nearest vertex A. It moves to vertex A in line 4. It finds the neighbors of vertex A (i.e., B and C) in line 5 and set the next escape vertex to the nearest neighbor B in line 6. It then checks the condition in line 7; if the distance from its current location A to the destination is less than the distance from its neighbor C and B to the destination, it moves directly to the destination (line 8); otherwise, it returns to line 3. Considering a human movement behavior of walking beside the edges of an obstructing body until it passes the section of the obstacle that blocks it, the algorithm behaves similarly by creating a path beside (not on the edges) the edges of the obstacle. Some of the existing works have proposed a bezier curve Wang et al. [2017] or branching to the closest neighbor node Wu et al. [2011], which is not always realistic because a human path of escaping obstacle cannot always be curved, just like an isolated obstacle may not have a closer neighbor.

5.4 Conclusion

We present a detailed description of the proposed model. We described the model domain, initial location selection, trip schedules, inter-cluster movement, intra-cluster movement, pause time, and collision avoidance with obstacle submodel. These details are sufficient enough to describe any mobility model. Additionally, we discuss the implementation of the proposed model on ONE simulator as a collection of submodels, which handles several routines. In the next chapter, an evaluation of the proposed model would be presented.

Chapter 6

Model Validation and Results Discussion

6.1 Introduction

This chapter validates the EPOM model presented in the previous chapter. We discuss some important results obtained from simulation runs with the proposed model and other models for comparison. Finally, we present a performance evaluation of routing protocols using different routing performance metrics.

6.2 Validation

In this section, we show that our conceptual model (EPOM) is generic enough to be fine-tuned with a few parameters to show matching characteristics with the KAIST and NCSU GPS traces Rhee et al. [2009], in terms of the spatial features: *travel distance* and *direction of movement* as well as the temporal feature (i.e *pause time*). We also show that EPOM connectivity features matched that of imote real traces Scott et al. [2009] in terms *contact duration*, and *inter-contact time* distribution.

6.2.1 settings

Our simulation environment is a map of the Universite Paris-Saclay (i.e. Universite Paris Sud, IUT, and Centrale Supélec), edited using OpenJUMP geographic information system program with 1000 nodes moving on the area of roughly 5000X3000m^2. We created different WKT files for the map roads, homes, lecture rooms, cafeterias, sport, off-campus activities locations, POIs, and obstacles. See Figure 6.1 for the map.

Each node is assigned with a unique home location on the map as its initial location, a wake-up time drawn from a normal distribution. After the wake-up, a node starts to walk using the current mobility model, switches to different locations from the current location using the five (5) steps markov model depending on the time of the day. See Table 6.1 for the list of simulation parameters.

We simulate a Random waypoint model on the same size simulation area with 1000 nodes uniformly distributed. Each node randomly chooses a waypoint and move with a speed of 0.5 - 1.5m/s. When it reaches the destination, it pauses for 1 - 3600s; both the speed and pause time are uniformly distributed.

The simulation was run for the length of $T = 5 \cdot 10^5 s$, which is approximately five days. We assume all events are uniformly distributed over a longer period of time and consider the probability of an event of length x, p(x). We record only events that begin and end within the observed interval. We create the Complementary Cumulative Density Function (CCDF, P[X > x]) for the distribution of contact duration, inter-contact time, inter-cluster travel distance, intra-cluster travel distance, intra-cluster movement direction and pause time.

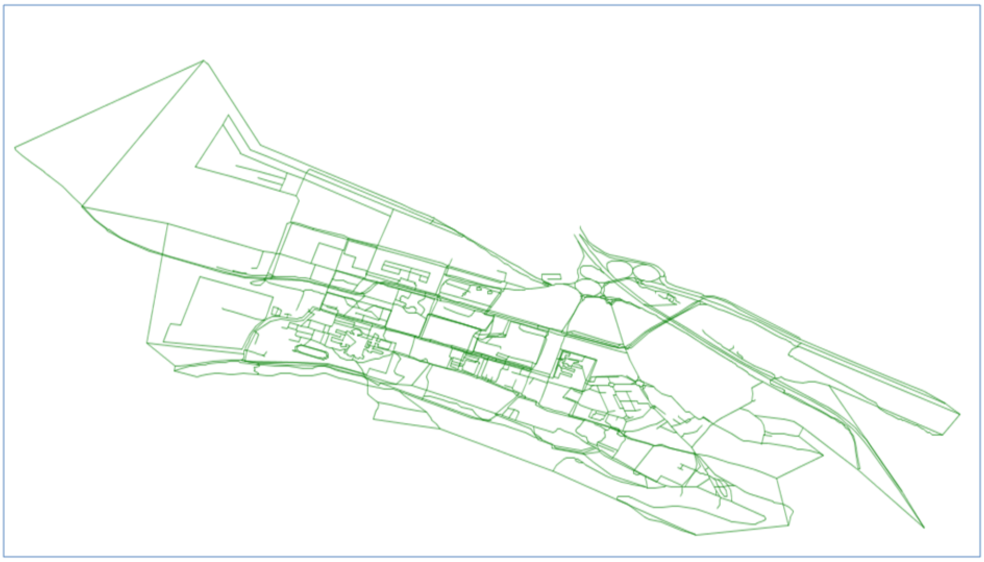


Figure 6.1: The Map of universite Paris Saclay (IUT, Centrale Superlec, and ENS) edited using OpenJump program. The Well-known text (wkt) geometry representation is used to develop the map.

Parameter	*Value*
Number of Nodes	1000
Simulation Length	500,000 sec
Transmit Range	10 m
Obstacle path Transmit Range	[5, 10] m
World Size	5000 x 3000 m^2
Walking speed	[0.5, 1.5] m/s
Bus Speed	[7, 10] m/s
Transmit Speed	250 kbps
Routing Protocol	Epidemic
Interface type	Simple Broadcast Interface
Buffer Size	50 mb
Message Size	[500 kb, 1 mb]
Message Interval	[25, 35] sec
Message TTL	1,430 sec

Table 6.1: Part I summary of the simulation parameters

6.2.2 Spatial Characteristics

We start with the *intra-cluster travel distance* feature being one of the most important aspects of our study. To account for the dynamic clusters, the main simulation domain is divided into communities of equal size denoted by c. c$\in\{1..N_c\}$, where N_c is the total number of communities in the domain. During the analysis of real traces, we find out that each individual walker is associated with an average of three dynamic clusters per day as shown in Figure 6.2 depending on the degree of the repetitiveness of the user's schedule, this type of temporal mobility feature can be exploited to predict the possible user location. Similarly, it can be exploited by the opportunistic routing protocol to schedule package forwarding.

After tuning our model, it generates matching clusters with the KAIST data. Figure 6.3 shows one-day dynamic clusters of node four generated from the EPOM synthetic traces.

Next, we show that the EPOM synthetic trace replicates the *neighborhood exploration* observed in the real traces. In our model, at each time instant, a node is either in *inter-cluster* or *intra-cluster* movement mode controlled by the two-step Markov model in Figure 5.3. When a node is in the intra-cluster movement mode, it explores the point of interest within its community and walks to the preferred POIs or generates a travel distance chosen from a log-normal distribution bounded by the community size. The direction of movement is chosen from a bias direction symmetry distribution in the range $[0, 2\pi[$, see Figure 4.6. The log-normal distribution of the intra-cluster travel distance means nodes visit closer locations more frequently than distant locations.

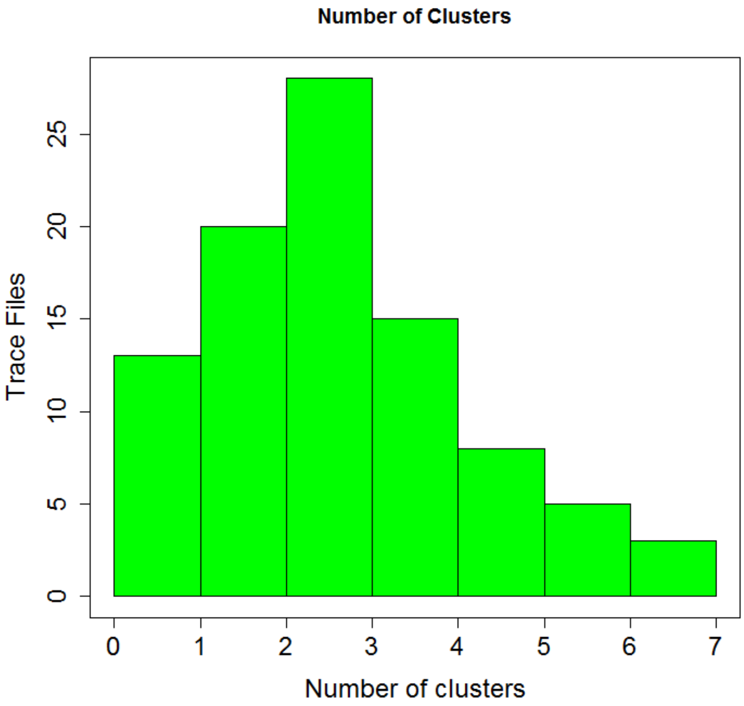


Figure 6.2: Number of dynamic clusters per trace file in KAIST traces. The average of three clusters is generated by each individual user

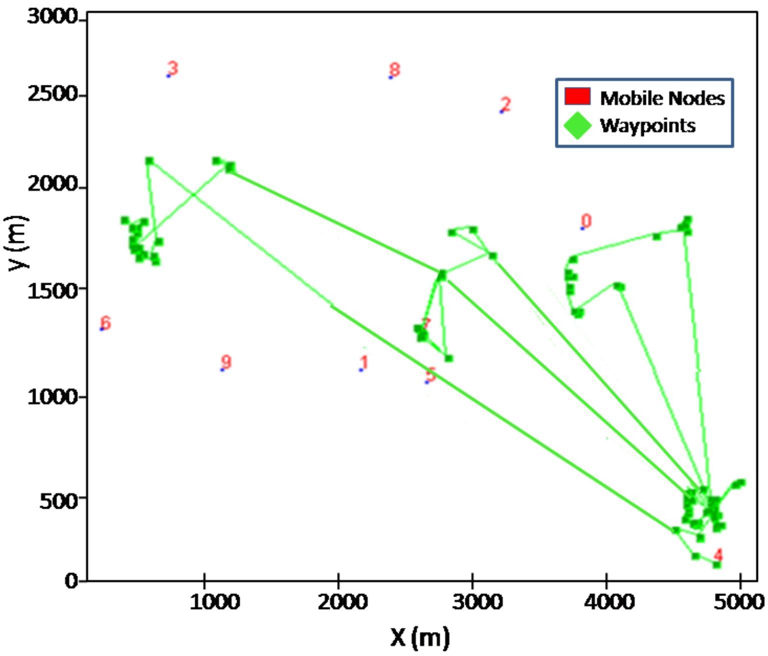


Figure 6.3: The figure shows the synthetic clusters generated by Node four (4) in the EPOM model. The red numbers represent mobile nodes, the green points indicate waypoints, and the green lines represent the node trajectories. We can see the concentrated waypoints as dynamic clusters.

Figure 6.4 shows the *intra-cluster travel distance* generated from the proposed model compared to that of the empirical data. The x-axis represents the travel distance in meters and the y-axis represents the cumulative probability. The figure shows that the curve for the two distributions is similar for a longer period of time but slightly differs at the tail, which is due to the community size in the simulation domain. Therefore, EPOM replicates intra-cluster travel distance as observed in the KAIST empirical data.

We show the travel distance for the main domain by analyzing an *inter-cluster travel distance* distribution generated by the EPOM synthetic trace and compare it with the empirical distribution observed from the empirical data. This is the approach adopted by most of the existing works in Altman et al. [2004]; González et al. [2008].

Figure 6.5 shows the inter-cluster travel distance distribution for the KAIST, EPOM, and RWP traces. The distribution of the EPOM and KAIST traces fits the truncated power-law distribution. It shows that users tend to undertake many short walks in a community and occasionally take long-distance walks. We also note that such short distance walks that evolve over time are the consequence of intra-cluster movements. In contrast, the curve for the conventional RWP model fits uniform distribution, which does not differentiate between short and long walks. This feature does not resemble the realistic nature of human mobility patterns.

Figure 6.6 shows the distribution of direction angle generated from the synthetic traces of the EPOM model. The distribution is similar to the distribution of NCSU trace in Figure 4.6. The main take-home message from the two distributions is that movement within the dynamic clusters is bias toward some POIs and popular locations within the community.

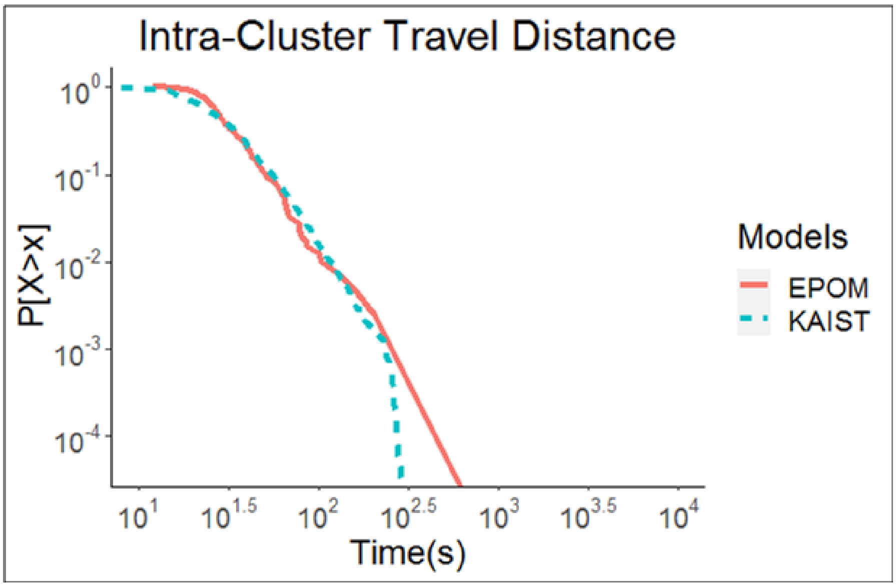


Figure 6.4: Intra-Cluster Travel Distance for the EPOM and KAIST traces. Both curves follow a log-normal distribution, meaning people visit some preferred nearby locations more than far distant locations.

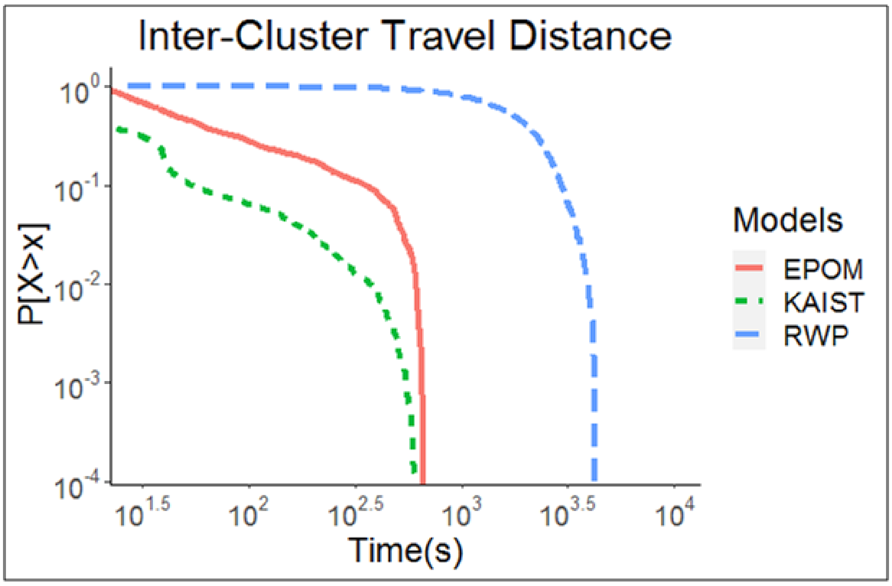


Figure 6.5: The figure shows the Inter-Cluster Travel Distance for the EPOM, KAIST, and RWP models. The curves for the EPOM model and KAIST traces exhibit power-law decay for a long period, supporting the realistic nature of the human mobility pattern for taking short walks more than a long journey. The RWP curve is uniformly distributed and does not differentiate between short walks and long journeys.

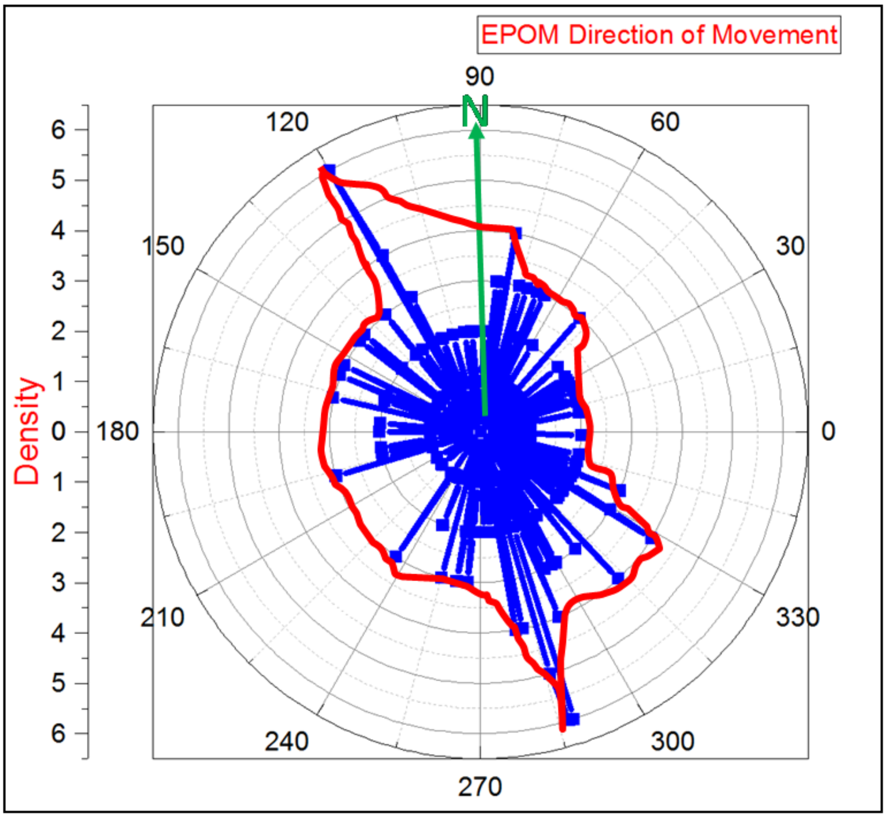


Figure 6.6: The bias symmetry distribution of Direction Angle for the EPOM clusters. The x-axis represents the Angular (Units are in degrees) and y-axis is the density of movement toward a given direction. The bin size is 1°. Each direction is weighted by the duration of its movement.

6.2.3 Temporal characteristics

Analysis of user's temporal locations at a certain period gives us insight into the possibility of predicting user's location, how long a user could stay at a given location, i.e., the *pause time.*, when the user is expected to return to a given location, i.e., the *return time* and why a user exhibit a skewed visiting behavior to some locations, i.e., *dynamic community walk*.

We study the *pause time* distribution of the KAIST campus traces in Rhee et al. [2009] and tune the EPOM model to generate *pause time* distribution similar to the empirical distribution observed. Figure 6.7 shows the pause time distribution of the

KAIST and EPOM traces. The distribution follows power-law decay with a heavy tail. It means students spent a long time at some locations such as lecture rooms but stayed for a shorter time at most locations like shopping malls and cafeterias. This distribution is consistent with the distribution of pause time observed in Dartmouth campus real traces in Kim et al. [2006].

The fact that users pause for a longer time at some preferred location also indicates that users predominantly take short walks within the vicinity of such locations, hence the possibility of creating a movement cluster. We observed that users are associated with an average of three dynamic clusters over the period of one working day, which evolve over time as shown in Figure 6.2. This fact is true for all users, with the exception of stationary users.

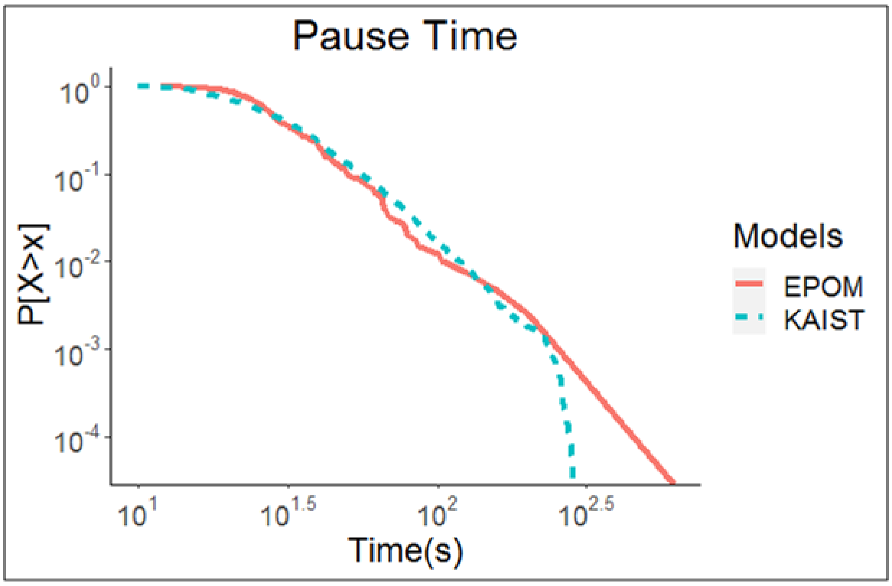


Figure 6.7: The Pause Time Distribution of the KAIST and EPOM traces. The figure indicates that humans mostly stay short in most places they visited and stay at few locations.

6.2.4 Connectivity features

This subsection investigates how closely the EPOM model reproduces the distribution of the empirical data for the connectivity metrics. We compare the distribution generated by the EPOM, iMote, and Random Waypoint traces on each plot.

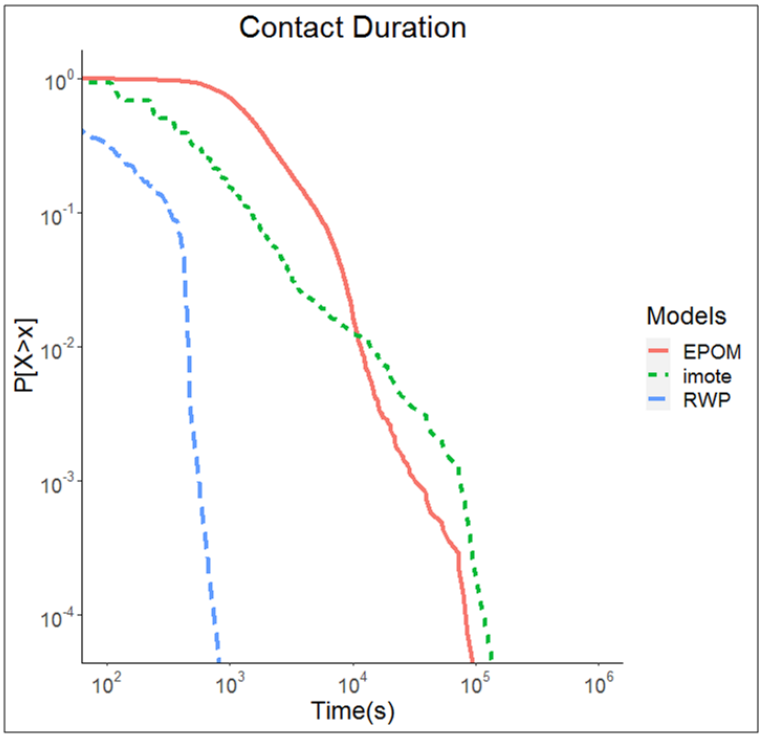


Figure 6.8: The figure shows the distribution of contact time for the EPOM, iMotes, and RWP model. The EPOM model follows power-law distribution for a long time, just like the iMotes traces, but RWP follows exponential distributions with very short contacts

Figure 6.8 shows the aggregate distribution of contact duration for the EPOM, iMote, and RWP. Each plot shows the complementary cumulative distribution function of a contact duration using a log-log scale. We see that EPOM distribution follows power-law decay for a long time, similar to the distribution of iMote traces. This result is consistent with most of the research on human mobility contact distribution Pirozmand et al. [2014]. The distribution of RWP consists of a much shorter time with exponential decay. The power-law feature of contact duration indicates that more nodes have contact opportunities for a shorter time while only a few nodes stay connected for a longer time. A DTN routing algorithm can be designed to exploit this feature in conjunction with the spatial-temporal features of human mobility to decide on the best way to route a message from the source to destination.

Figure 6.9 shows the inter-contact (ICTs) time distribution for the EPOM, iMote traces, and RWP. The figure shows that both curves of EPOM and iMote traces exhibit power-law decay with exponential cut-off, unlike RWP that entirely follows exponential distributions. The distribution of ICTs for the EPOM is also consistent with the feature of the realistic ICTs discovered in Chaintreau et al. [2007]. The power-law nature of ICTs plays an important role in DTNs as it fundamentally impacts the behavior of networking protocols Chaintreau et al. [2007]. Though shorter inter-contact time means more frequent connection, nodes with longer intercontact times are possibly assumed to have new data to share.

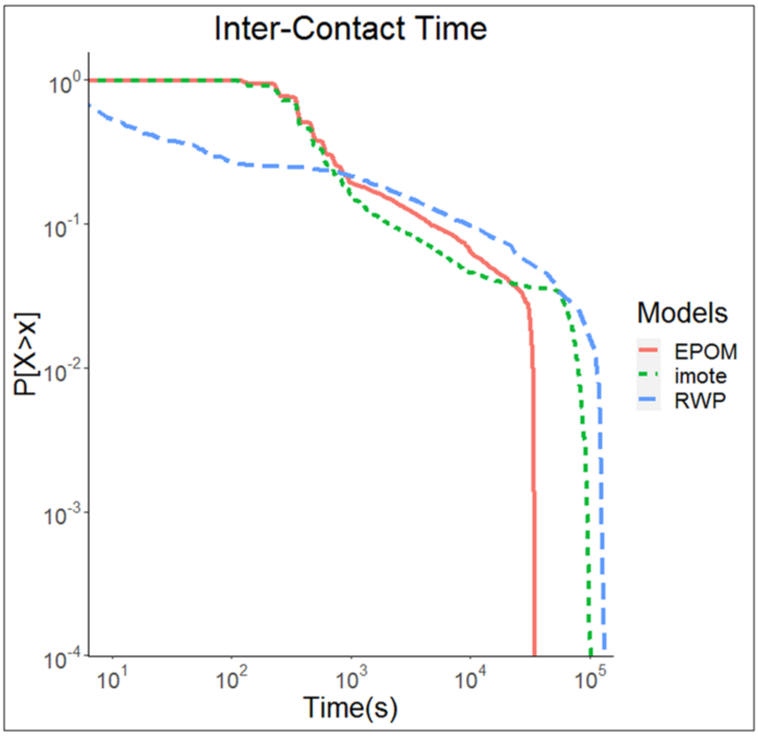


Figure 6.9: The figure shows the distribution inter-contact time distribution for the EPOM, iMotes, and RWP model. It shows that both the EPOM and iMote traces curves exhibit power-law decay with exponential cut-off, unlike RWP, which entirely follows an exponential distribution.

Figure 6.10 presents the contacts for each simulation hour, which shows the repetitiveness of hourly activities. We use 43200 seconds (12 hours) as working day length. The contact per hour of RWP is uniform throughout the simulation. Observe the repetitive behavior of the EPOM model, which captures student daily routine activities that took place at specific hours of the day.

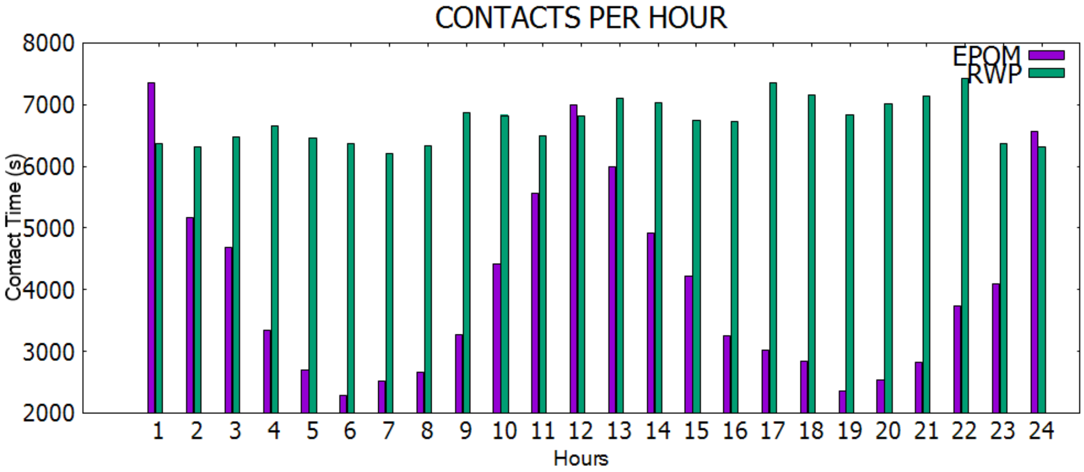


Figure 6.10: The distribution of contacts for each simulation hour, for two working days. The EPOM model has shown repetitiveness of hourly activities, unlike RWP, which shows uniform distribution of activities for each hour.

6.3 Conclusion

In this chapter, we validate our proposed model. We show that the model is generic enough to be fine-tuned with a few parameters to show matching characteristics with the real traces. After extensive simulation, the model generates matching dynamic movement clusters with the KAIST traces. It reproduces distributions of *intra-cluster travel distance* and *direction angle of movement* similar to the distributions uncovered in the analyzed traces. Similarly, it produces matching distributions of *inter-cluster travel distance, pause time, inter-contact time, contact duration, and contact per hour* with that of the analyzed traces.

Chapter 7

Conclusion and Future Work

7.1 Introduction

In this chapter, we summarized the contributions of the first part of the thesis on the characterization of human mobility and design of mobility model. We suggest further works that could extend our findings and further validate our claims.

7.2 Conclusion and Contributions

We conduct an in-depth study of human mobility patterns using realistic datasets for Bluetooth encounters, user associations to the mobile wireless networks, and Global Positioning System (GPS)tracklogs traces at the fine-grain level to understand the properties of human mobility better and to uncover hidden patterns. Consequently, we have discovered time-varying human mobility patterns associated with a dynamic evolution of movement clusters. We proposed a new synthetic mobility model called EPOM. We demonstrated that the EPOM model could recreate the statistical patterns observed in the studied real traces through series of simulations. We have also shown that the EPOM model can be used to evaluate routing protocols performance in wireless networks.

7.3 Future Work

We intend to extend the proposed model by designing an efficient predicting framework for human mobility. The new framework would exploit the existing and new uncovered features of human mobility to predict the user's next displacement, stay duration, and possible contact. The framework would be an excellent tool for transport agencies and network designers. Little have been done on the effects of obstacle on the signal propagation; We intend to extensively investigate the effects of obstacles (buildings, canopy, etc.) on signal propagation.

7.4 Publications

1. Bakura, S.A, Lambert, A., and Nowak, T. (2020). *An Obstacle-Aware Mobility Model for Campus Delay Tolerant Networks.* 2020 Zooming Innovation in Consumer Technologies Conference, 26-27 May, 2020. Novi Sad, Serbia.
2. A submitted Journal paper. Title: "EPOM: Escape Path Obstacle-based Mobility Model for Campus Delay Tolerant Network". Journal *Journal of Advance Transportation.* Date Submitted : 22nd June, 2021.

Part II

Clock Synchronization

Chapter 8

Introduction

In a smart campus scenario, users are passive beneficiaries of the services, active participants, and data explorers. In part I, we have characterized the mobility patterns of the users and exploit the mobility features to design a new synthetic mobility model for network protocols performance evaluation. In this part, we proposed a distributed clock synchronization algorithm for efficient information fusion in a campus wireless sensor network.

The use of sensors in a smart campus for environmental monitoring generates large hyperlocal data Prandi et al. [2019] that can better inform campus users (i.e., students, professors, administrators, and visitors) experience interacting with the campus space. The hyperlocal data like temperature level within a particular building recorded by thermal sensors, pictures snapped by surveillance cameras fixed at strategic locations to enhance students' safety, etc., is less reliable if only one sensor is used to record the data. More than one sensor is needed for the data gathering to increase data quality, reliability, and coverage, and then data fusion techniques are applied. Additionally, campus mobility plays a vital role in easing access to the campus environment. In this regard, a common notion of time among moving vehicles helps deliver services such as emergency interventions to avoid collisions, infotainment, navigation systems, and etc. Our proposed peer-to-peer distributed clock synchronization algorithm is aimed to provide a common notion of time amongst the distributed sensors for an efficient data fusion.

8.1 Motivation

Sensor networks are characterized by frequent changes in the network topology over time. Some of the critical factors that cause changes in the network topology include node mobility and changes in environmental conditions. However, despite all these challenges, a common notion of time among nodes is essential for data fusion algorithms. Some data fusion algorithms use timestamp package to estimate time measurement, which affects the accuracy of the measurement, as a shift between the time of measurement must be considered; hence the fusion imprecision is increased Han [2018]; Li et al. [2015]; Akyildiz et al. [2002]. Two techniques of synchronization are widely adopted ; (1) physical reference clock such as Global Synchronization for Satellite Navigation System Scopigno and Cozzetti [2009] and (2) virtual reference clock established using distributed clock synchronization algorithms Fugger et al. [2015]; Han [2018]. In the first approach, a Line of Sight (LOS) is required between the reference node (e.g., satellite) and the slave nodes, which may not always be available. Additionally, the failure of the reference node leads to the total loss of the network synchronization. On the other hand, a reference node is not required in the second approach; instead, all network nodes calculate and adjust their physical clocks based on exchanged information. In this regard, distributed approach is more robust in an environment with many obstructions, canopies, and indoor mobility. Moreover, most sensors are powered by a battery with limited power, which needs to be efficiently managed to prolong the network lifetime Ye et al. [2002]. Hence, the need for an optimal distributed synchronization algorithm that utilizes minimum computational resources without sacrificing the synchronization accuracy.

8.2 Background

Clock synchronization (e.g., Scopigno and Cozzetti [2009]; Sommer and Wattenhofer [2008]; Simeone and Spagnolini [2007]; Aissaoua et al. [2017]; Fugger et al. [2015]; Charron-Bost et al. [2015]; Han et al. [2018]) is an active research topic, which is categorized into reference-based and distributed algorithms. In reference-based algorithms Scopigno and Cozzetti [2009]; Sommer and Wattenhofer [2008]; Aissaoua et al. [2017], a reference node called *master* is used to broadcast synchronization packages to the *slave* nodes. Each slave, upon reception of the reference clock's value, updates its local clock accordingly. Reference-based algorithms are centralized, which means failure of the master node can lead to the loss of synchronization for all slave nodes. Considering the resource-scarce nature and topological changes of WSNs, such techniques are unreliable. Distributed algorithms use the concept of consensus between network nodes. Consensus allows each node to update its local clock value based on the time information received from its neighbors such that all nodes in the system will asymptotically converge to a common consensus value (i.e., agree on a common value).

Simeone and Spagnolini Simeone and Spagnolini [2007] presented a pulse-coupled synchronization algorithm that generates a correction term in each round. In their model, nodes have discrete-time clocks with phase-locked loop oscillators, constant and homogeneous frequencies. Függer et al. Fugger et al. [2015] improve upon Simeone and Spagnolini [2007] by presenting an algorithm for both unidirectional and bidirectional dynamic networks with clock frequencies not necessarily constant or homogeneous. The correction term $cor_i^1(k)$ was used in Fugger et al. [2015] for compensating skew. However, it is also beneficial to compensate for clock drift, which can vary due to changing ambient conditions. As a step to better take into account clocks drift, Han et al. Han et al. [2018] proposed a *timewheel* algorithm with an additional correction term $cor_i^2(k)$.

8.3 Objectives

The specific objectives of this part of the thesis, dealing with clock synchronization, are:

1. To implement a distributed clock synchronization algorithm that tolerates impacts of node mobility.
2. To design an optimal algorithm that utilizes minimum computational resources without sacrificing the synchronization accuracy.
3. To develop a new technique of reducing short-term fluctuations due to the unpredictable changes in the *clock drift*

8.4 Contributions

This part of the thesis proposed an algorithm that improves the work presented in Han et al. Han [2018]. We develop an exponential smoothing average algorithm for offset correction and drift compensation using present and previous time estimates. To ensure prediction accuracy in our algorithm, we develop an automatic and adaptive selection procedure for choosing the exponential decay term. Traditionally, the algorithm places exponential weights on previous estimates, with higher weights on more recent estimates. Still, due to the dynamic nature of WSNs, we study the impacts of different values of the smoothing factor on the performance of our algorithm. With this approach, short-term fluctuations characterized by unpredictable changes in clock drift are reduced as compared to the previous work Fugger et al. [2015]; Han [2018].

8.5 Outline

In Chapter 9, we review the existing distributed clock synchronization algorithms related to our study. We discuss the state-of-the-art methods and highlights their strength, shortcomings, and the need for improvement. Chapter 10 presents the clock synchronization problems, the system model and describes the proposed clock synchronization algorithm. We detail the three main functions of our algorithm: the sending function, receiving function, and timer function. Additionally, we discussed how the new technique performs the offset compensation, drift compensation, and adaptive weight factor selection. Finally, a mathematical analysis and proofs for the clock skew and convergence rate were presented. The simulation results and comparison of the proposed method with the existing works were presented in Chapter 11. Finally, we present a general conclusion, contributions, and the future perspective in Chapter 12

Chapter 9

Related Works

9.1 Introduction

This section focused on the non-centralized distributed clock synchronization algorithms to which our proposed algorithm belongs. Later in the chapter, an overview of more related algorithms is presented. Several works have been done on distributed clock synchronization algorithms, which can be classified into *package based* and *non-package based*. The package-based techniques Rome [2001]; Sommer and Wattenhofer [2008]; van Greunen and Rabaey [2003]; Ganeriwal et al. [2003]; Aissaoua et al. [2017]; Dolev et al. [1995]; Lamport and Melliar-Smith [1985] are based on the concept of exchanging data information (i.e., timestamp, global time, frequency, or some system parameters) between neighboring nodes, which is used to decide on a common value with which each node updates its clock.

Rome [2001] presents a timestamp synchronization algorithm which embedded *time-stamped* information in a package. The time offset is derived from the calculation of the round-trip delay between the transmitters and receivers.

A lightweight time synchronization algorithm was presented by van Greunen and Rabaey [2003] to reduce protocol overheads. The algorithm works by creating a spanning tree for the network with n nodes; then, it performs pair-wise synchronization along with n-1 nodes of the spanning tree. Ganeriwal et al. [2003] proposed Timing-Sync Protocol using a hierarchical approach. It operates in two phases; first, the *level discovery* in which all nodes at the high level (i.e., level 0) broadcast messages to the nodes at the lower level (i.e., level 1), the operation is repeated until all nodes in the network have been reached. Second, the *round-trip synchronization operation* where all nodes synchronize their clocks to the parent/root node in the tree.

Despite the efforts made on the package-based techniques, message delivery might not be reliable for larger networks due to their dynamic nature and resource scarcity Wu et al. [2011].

As a remedy to the package-based algorithms' problems, pulse-coupled synchronization algorithms were proposed Simeone and Spagnolini [2007]; Fugger et al. [2015]; Han [2018].

Simeone and Spagnolini [2007] presented a pulse-coupled synchronization algorithm that generates a correction term in each round. In their model, nodes have discrete-time clocks with phase-locked loop oscillators, constant and homogeneous frequencies.

Fugger et al. [2015] improves the work of Simeone and Spagnolini [2007] by proposing an algorithm for uni-directional and bi-directional dynamic networks with clock frequencies not necessarily constant or homogeneous.

The correction term $cor_i^1(k)$ was used in Fugger et al. [2015] for compensating both clock skew and clock drift. However, the sole use of $cor_i^1(k)$ is not efficient because clock drift can vary due to changing ambient conditions like temperature. After all, the characteristics of the quartz clock are influenced by temperature changes in the environment Ogrizovic et al. [2012].

As a step to solve this problem, Han [2018] proposed a *timewheel* algorithm with an additional correction term $cor_i^2(k)$ for compensating drift and thus increasing the precision of clock synchronization. Our algorithm improves the *timewheel* algorithm proposed in Han [2018], it proposed an adaptive exponential smoothing technique for compensating the clock offset and drift. Thereby making our new algorithm adaptive to unexpected fluctuations of clock drift due to node mobility, which causes variations in the network topology and environmental conditions or changes in ambient conditions.

9.2 State-of-the-Art

Considering the number of literature in clock synchronization algorithms, coupled with numerous assumptions specific to each class of algorithm, it's a good approach to concentrate on those more related to our proposed algorithm. This section presents details on WSNs clock synchronization algorithms that used the Average consensus technique to compensate for the clock time-varying variables *(i.e., clock offset and clock drift)*.

The goal of *consensus* algorithms is to ensure all agents in the network agreed on a common value. Therefore, each agent i maintains its own estimate $x_i(t)$ of the objective over time, which should *asymptotically* converge to a common limit c within the range of the input values. The *Average consensus* adds the constraint that the common limit c should be the average of the inputs values.

A consensus-based protocol, referred to as *Average TimeSync* (ATS), for synchronizing the clocks of a wireless sensor network was presented by Schenato and Fiorentin [2011]. The algorithm was realized by cascading two consensus algorithms such that the first consensus synchronizes clock speed (i.e., clock drift), and the second consensus synchronizes the clock offsets. These two variables in WSNs are mathematically modeled by Equation (9.1). So every node i in a WSN is associated with its own local clock whose first order dynamics is given by model in Equation (9.1).

$$\tau_i(t) = \alpha_i t + \beta_i, \tag{9.1}$$

Where τ_i is the local clock reading, α_i and βi are the frequency and offset of the local physical clock, respectively. Note that the absolute time t is not available to the nodes, so the values of the variables α_i and βi for node i cannot be directly computed but logically derived using relative information of the local clock of node i with respect to another node j. Moreover, α_i and βi may not be the same for all sensors due to manufacturing defects or ambient conditions. To solve the problem of the local clock, each sensor i maintains a virtual reference clock (software clock) given by Equation (9.2).

$$\bar{\tau}_i(t) = \bar{\alpha}_i t + \bar{\beta}_i, \tag{9.2}$$

and each sensor keeps an estimate of the virtual time using a linear function of its own local clock as

$$\bar{\tau}_i(t) = \bar{\alpha}_i(t)\tau_i(t) + \bar{\beta}_i, \tag{9.3}$$

The ATS main goal is to find $(\bar{\alpha}_i, \bar{\beta}_i)$ for each sensor such that the difference between the estimate $\bar{\tau}_i$ and the *virtual reference clock* $\bar{\tau}$ converged to a common consensus value. This is achieved using the *offset compensation* to evaluate the best value of $\bar{\beta}_i$ and the *drift compensation* to evaluate the best value of $\bar{\alpha}_i$ respectively.

Relative Drift Estimation in ATS: Each sensor i estimates its relative drift $\alpha_{ij} = \frac{\alpha_j}{\alpha_i}$ from the package it receives from its neighbors $j \in N_i$, where N_i is the number of its neighbors. The package received from j at time update k is a tuple of $[ID_j, \bar{\alpha_j}(t_k), \bar{\beta_j}(t_k), \tau_i(t_k)\ ]$, where ID_j is the unique identifier of the sensor j. The communication between sensor i and its neighbor j is assumed to be instantaneous otherwise the protocol most be modified to account for the transmission delay, which is one of the limitation of ATS.

Drift Compensation: As soon as sensor i receives the tuple package from sensor j, it estimates the value of its $\bar{\alpha_j}(t_k)$ as follows:

$$\bar{\alpha_i}(k) = \rho_\alpha \bar{\alpha_i}(k-1) + (1-\rho_\alpha)\bar{\alpha_{ij}}(k)\bar{\alpha_j}(k-1) \tag{9.4}$$

The value of $\bar{\alpha_{ij}}$ is obtained using Equation (9.5)

$$\bar{\alpha_{ij}}(k) = \rho_\alpha \bar{\alpha_{ij}}(k-1) + (1-\rho_\alpha)\frac{\tau_j^{new} - \tau_j^{old}}{\tau_{ij}^{new} - \tau_{ij}^{old}} \tag{9.5}$$

$\rho_\alpha \in (0, 1)$ is a tuning parameter use to tune the trade-off between a fast rate of convergence (ρ_α close to zero) and a high noise immunity (ρ_α close to unity). The values τ_j^{new} and τ_j^{old} are the present and previous value of the node j local time recorded by node i at the time of package reception.

Offset Compensation: It is assumed that after the drift compensation algorithm is applied, all local virtual clock estimators will eventually have the same drift, i.e., they run at the same speed. what remains is to compensate for the time difference, which is achieved using Equation (9.6)

$$\bar{\beta}(k) = \bar{\beta}(k-1) + (1-\rho_\beta)(\bar{\tau}_j(k) - \bar{\tau}_i(k)), \tag{9.6}$$

Where $\bar{\tau}_j(k-1)$ and $\bar{\tau}_i(k-1)$ are computed at the same time instant. Some of the main limitations of ATS are its inability to handle transmission delay and its asymmetric nature. There is a need for a protocol that can implement symmetric communication between nodes or handle transmission delay.

Garone et al. [2015] proposed a Clock synchronization protocol for wireless sensor networks with bounded communication delays (RoATS), which improves the ATS protocol by integrating two concurrent algorithms for a *drift compensation* and *offset compensation*. The main contribution is the implementation of symmetric communication between neighbor nodes to achieve robustness against communication delays.

Drift Compensation: At time update k, node i and node j execute drift compensation by exchanging the first package, then each node computes its drift estimate and resend the calculated estimate as the second package, upon reception of the second package, each node performs symmetric *drift compensation* and compensate the effect of drift change on the offset using Equation (9.7)

$$\bar{\alpha_i}(k) = \alpha_i(k-1) + [1-\rho_\alpha]\mu(k)\Gamma_\alpha(k) \tag{9.7}$$

$$\bar{\alpha_j}(k) = \alpha_j(k-1) - [1-\rho_\alpha]\mu(k)\Gamma_\alpha(k)$$

Where $\rho_\alpha \in \Big(1 - \frac{2\alpha min}{\alpha max + \alpha min}, 1\Big)$, $\mu(k)$ is the update direction of $\bar{\alpha}(k)$ and $\Gamma_\alpha(k)$ is the magnitude of the correction. The RoATS, converges the virtual clock $\bar{\tau}(k)$ even in the presence of delay but the additional communication step makes it a resource-consuming protocol, which can affect the scarce resources of the sensor network.

Offset Compensation: In RoATS sensors compensate for the time difference considering delay but in one direction (asymmetric) as opposed to the drift compensation that uses symmetric estimation. In this regard, when node j sent a single package to node i at update time k, node i stores its virtual clock $\bar{\tau_i}(k)$ as well as virtual clock of node j extracted from the received package as $\bar{\tau_j}(k_h - \delta_h)$, where δ_k is the transmission delay bounded from above by $\delta_h \leq \delta_{max}$. Finally node i performs offset compensation as follows:

$$\bar{\alpha_i}(k) = \bar{\alpha}(k) - [1-\rho_\alpha][\bar{\tau_j}(k_h - \delta_h) - \bar{\alpha_i}(k)] \tag{9.8}$$

with the assumption that the interval between the transmission of two consecutive offset compensation package is upper bounded by

$$k_{h+1} - k_h \leq \Delta t^h_{max} \tag{9.9}$$

The above package-based algorithms proposed an interesting paradigm but extensive communication and packages exchange make them more complex to be adapted in a sensor network with sparse resources like energy and memory. These motivated some work in pulse-couple clock synchronization, allowing sensors to exchange zero-bit pulse and synchronize their clock with local information from its neighbor's pulse.

In Fugger et al. [2015] a pulse-coupled clock synchronization algorithm is proposed. They solve the open question posed by Simeone and Spagnolini [2007], which requires solving a clock synchronization in dynamic networks in the presence of non-zero clock drift. This solution reduces the assumption that the network stays connected over time. They proposed a correcting term with which all nodes adjust their clock and predict the next pulse time. The next time pulse is given as:

$$t_i(k+1) = t_i(k) + T_i(k+1) + corr^1(k+1) \tag{9.10}$$

where $(1-\rho) \leq T_i(k+1) \leq (1+\rho)$ and the correcting term $corr_i^1(k+1)$ is a weighted average of the time differences of received round k pulses given by

$$corr_i^1(k+1) = [\varepsilon_{k+1}] \sum_{j \in In_i(k+1)} [W_{k+1}]_{i,j}(t_j(k) - t_i(k)) \tag{9.11}$$

Where the Weights $[W_{k+1}]_{i,j} \in$ [0,1] for $j \in In_i(k+1)$ and Weight $\varepsilon_{k+1} \in [0,1)$. In Fugger et al. [2015], a round number k is assumed to be known by all sensors even though the round estimated by different sensors can be slower or faster depending on the sensor clock; this may not always be realistic. In addition to that, their is need for drift compensation to compliment the correcting term $corr_i^1(k+1)$.

9.2.1 Time Wheel Algorithm

Han [2018] improves the clock synchronization algorithm presented by Fugger et al. [2015] and proposed an algorithm named a *time wheel algorithm*. In this algorithm, the round numbers are assumed not to be known by sensors; instead, each sensor$_i$ broadcast a value $P_i \in \mathbb{Z}$, $0 \leq P \leq p_{max} - 1$. The value p_i serves a similar role as k_i, which increase every round, except when p_i reaches p_{max}, it returns to zero for the next round. Han [2018] also proposed a drift compensation technique to reduce clock drift due to the mobility, ambient conditions, or manufacturing defects. The *time wheel* algorithm operates in rounds. Sensor$_i$ broadcast a pulse p_i to its neighbors at the beginning of round k_i. Upon reception of the pulses from its neighbors j, sensor$_i$ records the local time of the pulses, converges the timing and switched to the next round. A sensor i uses a circular buffer called time wheel wi to record the pulse pj from neighboring sensor j. The structure of *time wheel* is depicted on Figure 9.1. Each sensor executes the *time wheel* algorithm

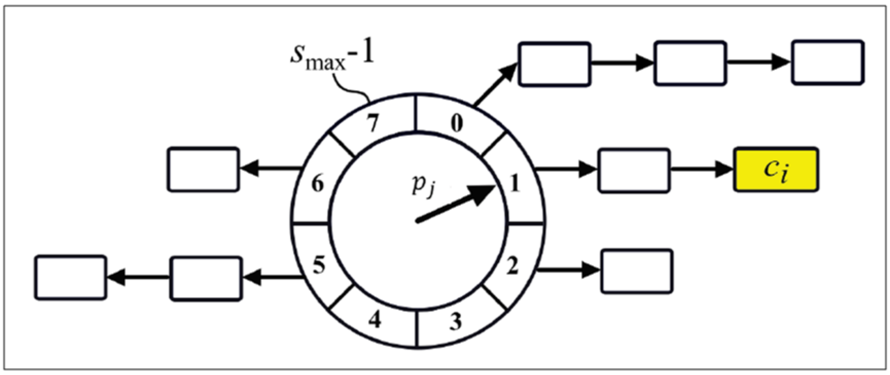


Figure 9.1: When sensor i receives a pulse with a tag p$_j$ from sensor j, it appends the physical local clock time c_i to the list of p_j-th slot of w_i

divided into three parts: initial function, receive function, and state transition function. Each part is triggered by different events. The most important part of the consideration in our study is the *Timer set function* of the algorithm. The timer set function is used to estimate the counter value for the next pulse $\tau(k+1)$ as follows:

$$\tau_i(k+1) = \tau_i(k) + R + corr_i^1 + corr_i^2, \tag{9.12}$$

Where R is a constant representing the time period between the next and present rounds. Though all sensors use the same value of R but the next round pulse $\tau(k+1)$ may not be the same due to the frequency variations. So the correcting terms $corr_i^1$ and $corr_i^2$ are the adjustment of the clock offset, and the clock drifts to reduce the clock skew. Note that the correcting term $corr_i^2$ is the improvement proposed by Han [2018] on the work of Fugger et al. [2015].

9.2.1.1 Offset Compensation

To compensate for the clock offset, sensor$_i$ estimates its correcting term as a time difference between the average pulse of the neighboring sensors and the pulse of sensor$_i$ itself in round k. This can be represented as follows:

$$Corr_i^1(k) = \frac{1}{|In_i(k+1)|} \sum_{j \in In_i(k+1)} (\tau_j(k) - \tau_i(k)) \tag{9.13}$$

Where $|In_i(k+1)|$ represents the number of incoming pulses at the $k+1-th$ round and $\sum_{j \in In_i(k+1)}$ (τ_j(k) - τ_i(k) is evaluated by averaging the values in the current slot of *time wheel wi*. We observe that using simple averaging to estimate correcting offset does not provide stable estimates due to the unexpected changes in clock drift due to the mobility and ambient conditions. Consequently, we propose a learning technique that utilizes previous rounds history to estimate a more accurate value of the correcting term. Details of our proposed algorithm is presented in the next Chapter.

9.2.1.2 Drift compensation

After an offset compensation, it is assumed that all sensors converge their time to a common value but possibly with different clock cycles. To make sure all sensors has common clock cycle, an estimate of the time period R (in terms of the number of clock ticks) (R + $corr_i^2$(k)) in the current round is derived as follows:

$$corr_i^2(k) = \Big(\tau_i(k) + corr_i^1(k)\Big)\frac{1}{R} - R \tag{9.14}$$

The correcting term in Equation (9.14) must be delayed until after some rounds l otherwise, the algorithm would not converge. Additionally, after a long run with the drift correction, the algorithm is affected by inertia due to the averaging nature of the correcting term $corr_i^2(k)$. To get rid of these limitations, we proposed a new drift compensation technique, which utilizes the present and previous time to estimate the value of a correcting term for drift compensation, which does not necessarily need to be delayed for some number of rounds. Hence reduces the clock drift, offset, and improve clock precision. The technique also corrects the impacts of inertia associated with the correcting term in Equation (9.14)

9.3 Conclusion

We present an overview of the existing distributed clock synchronization algorithms. We also review the existing state-of-the-art techniques widely adopted in the existing works. For each method, we discuss its strength, shortcomings, and where it needs improvements. We note that clock synchronization is required to implement energy-efficient and fault-tolerant protocols in a dynamic network. Therefore, synchronization algorithms that reduce the clock skew and offset in a dynamic network in the presence of scarce resources are of tremendous need. The next Chapter introduces our proposed pulse-coupled synchronization algorithm.

Chapter 10

Proposed Algorithm

This Chapter presents our solution for the clock synchronization problem. It describes the system model for the proposed algorithm. A detailed description of the synchronization functions, offset compensation, drift compensation, adaptive decay factor selection, and mathematical analysis of the new algorithm is presented.

10.1 Problem Description

The goal of clock synchronization is to make the virtual clock signal start at different sensors at the same time; the virtual clock of the sensors should be synchronized. The instantaneous difference between the readings of any two clocks (ticks) is called their skew. If the skew is null (and stays null), then the synchronization is achieved, and all the network nodes share the same common time.

The local physical clocks of the sensors may not run at the same frequency due to the ambient conditions, mobility, and manufacturing defects. The most common phenomenons that cause clocks to run at different frequencies and eventually diverge from the reference clock or their neighbors is the clock *drift* (see Fig. 10.1), the time difference between the clock period of node i and the clock period of node j at round k,

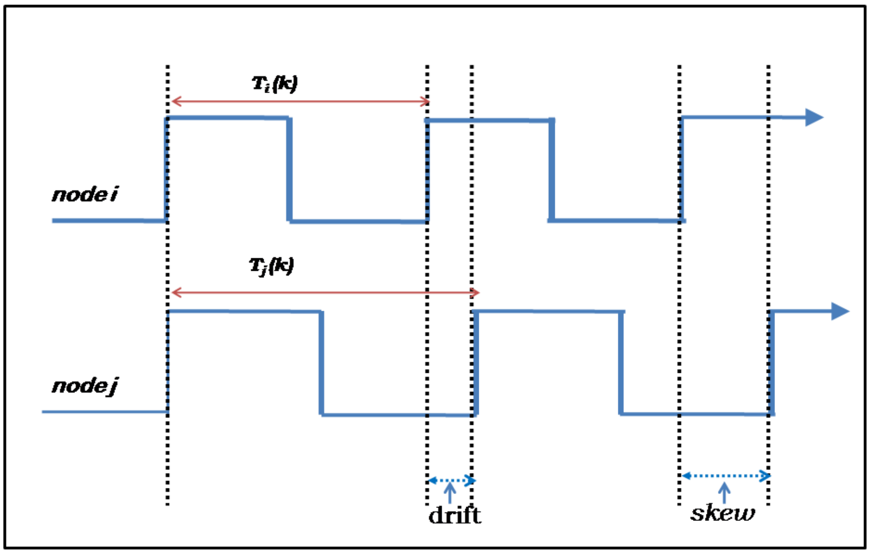


Figure 10.1: Virtual clocks of node i and j. Clock skew accumulates and grows after several rounds if no correction is applied, eventually the clocks of the nodes would diverge.

and is given by $drift(k) = |T_i(k) - T_j(k)|$, where $T_x(k)$ is the period of node x at update round k. The accumulated drift (time difference between clock periods or difference in clock frequency) modifies the skew. Figure 10.1 shows the evolution of virtual clocks of nodes i and j. Notice the growth of the clock skew on each round; it makes the clocks of the nodes diverge if synchronization is not applied. We need to minimize the clock *drift* and the clock *skew* as much as possible to enable clocks of the nodes to runs on the same frequency.

10.2 System Model

We consider a finite set of nodes $\mathcal{N} = \{1, 2, \ldots, N\}$ communicating by exchanging pulses. Each node is equipped with an imperfect local clock evolving in the time-base $\tau = [0, \infty)$. An environment is modeled by a set of scenarios (communication graphs) $G_t = (\mathcal{N}, E_t)$, $t \in \tau$. Where N is a set of nodes and E_t is a set of directed edges. The evolution of different communication graphs at different time instant is considered as mobility. Node i

receives messages from node j at time t only if a relation $(i, j) \in E_t$ exist between the node i and node j. We consider execution of an algorithm on a scenario G_t as sequence of events $(e_k)_{k \geq 0}$ triggered at nodes $(i_k)_{k \geq 0}$ in the current states $(s_k)_{k \geq 0}$ at times $(t_k)_{k \geq 0}$. At the initial state, each node triggers its first event at time $t = 0$. The state transition function maps a state and an event to a new state. Then, all nodes move to the next state from its current state $(s \mapsto s')$ using the state transition function. A node uses the *sending function* to send a pulse to its neighbors; likewise, it uses the *receiving function* to receives pulses from the neighboring nodes. The received pulses changes the node's internal state.

Assumptions

1. At any time instant t, a pulse sent by node i is received by node j only if $(i, j) \in E_t$.

2. Time between sending and receiving a pulse is assumed to be zero. The assumption is reasonable if the message contains only a zero bits.

3. All nodes are equipped with a local physical clock $\mu_i(c)$. The variable c is a counter that is initiated to 0 increases over time whenever the local physical clock ticks while μ_i is a function that subsumes manufacturing defects, hardware differences, and different temperatures for quartz oscillators. We assumed the local physical clock tick to be 1. The sensor i read real-time t_i as follows:

   $$t_i(k) = \sum_{c=1}^{\tau_i(k)} \mu_i(c) \tag{10.1}$$

   Assuming that

   $$(1 - \varrho)\mu \leq \mu_i(c) \leq (1 + \varrho)\mu \tag{10.2}$$

   where ϱ represents the bound of the clock period and $\mu_i(c)$ is the time period of the clock cycle.

4. *Count-down timer* Δc_i: each node is equipped with a count-down timer Δc_i, decreasing whenever the counter c_i increases. The value of Δc_i is set using the *timer set function*; whenever Δc_i reaches 0, an event $e_{tim} \in \varepsilon$ is triggered. Then, Δc_i stays at zero until it is set up again.

10.3 Synchronization Algorithm With an Exponential Weight (SAWEW)

This section introduces our proposed algorithm and how it performs offset compensation, drift compensation, and dynamic adaptive weight selection. Our algorithm adopted the *time wheel* round structure presented in Han [2018]. Firstly, nodes start in round $k = 0$ and broadcast a pulse in round $k-1$, wait for the time duration T_i(k) (R) before moving to the next round. All nodes use the same value for R, but the waiting time and next round start time for each node may not be the same due to the manufacturing defects and clock drifts. Therefore, pulses of the nodes are not perfectly synchronized with each other and hence require correction.

Each sensor i runs Algorithm 2, which has three main functions triggered by different events. See Section 10.3.1 for more detail.

10.3.1 Algorithm description

Algorithm 2 consists of three main parts: *Initial function, receive function, and state transition function.* Different events trigger each function; these events are the e_{beg} triggered at the beginning of the local algorithm, e_{rec} triggered when a sensor receives a message, and e_{tim} is triggered when the timer in a sensor reaches a certain value.

Algorithm 2 State transitions for node i

```
    Initial function
 1: if $e_{beg}$ then
 2:     $w_i$ = TimeWheel($S_{max}$)
 3:     $p_i = 0$
 4:     $c_i = 0$
 5:     Ptb = PreTimeBuffer($G_t$)
 6:     sending_function($p_i$)
    Receive function
 7: if $e_{rec}$ then
 8:     Obtain $p_j$ from $rec$
 9:     Append $c_i$ to the list in $p_j$
10:     Append $c_i$ to the $i - th$ slot of $Ptb$
11:     $\tau(k+1) = timer\_set\_function(w_i, \tau(k))$
12:     $\Delta c_i = \tau(k+1) - c_i$
    State transition function
13: if $e_{tim}$ then
14:     Clear the $pi - th$ slot
15:     $P_i = (P_i + 1) \ mod \ (P_{max})$
16:     sending_function($p_i$)
```

10.3.1.1 Initial function

At the beginning (lines 1–5), when an event is an e_{beg} (line 1), node i initializes an empty timewheel w_i (line 2) with a maximum slot S_{max} then reset round number p_i to zero (line 3) and initiate local physical clock counter at 0 (line 4). In (line 5) it initializes an empty *previous time buffer* and broadcasts a pulse P_i to all its neighbors in (line 6).

10.3.1.2 Receive function

At (line 7) When an event is a received event (e_{rec}), the node i receives a pulse from a node j (line 8), it appends the current value of c_i to the j-th slot of the time wheel (line 9) and also appends c_i to the i-th slot of *previous time buffer* (ptb) (line 10) then predicts the time period for its next pulse (line 11) and set its timer to $\tau_i(k+1)$ - c_i (line 12). The detail of the timer set function is in section 10.3.2. At this stage, the iterative process starts. Node i waits for the time period of its timer; whenever its timer reaches zero, it executes its state transition function and starts the next round.

10.3.1.3 State transition function

The Node i empties the current slot p_i since its values are used and no more needed in the next round (line 14). It updates the value of p_i (line 15) and finally broadcast pulse P_i in (line 16).

10.3.2 Timer Set Function

In Algorithm 2, the timer set function estimates the value of count down timer Δc_i for the next round $\tau_i(k+1)$ as follows:

$$\tau_i(k+1) = \tau_i(k) + R + \mathrm{C''_{es_i}(k)} + \mathrm{C_{dr_i}(k)}, \tag{10.3}$$

where R is the constant representing the time between the next and present rounds, $\mathrm{C''_{es_i}(k)}$ and $\mathrm{C_{dr_i}(k)}$ are the adjustment of time to compensate for the clock offset and drift.

10.3.3 Offset Compensation

The purpose of this stage is to make all nodes converge their clock time to a common consensus value. In each round k, node i broadcasts a pulse to its neighbors j and receives its neighbors' pulses at the same round; using the received pulses node i calculates and compensates its clock offset using Equation (10.4).

$$\mathrm{C'_{es_i}(k)} = \lambda\ \mathrm{C_{avg_i}(k)} + (1-\lambda)\mathrm{C'_{es_i}(k-1)}, \tag{10.4}$$

where $C'_{avg_i}(k)$ is the average time difference between pulses of the neighboring nodes j and the pulse of the node i in round k, $C'_{es_i}(k-1)$ is the exponentially weighted estimate at the previous round $k-1$ while λ represents the adaptive weight factor which controls the closeness of the interpolated value $C'_{es_i}(k)$ to the most recent estimate. The values of λ lies between 0 and 1: 0< λ <1. Observe that, Equation (10.4) is a Single Exponential Smoothing, which is applicable for time series with low fluctuations. In a highly dynamic environment, clock drift due to the ambient condition can change frequently. To address this phenomenon, we derive the Double Exponential Smoothing (DES) that is used for our analysis as Equation (10.5). The DES is represented in the Equation (10.5) as $C''_{es}(k)$.

$$C''_{es}(k) = \lambda C'_{es}(k) + (1 - \lambda)C''_{es}(k - 1). \tag{10.5}$$

We can see that in Equation (10.5) the $C''_{es}(k)$ is an exponential weighted estimated of the single exponential smoothing $C'_{es}(k)$. It ensures accurate prediction even when the time series shows trends, as in the case when a clock frequently drifts due to the change in ambient condition. Similarly, $C_{avg_i}(k)$ in Equation (10.4) is computed as

$$C_{avg_i}(k) = \frac{1}{|In_i(k+1)|} \sum_{j \in In_i(k+1)} (\tau_j(k) - \tau_i(k)) \tag{10.6}$$

Where $|In_i(k+1)|$ represents the number of incoming pulses at the $k+1-th$ round and $\sum_{j \in In_i(k+1)}$ (τ_j(k) - τ_i(k) is evaluated by averaging the values in the current slot of *time wheel wi*. Equation (10.4) is recursively written as Equation (10.7)

$$C'_{es_i}(k) = \lambda \sum_{n=0}^{k-1} \big((1-\lambda)^n C_{avg_i}(k-n)\big) + (1-\lambda)^k C'_{es_i}(k-k) \tag{10.7}$$

Equation (10.7) shows that $C'_{es_i}(k)$ is an exponentially weighted average of all previous corrections. Refer to Appendix D for the derivation of Equation (10.7).

Note that after the offset compensation, all network clocks have converged to a common clock, possibly with different clock cycles due to the frequency variation (*drift errors*) as in Equation (10.8).

$$t_i(k) + C'_{es_i}(k) \approx t_j(k) + C'_{es_j}(k). \tag{10.8}$$

10.3.4 Drift Compensation

In this stage, we need to ensure all nodes clocks converge their clock rate to a common global clock cycle, this is achieved by correcting the time period $T_i(k+1)$ for all nodes such that $T^{global} \approx T_i(k+1) + C_{dr_i} \approx T_j(k+1) + C_{dr_j}$ Where C_{dr_i} is the drift correction for the node i. We obtain the approximation of the common clock cycle T^{global} as

$$T^{global} = \lambda(\tau_i(k) - \tau_i(k-1)) + ((1-\lambda)(R)) \tag{10.9}$$

Where λ denotes the adaptive weight factor, R is the constant which represents the time period between current round (k) and next round $(k+1)$, $\tau_i(k)$ and $\tau_i(k-1)$ are the time estimates for the present round k and previous round $k-1$ respectively. Finally, the drift correction $C_{dr_i}(k)$ is computed in Equation (10.10)

$$\mathrm{C}_{\mathrm{dr_i}}(k) = (\lambda(\tau_i(k) - \tau_i(k-1)) + ((1-\lambda)(R))) - R \tag{10.10}$$

10.3.5 Adaptive Exponential Weight selection

This stage describes how the proposed algorithm selects the best value of adaptive weight factor (λ) for prediction. This parameter plays a vital role in the performance of the algorithm; it dictates the closeness of the predicted value to the most recent estimate. Our approach was motivated by several research works, which show the influence of recent clock skew more than the prior ones due to the change in ambient conditions Sommer and Wattenhofer [2008]; Simeone and Spagnolini [2007]; Aissaoua et al. [2017].

Our algorithm operates in two stages; the first state is called *learning stage* while

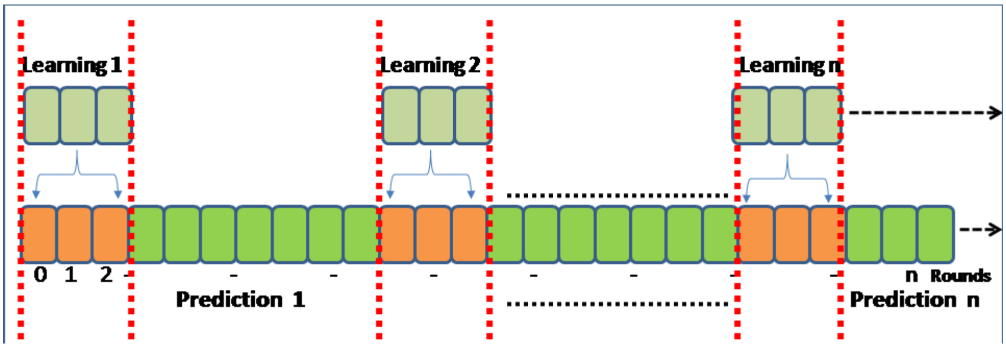


Figure 10.2: Adaptive λ selecting phases. The algorithm executes the learning stage and prediction stage concurrently. The value of λ estimated in the learning stage is used in the prediction stage for prediction.

the second stage is the *prediction stage*. Both stages execute concurrently such that the *learning stage* only evaluates the best value of λ and forwards it to the *prediction stage* for onward prediction. Note that learning occurs only at specific round intervals referred to as *learning period*. During the learning period, the algorithm maintains the existing λ value until the end of the current learning stage; it then replaces the existing λ with the new value only when the new λ value differs from the existing value. Refer to Figure 10.2 for a pictorial representation of the stages.

To evaluate the best value of λ at each *learning interval*, we generate 49 different values of λ within the interval [0, 1: 0.02] with 0.02 as the first value and 0.98 as the last value. For each value, we generate time series, and for each series, we calculate the Mean Squared Error (MSE), then we choose the λ value with small MSE to minimize the prediction errors and forward it to the *prediction stage* for onward prediction. The MSE is used to measure the prediction accuracy of the algorithm with different λs. Equation (10.11) presents MSE.

$$MSE_{(k)} = \sum_{i=0}^{k} \frac{(x_k - (\bar{x})_{(k)})^2}{k} \tag{10.11}$$

where x is the observed value in round k, $\bar{x}$ is the predicted value of DES in the same round.

10.4 Mathematical Analysis

This section analyze the effect of Synchronization Algorithm With an Exponential Weight (SAWEW) in two instances

1. With only $\mathrm{C}'_{\mathrm{es_i}}(\mathrm{k})$ (offset compensation)
2. With both $\mathrm{C}'_{\mathrm{es_i}}(\mathrm{k})$ and $\mathrm{C}_{\mathrm{dr_i}}(k)$ (offset + drift compensation)

Before we begin the main proof let us understand some of the system properties

1. The real-time t_i and sensor i local clock counter is related by

$$t_i(k) = \sum_{c=1}^{\tau_i(k)} \mu_i(c) \tag{10.12}$$

   where $\tau_i(k)$ is the local physical clock counter for sensor i at the time of k-th pulse broadcast. $t_i(k)$ is the real-time of k-th pulse broadcasted by sensor i while $\mu_i(c)$ represents time period of clock cycle. The time period of clock cycle $\mu_i(c)$ is bounded by the inequality

$$(1-\varrho)\mu \leq \mu_i(c) \leq (1+\varrho)\mu \tag{10.13}$$

2. The local tick number of sensor i at which it receives the pulse of sensor j at round k is denoted by $\tau_j^i(k)$. With assumption of negligible or zero transmission delay between sensor i and sensor j, sensor i reads the clock timing of the pulse from sensor j within the bound.

$$\sum_{c=1}^{\tau_j^i(k)-1} \mu_i(c) \leq t_j(k) \leq \sum_{c=1}^{\tau_j^i(k)} \mu_i(c) \tag{10.14}$$

   Note that, Equation (10.14) holds only when sensor i receives pulse from sensor j in round k.

Lemma 1. *Local clock difference* $\tau_i^j(k)$ - $\tau_i(k)$ *and the time difference* $t_j(k)$ - $t_i(k)$ *is related as*

$$\tau_i^j(k) - \tau_i(k) = (t_j(k) - t_i(k)))/\beta_i^j(k) + \alpha_i^j(k), \tag{10.15}$$

where $\beta_j^i \in [\ \mu(1\text{-}\ \varrho),\ \mu(1\text{+}\ \varrho)]$ *and* $\alpha_j^i(k) \in [\text{-}1,\ 1]$.

Proof. See Appendix A ☐

Lemma 2. *The local clock difference* $\tau_i^j(k+1)$ - $\tau_i(k)$ *and the time difference* $t_j(k+1)$ - $t_j(k)$ *is related as*

$$\tau_i(k+1) - \tau_i(k) = (t_i(k+1) - t_i(k)))/\beta_i(k) + \alpha_i(k), \tag{10.16}$$

where $\beta_i \in [\ \mu(1\text{-}\ \varrho),\ \mu(1\text{+}\ \varrho)\ ]$ *and* $\alpha_i(k) \in [\text{-}1,\ 1]$.

Proof. See Appendix B ☐

10.4.1 Convergence Analysis of SAWEW with $\mathrm{C}'_{\mathrm{es_i}}(\mathrm{k})$ only

Let's begin with some basic concepts considered in the proof. The state variable *x(k)* is control by linear recursive law define as

$$x(k) = A(k).x(k-1). \tag{10.17}$$

where $A(k)$ is a row stochastic matrix whose entries are all non-negative and its row sums are all 1. $A(k)$ is given by

$$A(k) = \sum_{j=1}^{n} A_{i,j}(k) = 1. \tag{10.18}$$

It can also be written in terms of product matrix as:

$$P(k) = A(k).A(k-1)......A(1). \tag{10.19}$$

Now, considering only $\mathrm{C}'_{\mathrm{es}}(\mathrm{k})$, SAWEW algorithm estimates its counter value for the next pulse $\tau(k+1)$ recursively as:

$$\tau_i(k+1) - \tau_i(k) = \lambda \sum_{n=0}^{k-1} \left((1-\lambda)^n (\frac{1}{|In_i(k-n)|} \sum_{j \in In_i(k-n)} \tau_j^i(k-n) - \tau_i(k-n)) \right) + R \tag{10.20}$$

Where λ is a weight factor, R is the original number of clock ticks between pulses.

We use the definition of Dobrushin semi-norm define on the real vector $\mathbf{x}$ to calculate maximum pulse skew differences of sensors in round k.

$$\delta(\boldsymbol{x}) = \underset{i,j}{max} \left| x_i - x_j \right| . \tag{10.21}$$

We set the vector $\mathbf{x} = \mathbf{t}$(k). where $\mathbf{t}$(k) is a column vector, and also set $\mathbf{t}(\mathrm{k}) = [t_1(\mathrm{k}), ...,t_n(\mathrm{k})]^T$. With help of Dobrushin semi-norm properties under the following condition define on vector $\mathbf{x}$ and $\mathbf{y} \in \mathbb{R}^n$, and any scaler m.

- 1. $\delta(\boldsymbol{x}) \geq 0$,
- 2. $\delta(\mathrm{m}\boldsymbol{x}) = |\mathrm{m}|\delta(\boldsymbol{x})$, and
- 3. $\delta(\boldsymbol{x} + \boldsymbol{y}) \leq \delta(\boldsymbol{x}) + \delta(\boldsymbol{y})$

We calculate an upper bound of the maximum pulse skew. The network G_t is interpreted as $\boldsymbol{A}(k)$ in order to estimate the averaging operation of sensors. $\boldsymbol{A}(k)$ is represented as:

$$\boldsymbol{A}(k)_{i,j} = \begin{cases} \lambda \sum_{n=0}^{k-1} \left((1-\lambda)^n \frac{1}{|In_i(k-n)|} \right) + (1-\lambda)^k \frac{1}{|In_i(0)|} & \text{(i,j)} \in V \\ 0 & \text{else} \end{cases} \tag{10.22}$$

Lemma 3. *Matrix* $\boldsymbol{A}(k)$ *is stochastic since its row summation*$\sum_{j=1}^{n} A_{i,j}(k)$ *is equal to one for* $1 \leq j \leq n$*.*

Proof. See Appendix E ☐

The Dobrushin semi-norm is defined on the real matrix $\boldsymbol{A}$ as follows:

$$\delta \boldsymbol{A} = \underset{\boldsymbol{x} \in \mathbb{R}^n, \delta(\boldsymbol{x}) \neq 0}{sup} \frac{\delta(\boldsymbol{A}\boldsymbol{x})}{\delta(\boldsymbol{x})}. \tag{10.23}$$

Lemma 4. *Equation (10.20) can be transform to real time in matrix form as*

$$t(k+1) = A(k)t(k) + b(k) + c(k), \tag{10.24}$$

where $A(k)$ represent weighted adjacency matrix, which satisfy

$$\boldsymbol{A}_{i,j}(k) = \begin{cases} \lambda \sum_{n=0}^{k-1} \left((1-\lambda)^n \frac{1}{|In_i(k-n)|} \right) + (1-\lambda)^k \frac{1}{|In_i(k-k)|} & if j \in In_i(k-n) \\ 0 & else, \end{cases} \tag{10.25}$$

$$b_i(k) = \frac{\beta_i(k)}{\overline{\beta_i^j}(k)} \Big(A(k)t(k)_i - t(k)_i \Big), \tag{10.26}$$

$$c_i(k) = (-\alpha_i(k) + \overline{\alpha_i^j}(k) + R)\beta_i(k). \tag{10.27}$$

with existence of some $\beta_i(k)$, $\overline{\beta_i^j}(k) \in$ [μ(1- ϱ), μ(1+ ϱ)] and $\alpha_i(k)$, $\bar{\alpha}_i(k) \in$ [-1, 1].

Proof. Substitute the value of $\tau_i^j(k)$ - $\tau_i(k)$ from Lemma 2 in Equation (10.20), we get

$$\begin{aligned} \tau_i(k+1) - \tau_i(k) &= \lambda \sum_{n=0}^{k-1} \Bigg((1-\lambda)^n \frac{1}{|In_i(k-n)|} \sum_{j \in In_i(k-n)} (\tau_j^i(k-n) - \tau_i(k-n) \Bigg) + R \\ &= \lambda \sum_{n=0}^{k-1} \Bigg((1-\lambda)^n \frac{1}{|In_i(k-n)|} \sum_{j \in In_i(k-n)} (t_j(k-n) - t_i(k-n))/\beta_j^i(k-n) + \alpha_i^j(k-n) \Bigg) + R \\ &= \lambda \sum_{n=0}^{k-1} \Bigg((1-\lambda)^n \frac{1}{|In_i(k-n)|} \sum_{j \in In_i(k-n)} (t_j(k-n) - t_i(k-n))/\beta_j^i(k-n) \Bigg) + \end{aligned}$$

$$\lambda \sum_{n=0}^{k-1} \Bigg((1-\lambda)^n \frac{1}{|In_i(k-n)|} \sum_{j \in In_i(k-n)} \alpha_i^j(k-n)/ \Bigg) + R \tag{10.28}$$

The lower bound can be calculated by using largest value of $\beta_i^j(k)$ and smallest value of $\alpha_i^j(k)$ i.e μ(1+ ϱ) and -1. substituting these values gives

$$\begin{aligned}\tau_i(k+1)-\tau_i(k) &= \lambda\sum_{n=0}^{k-1}\Bigg((1-\lambda)^n\frac{1}{|In_i(k-n)|}\sum_{j\in In_i(k-n)}(t_j(k-n)-t_i(k-n))/\mu(1+\varrho)\Bigg)- \\ &\frac{1}{|In_i(k-n)|}\sum_{j\in In_i(k-n)}1\Bigg)+R \\ &= \lambda\sum_{n=0}^{k-1}\Bigg((1-\lambda)^n\frac{1}{|In_i(k-n)|}\frac{1}{\mu(1+\varrho)}\sum_{j\in In_i(k-n)}(t_j(k-n)-t_i(k-n))\Bigg)-1+R\end{aligned} \tag{10.29}$$

The upper bound can be calculated using smallest value of $\beta_i^j(k)$ and largest value of $\alpha_i^j(k)$ i.e μ(1-ϱ) and 1. The result is Equation (10.30)

$$\tau_i(k+1)-\tau_i(k) = \lambda\sum_{n=0}^{k-1}\Bigg((1-\lambda)^n\frac{1}{|In_i(k-n)|}\frac{1}{\mu(1-\varrho)}\sum_{j\in In_i(k-n)}(t_j(k-n)-t_i(k-n))\Bigg)+1+R \tag{10.30}$$

Combining Equation (10.29) and Equation (10.30) shows the existence of $\bar{\beta}_j^i \in$ [μ(1- ϱ), μ(1+ ϱ)] and $\bar{\alpha}_j^i$(k) $\in$ [-1, 1] such that

$$\tau_i(k+1)-\tau_i(k) = \lambda\sum_{n=0}^{k-1}\Bigg((1-\lambda)^n\frac{1}{|In_i(k-n)|\,\overline{\beta_i^j}(k)}\sum_{j\in In_i(k-n)}(t_j(k-n)-t_i(k-n))\Bigg)+\overline{\alpha_i^j}(k)+R \tag{10.31}$$

Combining Lemma 2 and Equation (10.31), gives

$$\tau_i(k+1)-\tau_i(k) = \lambda\sum_{n=0}^{k-1}\Bigg((1-\lambda)^n\frac{\beta_i(k)}{|In_i(k-n)|\,\overline{\beta_i^j}(k)}\sum_{j\in In_i(k-n)}(t_j(k-n)-t_i(k-n))\Bigg)+\Bigg(\alpha_i^j(k)+\overline{\alpha_i^j}(k)+R\Bigg)\beta_i(k) \tag{10.32}$$

Let $\gamma_i(k)$ = ($\frac{\beta_i(k)}{\bar{\beta}_i^j(k)}$) - 1). where $\gamma_i(k)$ is the minimal positive entry for the matrix A(k) putting $\gamma_i(k)$ in Equation (10.32) and adding t_i(k) on both sides of the Equation (10.32) gives

$$\begin{aligned}t_i(k+1) = \lambda\sum_{n=0}^{k-1}\Bigg((1-\lambda)^n t_j(k-n)\Bigg)+\lambda\sum_{n=0}^{k-1}\Bigg((1-\lambda)^n\frac{\gamma_i(k-n)}{|In_i(k-n)|}\sum_{j\in In_i(k-n)}(t_j(k-n)-t_i(k-n))\Bigg)+ \\ \Bigg(-\alpha_i^j(k)+\overline{\alpha_i^j}(k)+R\Bigg)\beta_i(k)\end{aligned} \tag{10.33}$$

$$t_i(k+1) = (A(k)t(k))_i + \gamma_i(k)\Big((A(k)t(k)_i - t_i(k))\Big) + \Big(-\alpha_i^j(k) + \overline{\alpha_i^j}(k) + R\Big)\beta_i(k) \tag{10.34}$$

or in vector form as Equation (10.24) $\square$

Lemma 5. *Semi-norm of b(k) is*

$$\delta(b(k)) = \delta\left(\left(\frac{\beta_i(k)}{\bar{\beta}_j^i(k)} - 1\right)((A(k)t(k)) - t(k))\right) \leq \frac{4\varrho}{1-\varrho} + \delta(k) \tag{10.35}$$

where $\delta(k)$ represent clock skew $\delta(t(k))$ in round k.

Proof. Using Dobrushin semi-norm define on real vectors **X** by

$$\delta(X) = max_{i,j}\left|x_i - x_j\right|. \tag{10.36}$$

Since A(k) is a stochastic matrix, i.e, $\sum_j$ [A(k)]$_{i,j}$ = 1. suppose we have t_{i1} and t_{i2} such that $\delta(k)$ = ($t_{i1}(k)$ - $t_{i2}(k)$), then $t_{i2}(k) \leq$ A(k)t(k) $\leq t_{i1}(k)$, which implies

$$\delta\left((A(k)t(k)) - t(k))\right)_i \leq \delta(k) \tag{10.37}$$

The value (($\frac{\beta_i(k)}{\bar{\beta}_i^j(k)}$) - 1) is within the bound

$$-\frac{2\varrho}{1+\varrho} \leq \left(\frac{\beta_i(k)}{\bar{\beta}_j^i}) - 1\right) \leq +\frac{2\varrho}{1-\varrho} \tag{10.38}$$

Considering Equation (10.37) and Equation (10.38) with Equation (10.26), we get Equation (10.39)

$$-\frac{2\varrho}{1-\varrho}\delta(k) \leq b_i(k) \leq \frac{2\varrho}{1+\varrho}\delta(k) \tag{10.39}$$

which means,

$$\delta(b(k)) \leq max_{i1}b_{i1}(k) - min_{i2}b_{i2}(k) \leq \frac{4\varrho}{1-\varrho}\delta(k) \tag{10.40}$$

□

Lemma 6. *By assuming the original number of clock ticks between pulses R>2, the maximum difference of the vector c(k) is*

$$\delta(c(k)) \leq (4 + 2\varrho R)\mu \tag{10.41}$$

Proof. If R >2, we have $c_i(k) \geq 0$ in Equation (10.27), therefore $c_i(k)$ is within the bound:

$$(-2 + R)\mu(1 - \varrho) \leq c_i(k)) \leq (2 + R)\mu(1 + \varrho) \tag{10.42}$$

Which shows that

$$\begin{aligned} \delta(c(k)) &\leq \max_{i1,i2} |c_{i1}(k) - c_{i2}(k)| \\ &\leq (2 + R)\mu(1 + \varrho) - (-2 + R)\mu(1 - \varrho) \\ &= (4 + 2\varrho R)\mu. \end{aligned} \tag{10.43}$$

□

Theorem 1. *The clock skew $\delta(k)$ is within a bound:*

$$\delta(k) \leq \left(1 - \gamma + \frac{4\varrho}{1 - \varrho}\right)^k \delta(0) + (4 + 2\varrho R)\mu \frac{(1 - (1 - \gamma)^k}{\gamma} \tag{10.44}$$

Proof. Transforming Equation (10.20) to precision domain using Lemma 6 and Lemma 7 we get:

$$\delta(k + 1) \leq \left(\delta(A(k)) + \frac{4\varrho}{1 - \varrho)}\right)\delta(k) + \delta(c(k)) \tag{10.45}$$

The right hand side of Equation (10.44) gives:

$$
\begin{aligned}
\delta(k) &\leq \prod_{n=0}^{k-1}\left(\delta(A(k)) + \frac{4\varrho}{1-\varrho}\right)\delta(0) + \sum_{l=0}^{k-1}\left(\prod_{n=l}^{k-1}\delta(A(n))\delta(c(k))\right) \\
&= \left(1-\gamma+\frac{4\varrho}{1-\varrho}\right)^{k}\delta(0) + (4+2\varrho R)\mu\sum_{l=0}^{k-1}(1-\gamma)^{k-l} \\
&\leq \left(1-\gamma+\frac{4\varrho}{1-\varrho}\right)^{k}\delta(0) + (4+2\varrho R)\mu\frac{(1-(1-\gamma)^{k}}{\gamma},
\end{aligned}
\tag{10.46}
$$

Assume $\gamma > \frac{4\varrho}{1-\varrho}$, and let k $\to \infty$ and find

$$
\lim_{k\to\infty}\delta(k) \leq (4+2\varrho R)\frac{\mu}{\gamma}. \tag{10.47}
$$

$\square$

10.4.2 Convergence Analysis of SAWEW with $\mathrm{C}'_{\mathrm{es_i}}(\mathrm{k}) + \mathrm{C}_{\mathrm{dr_i}}(\mathrm{k})$

Lemma 7. *Equation* (10.20) *can be written as Equation* (10.48) *considering* $\mathrm{C}_{\mathrm{dr_i}}(\mathrm{k})$

$$
\begin{aligned}
\tau_i(k+1)-\tau_i(k) = \Bigg(\lambda\sum_{n=0}^{k-1}(1-\lambda)^{n}\Big(&\Big(\frac{1}{|In_i(k-n)|}\sum_{j\in In_i(k-n)}\tau_j^{i}(k-n)-\tau_i(k-n)\Big)- \\
&\Big(\frac{1}{|In_i(k-1-n)|}\sum_{j\in In_i(k-1-n)}\tau_j^{i}(k-1-n)-\tau_i(k-1-n)\Big)\Big)\Bigg)\lambda+(1-\lambda)R
\end{aligned}
\tag{10.48}
$$

Lemma 8. *Equation* (10.48) *can be reformulated in matrix form as*

$$
t(k+1) = A(k)t(k) + b(k) + c(k) \tag{10.49}
$$

With

$$
\boldsymbol{A}_{i,j}(k) = \begin{cases} \lambda\sum_{n=0}^{k-1}\left((1-\lambda)^{n}\frac{1}{|In_i(k-n)|}\right) & \mathit{if} j \in In_i(k-n) \\ 0 & \mathit{else} \end{cases} \tag{10.50}
$$

and

$$b_i(k) = \frac{\lambda\beta_i(k)}{\overline{\beta_i^j}(k)}\Bigg(\Big(A(k)t(k)_i - t(k)_i\Big) - \Big(A(k-1)t(k-1)_i - t(k-1)_i\Big)\Bigg) \tag{10.51}$$

and

$$c_i(k) = (-\alpha_i(k) - \alpha_i(k-1) + \lambda(\overline{\alpha_i^j}(k)) + \lambda(\overline{\alpha_i^j}(k-1)) + (1-\lambda)R)\beta_i(k) \tag{10.52}$$

with existence of some $\beta_i(k)$, $\overline{\beta_i^j}(k) \in$ *[μ(1- ϱ), μ(1+ ϱ)] and* $\alpha_i(k)$, $\bar{\alpha}_i(k) \in$ *[-1, 1].*

Proof. Substitute the value of $\tau_i^j(k)$ - $\tau_i(k)$ from lemma 2 in Equation (10.48), we get

$$\begin{aligned}
\tau_i(k+1) - \tau_i(k) = {} & \Bigg(\lambda\sum_{n=0}^{k-1}(1-\lambda)^n\Bigg(\Big(\frac{1}{|In_i(k-n)|}\sum_{j\in In_i(k-n)}\tau_j^i(k-n) - \tau_i(k-n)\Big) - \\
& \Big(\frac{1}{|In_i(k-1-n)|}\sum_{j\in In_i(k-1-n)}\tau_j^i(k-1-n) - \tau_i(k-1-n)\Big)\Bigg)\Bigg)\lambda + (1-\lambda)R \\
= {} & \Bigg(\lambda\sum_{n=0}^{k-1}(1-\lambda)^n\Big(\frac{1}{|In_i(k-n)|}(t_j(k-n) - t_i(k-n))/\beta_j^i(k-n) + \alpha_i^j(k-n) - \\
& \Big(\frac{1}{|In_i(k-1-n)|}(t_j(k-1-n) - t_i(k-1-n))/\beta_j^i(k-1-n) + \alpha_i^j(k-1-n)\Bigg)\lambda + (1-\lambda)R \\
= {} & \Bigg(\lambda\sum_{n=0}^{k-1}(1-\lambda)^n\Big(\frac{\lambda}{|In_i(k-n)|}(t_j(k-n) - t_i(k-n))/\beta_j^i(k-n) + \frac{\lambda}{|In_i(k-n)|}\sum_{j\in In_i(k-n)}\alpha_i^j(k-n) - \\
& \Big(\frac{\lambda}{|In_i(k-1-n)|}(t_j(k-1-n) - t_i(k-1-n))/\beta_j^i(k-1-n) + \frac{\lambda}{|In_i(k-1-n)|}\sum_{j\in In_i(k-1-n)}\alpha_i^j(k-1-n)\Bigg) + (1-\lambda)R
\end{aligned} \tag{10.53}$$

The lower bound can be calculated by using largest value of $\beta_i^j(k)$ and smallest value of

$\alpha_i^j(k)$ i.e $\mu(1+\varrho)$ and -1. substituting the values in Equation (10.53) gives

$$\begin{aligned}\tau_i(k+1)-\tau_i(k) = {} & \left(\lambda\sum_{n=0}^{k-1}(1-\lambda)^n(\frac{\lambda}{|In_i(k-n)|}(t_j(k-n)-t_i(k-n))/(1+\varrho)\mu - \frac{\lambda}{|In_i(k-n)|}\sum_{j\in In_i(k-n)} 1 - \right.\\ & \left.(\frac{\lambda}{|In_i(k-1-n)|}(t_j(k-1-n)-t_i(k-1-n))//(1+\varrho)\mu - \frac{\lambda}{|In_i(k-1-n)|}\sum_{j\in In_i(k-1-n)} 1\right) + (1-\lambda)R\\ = {} & \left(\lambda\sum_{n=0}^{k-1}(1-\lambda)^n(\frac{\lambda}{|In_i(k-n)|\,(1+\varrho)\mu}(t_j(k-n)-t_i(k-n)) - \right.\\ & \left.(\frac{\lambda}{|In_i(k-1-n)|\,(1+\varrho)\mu}(t_j(k-1-n)-t_i(k-1-n))\right) + \lambda(-1)+\lambda(-1)+(1-\lambda)R\\ = {} & \left(\lambda\sum_{n=0}^{k-1}(1-\lambda)^n(\frac{\lambda}{|In_i(k-n)|\,\beta_i^j(k-n)}(t_j(k-n)-t_i(k-n)) - \right.\\ & \left.(\frac{\lambda}{|In_i(k-1-n)|\,\beta_i^j(k-1-n)}(t_j(k-1-n)-t_i(k-1-n))\right) - \lambda\alpha_i^j(k) - \lambda\alpha_i^j(k-1) + (1-\lambda)R\end{aligned} \tag{10.54}$$

With similar derivation, the upper bound can be calculated using smallest value of $\beta_i^j(k)$ and largest value of $\alpha_i^j(k)$ i.e $\mu(1\text{-}\varrho)$ and 1. The result is Equation (10.55)

$$\begin{aligned}\tau_i(k+1)-\tau_i(k) = {} & \left(\lambda\sum_{n=0}^{k-1}(1-\lambda)^n(\frac{\lambda}{|In_i(k-n)|\,\beta_i^j(k-n)}(t_j(k-n)-t_i(k-n)) - \right.\\ & \left.(\frac{\lambda}{|In_i(k-1-n)|\,\beta_i^j(k-1-n)}(t_j(k-1-n)-t_i(k-1-n))\right) + \lambda\alpha_i^j(k) + \lambda\alpha_i^j(k-1) + (1-\lambda)R\end{aligned} \tag{10.55}$$

Combining Equation (10.54) and Equation (10.55) shows the existence of $\bar{\beta}_j^i \in [\ \mu(1\text{-}\varrho), \mu(1+\varrho)]$ and $\bar{\alpha}_j^i$(k) $\in$ [-1, 1] such that

$$\begin{aligned}\tau_i(k+1)-\tau_i(k) = {} & \left(\lambda\sum_{n=0}^{k-1}(1-\lambda)^n(\frac{\lambda}{|In_i(k-n)|\,\overline{\beta_i^j}(k-n)}(t_j(k-n)-t_i(k-n)) - \right.\\ & \left.(\frac{\lambda}{|In_i(k-1-n)|\,\overline{\beta_i^j}(k-1-n)}(t_j(k-1-n)-t_i(k-1-n))\right) + \lambda\overline{\alpha_i^j}(k) + \lambda\overline{\alpha_i^j}(k-1) + (1-\lambda)R\end{aligned} \tag{10.56}$$

Combining lemma 1 and Equation (10.56) resulted to Equation (10.57)

$$
\begin{aligned}
t_i(k+1) - t_i(k) = {} & \Bigg(\lambda \sum_{n=0}^{k-1} (1-\lambda)^n \big(\frac{\lambda\beta_i(k)}{|In_i(k-n)|\,\overline{\beta_i^j}(k-n)}(t_j(k-n) - t_i(k-n))- \\
& \big(\frac{\lambda\beta_i(k)}{|In_i(k-1-n)|\,\overline{\beta_i^j}(k-1-n)}(t_j(k-1-n) - t_i(k-1-n))\Bigg) \\
& +\Bigg(-(\alpha_i(k) - \alpha_i(k-1) + \lambda\overline{\alpha_i^j}(k) + \lambda\overline{\alpha_i^j}(k-1) + (1-\lambda)R\Bigg)\beta_i(k) \\
= {} & \frac{\lambda\beta_i(k)}{\overline{\beta_i^j}(k)}\Bigg(\lambda \sum_{n=0}^{k-1} (1-\lambda)^n \big(\frac{1}{|In_i(k-n)|}(t_j(k-n) - t_i(k-n))- \\
& \big(\frac{1}{|In_i(k-1-n)|}(t_j(k-1-n) - t_i(k-1-n))\Bigg) \\
& \Bigg(-(\alpha_i(k) - \alpha_i(k-1) + \lambda\overline{\alpha_i^j}(k) + \lambda\overline{\alpha_i^j}(k-1) + (1-\lambda)R\Bigg)\beta_i(k)
\end{aligned}
\tag{10.57}
$$

Let $\gamma_i(k) = \frac{\lambda\beta_i(k)}{\overline{\beta_i^j}(k)}$

putting $\gamma_i(k)$ in Equation (10.57) and adding t_i(k) on both sides of the Equation gives

$$
\begin{aligned}
t_i(k+1) = {} & \Bigg(\lambda \sum_{n=0}^{k-1} (1-\lambda)^n \frac{1}{|In_i(k-n)|}(t_j(k-n)\Bigg) + \gamma_i(k)\lambda \sum_{n=0}^{k-1} (1-\lambda)^n \Bigg(\big(\frac{1}{|In_i(k-n)|}(t_j(k-n) - t_i(k-n))- \\
& \big(\frac{1}{|In_i(k-1-n)|}(t_j(k-1-n) - t_i(k-1-n))) + (-(\alpha_i(k) - \alpha_i(k-1) + \lambda(\overline{\alpha_i^j}(k)\Bigg)+ \\
& \Bigg(-(\alpha_i(k) - \alpha_i(k-1) + \lambda(\overline{\alpha_i^j}(k)) + \lambda(\overline{\alpha_i^j}(k-1)) + (1-\lambda)R\Bigg)\beta_i(k) \\
= {} & \Bigg(\lambda \sum_{n=0}^{k-1} (1-\lambda)^n \frac{1}{|In_i(k-n)|}(t_j(k-n)\Bigg) + \lambda \sum_{n=0}^{k-1} (1-\lambda)^n \Bigg(\big(\frac{\gamma_i(k)}{|In_i(k-n)|}(t_j(k-n) - t_i(k-n))- \\
& \big(\frac{\gamma_i(k)}{|In_i(k-1-n)|}(t_j(k-1-n) - t_i(k-1-n))\Bigg) + \Bigg(-(\alpha_i(k) - \alpha_i(k-1) + \lambda(\overline{\alpha_i^j}(k)) + \lambda(\overline{\alpha_i^j}(k-1)) + (1-\lambda)R\Bigg)\beta_i(k)
\end{aligned}
\tag{10.58}
$$

or in vector form

$$
\begin{aligned}
t(k+1) = {} & A(k)t(k) + \gamma_i\Bigg((A(k)t(k)_i - t(k)_i) - (A(k-1)t(k-1)_i - t(k-1)_i)\Bigg)+ \\
& \Bigg(-(\alpha_i(k) - \alpha_i(k-1) + \lambda(\overline{\alpha_i^j}(k)) + \lambda(\overline{\alpha_i^j}(k-1)) + (1-\lambda)R\Bigg)\beta_i(k)
\end{aligned}
\tag{10.59}
$$

□

Lemma 9. *Semi-norm of $b(k)$ is*

$$\delta(b(k)) = \delta\left(\left(\frac{\lambda\beta_i(k)}{\overline{\beta_j^i}(k)} - 1\right)((A(k)t(k)) - t(k))\right) \leq \frac{4\varrho\lambda}{1-\varrho} + \delta(k) \tag{10.60}$$

where $\delta(k)$ represent clock skew $\delta(t(k))$ in round k.

Proof. Using Dobrushin semi-norm define on real vectors **X** by

$$\delta(X) = max_{i,j} \left| x_i - x_j \right|. \tag{10.61}$$

Since A(k) is a stochastic matrix, i.e, $\sum_j$ [A(k)]$_{i,j}$ $= 1.$ suppose we have t_{i1} and t_{i2} such that $\delta(k)$ = ($t_{i1}(k)$ - $t_{i2}(k)$), then $t_{i2}(k) \leq$ A(k)t(k) $\leq t_{i1}(k)$, which implies

$$\delta\left((A(k)t(k)) - t(k))\right)_i \leq \delta(k) \tag{10.62}$$

The value (($\frac{\lambda\beta_i(k)}{\bar{\beta}_i^j(k)}$) - 1) is within the bound

$$-\frac{2\varrho\lambda}{1+\varrho} \leq \left(\frac{\lambda\beta_i(k)}{\bar{\beta}_j^i}) - 1\right) \leq +\frac{2\varrho\lambda}{1-\varrho} \tag{10.63}$$

Considering Equation (10.62) and Equation (10.63) with Equation (10.51), we get Equation (10.64)

$$-\frac{2\varrho\lambda}{1-\varrho}\delta(k) \leq b_i(k) \leq \frac{2\varrho\lambda}{1+\varrho}\delta(k) \tag{10.64}$$

which means that,

$$\delta(b(k)) \leq max_{i1}b_{i1}(k) - min_{i2}b_{i2}(k) \leq \frac{4\varrho\lambda}{1-\varrho}\delta(k) \tag{10.65}$$

□

Lemma 10. *By assuming the original number of clock ticks between pulses R>2, the* *maximum difference of the vector c(k) is*

$$\delta(c(k)) \leq ((4\lambda + 2\varrho R - 2\varrho R\lambda)\mu \tag{10.66}$$

Proof. If R >2, we have $c_i(k)$ ≥0 in Equation (10.52), therefore $c_i(k)$ is within the bound:

$$(-2\lambda - \lambda R + R)\mu(1-\varrho) \leq c_i(k)) \leq (2\lambda - \lambda R + R)\mu(1+\varrho) \tag{10.67}$$

Which shows that

$$\begin{aligned} \delta(c(k)) &\leq \underset{i1,i2}{max} |c_{i1}(k) - c_{i2}(k)| \\ &\leq (2\lambda - \lambda R + R)\mu(1+\varrho) - (-2\lambda - \lambda R + R)\mu(1-\varrho) \\ &= (4\lambda + 2\varrho R - 2\varrho R\lambda)\mu. \end{aligned} \tag{10.68}$$

□

Theorem 2. *The clock skew δ(k) is within a bound:*

$$\delta(k) \leq \left(1 - \gamma + \frac{4\varrho\lambda}{1-\varrho}\right)^k \delta(0) + ((4\lambda + 2\varrho R - 2\varrho R\lambda)\mu\frac{1-(1-\gamma)^k}{\gamma} \tag{10.69}$$

Proof. Transforming Equation (10.48) to precision domain using Lemma 9 and Lemma 10 we get:

$$\delta(k+1) \leq \left(\delta(A(k)) + \frac{4\varrho\lambda}{1-\varrho)}\right)\delta(k) + \delta(c(k)) \tag{10.70}$$

The right hand side of Equation (10.44) gives:

$$\begin{aligned}\delta(k) &\leq \prod_{n=0}^{k-1}\left(\delta(A(k)) + \frac{4\varrho\lambda}{1-\varrho}\right)\delta(0) + \sum_{l=0}^{k-1}\left(\prod_{n=l}^{k-1}\delta(A(n))\delta(c(k))\right) \\ &= \left(1-\gamma+\frac{4\varrho\lambda}{1-\varrho}\right)^{k}\delta(0) + (4\lambda+2\varrho R-2\varrho R\lambda)\mu\sum_{l=0}^{k-1}(1-\gamma)^{k-l} \\ &\leq \left(1-\gamma+\frac{4\varrho\lambda}{1-\varrho}\right)^{k}\delta(0) + (4\lambda+2\varrho R-2\varrho R\lambda)\mu\frac{1-(1-\gamma)^{k}}{\gamma},\end{aligned} \tag{10.71}$$

Assume $\gamma > \frac{4\varrho\lambda}{1-\varrho}$, and let k $\to \infty$ and find

$$\lim_{k\to\infty}\delta(k) \leq ((4\lambda+2\varrho R-2\varrho R\lambda)\frac{\mu}{\gamma}. \tag{10.72}$$

□

In this section, we present a mathematical analysis of the proposed clock synchronization algorithm. The upper bound on the clock skew for the offset and drift compensation have been derived. For the drift compensation, the clock skew is bound above by the term

$$\lim_{k\to\infty}\delta(k) \leq ((4\lambda+2\varrho R-2\varrho R\lambda)\frac{\mu}{\gamma}. \tag{10.73}$$

as derived in Equation (10.72). The term $2\varrho R\lambda < 2\varrho R\ \forall\lambda$ except $\lambda = 1$. Similarly $\lambda \leq 1$. It follows that the term $(4\lambda + 2\varrho R - 2\varrho R\lambda) < 4$ except when $\lambda = 1$. Therefore, the clock skew $\delta(k)$ is guaranteed below $4\frac{\mu}{\gamma}$

10.5 Conclusion

This chapter presents our solution to the clock synchronization problems, the system model and describes the proposed clock synchronization algorithm. We detail the three main functions of the proposed algorithm: the sending function, receiving function, and timer function. Additionally, we discussed how the new algorithm performs the offset compensation, drift compensation, and adaptive weight factor selection. Finally, a mathematical analysis and proofs for the clock skew and convergence rate were presented.

In the next chapter, the performance of proposed algorithm is evaluated. The proposed is compared with the previous work by Han [2018] and Fugger et al. [2015] in static and dynamic settings.

Chapter 11

Simulation Results

This chapter presents the results of extensive simulations conducted under different settings and parameter values. We compare the proposed algorithm with the previous works in Fugger et al. [2015], and Han [2018] in terms of different evaluating functions under static and highly dynamic environment with drifting clocks.

11.1 Simulation Environment

The simulation environment is written in the JAVA programming language and runs on the Windows 10 operating system with Intel(R) Core(TM) i5-4300M CPU (2.60GHz, 2.60 GHz) processor and 8 gigabyte RAM.

11.1.1 Evaluation Functions

The evaluation functions used to compare the proposed algorithm and the previous works are as follows:

11.1.1.1 Maximum Phase Difference

This Maximum Phase Difference function is calculated using Equation (11.1) by considering pulses from the neighboring sensors j to the target sensor i in the same round k:

$$\delta(k) = Max_{i,j} \left| t_i(k) - t_j(k) \right| . \tag{11.1}$$

The value of the function determines how well the nodes in the network synchronized their clocks and determine the clock skew. We need to have a small and stabilized time value of the phased difference between rounds for better synchronization.

11.1.1.2 Round Duration

Round duration dictates the strength of an algorithm to tolerate changes in a network topology. This function is tested under different mobility settings in the simulation. Equation (11.2) is used to calculate the round duration.

$$duration_i(k) = t_i(k+1) - t_i(k). \tag{11.2}$$

11.1.1.3 Pulse Generation Time

Pulse time shows how different pulses broadcasted by different sensors in each round converge to a common interval. Pulses broadcasted by the same sensor are shown with the same color.

11.2 Static Environment

We evaluate the performance of the proposed algorithm under static settings by deploying sensor nodes at the initial strategic locations with no consideration to the node mobility. We consider two scenarios; the first scenario uses a sparsely connected network while the second scenario considers a densely connected network. We want to show that our algorithm adapts to both scenario.

11.2.1 Sparse Scenario

In this setting, ten sensors were randomly scattered in an area of 100 x 100 m^2 as shown in Figure 11.2. We set different sensing ranges for each sensor, enabling a sensor to detect pulses from its neighboring sensors. The average connectivity density $\frac{\|E\|}{\|N\|}$ is set to 4 per node. Sensor i exchanges pulses with sensor j only within the distance $r \in \mathbb{R}$, bounded within a range: $r(1-\epsilon) \leq r_i \leq r(1+\epsilon)$. We set the sensing range variation value $\epsilon = 0.5$ and the average sensing range for all sensors to $r = 30$ m. The algorithm-related parameters are summarized in Table 11.1. All sensors wake up at time 0 and starting to broadcast at time $T_i(0)$. We compare the following algorithms:

1. Clock synchronization using $corr^1 + corr^2$ (Han [2018]), See Section 9.2.1
2. Clock synchronization using $C''_{es} + C_{dr}$ (Bakura et al. [2020]), See Section 10.3

The first algorithm is the algorithm proposed by Han [2018], which improves $corr^1$ Fugger et al. [2015] by proposing additional $corr^2$ for drift compensation. Henceforth, $corr^2$ would be used to refer to the first algorithm ($corr^1 + corr^2$). The second algorithm is our proposed algorithm that improves $corr^2$ by implementing learning techniques for offset and drift compensation to improve clock precision and reduce clock skew. Similarly C_{es} would be used to refer to the second algorithm ($C''_{es} + C_{dr}$).

Figure 11.1 shows a comparison between $corr^2$ and C_{es} in terms of maximum phase difference. The x-axis represents simulations rounds. The time on the y-axis represents the maximum phase difference (skew) in each round. For better synchronization, we need to have a small and stabilized value of time. We can see that in the first seven-round, $corr^2$ phase difference increases because its drift correction has to be delayed until after some number of rounds (i.e., round filter), but as soon as its drift correction is enabled, the phase difference reduces with increase in rounds. On the other hand,

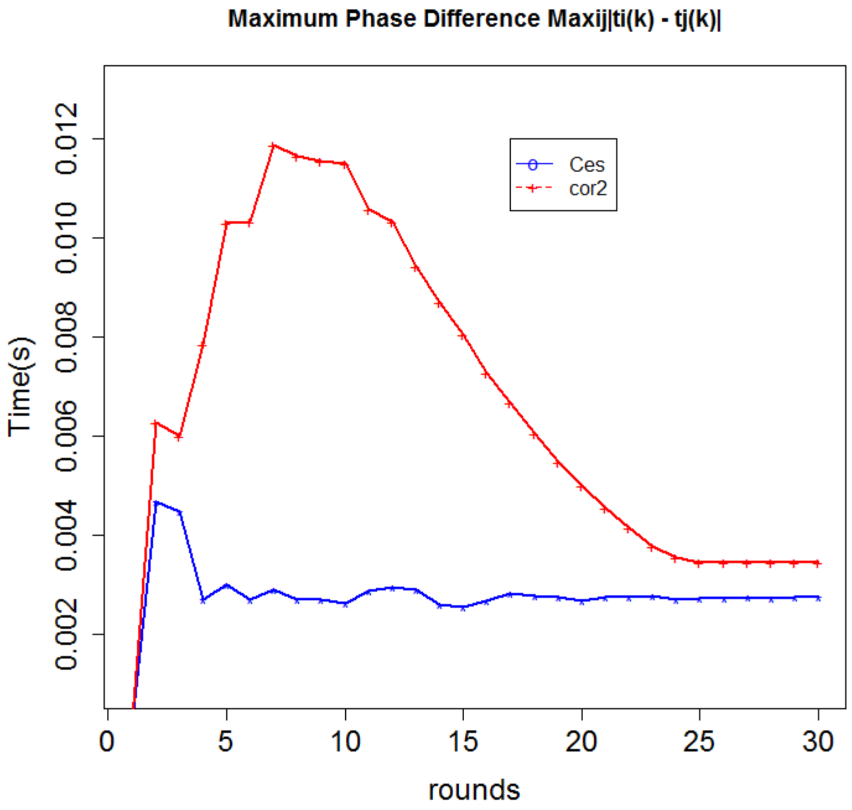


Figure 11.1: Maximum Phase Difference between Rounds for C_{es} and $corr^2$ for a Sparsely connected scene. The C_{es} converges to a stable common time after few rounds of fluctuations; the $corr^2$ takes more rounds before it stabilizes. The C_{es} was able to maintained small phase difference in a sparsely connected scene.

C_{es} converges its phase difference to a smaller and stable value after just a few rounds of fluctuations. There is no need for a round filter in C_{es}, which reduces the initial fluctuations observed in the $corr^2$. Considering the scenario in which sensors have a limited number of neighbors, C_{es} utilizes the previous history to increase prediction accuracy.

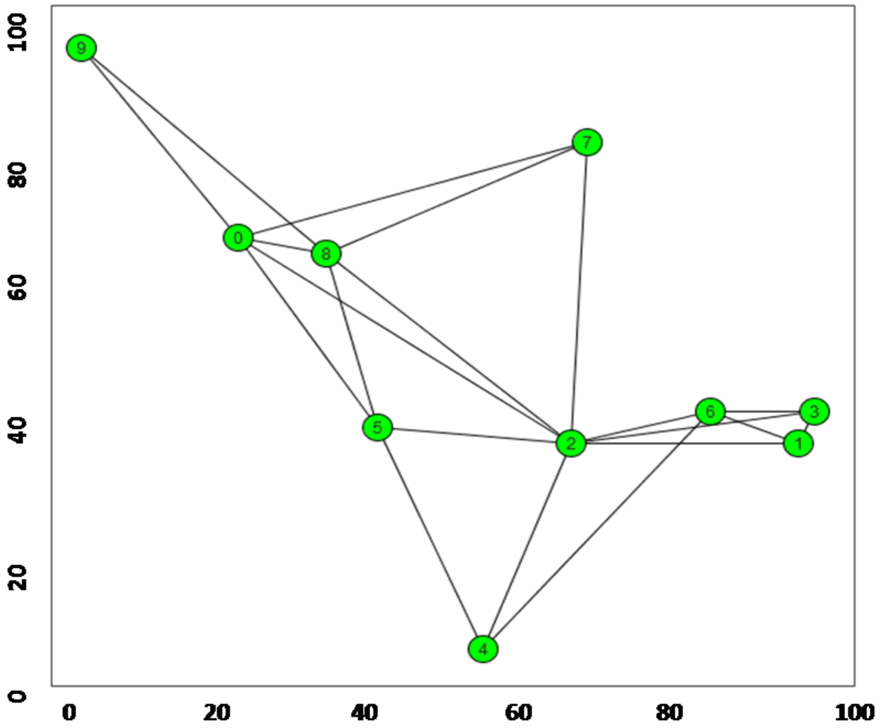


Figure 11.2: The figure shows 10 nodes for sparsely connected Scene in the Simulation

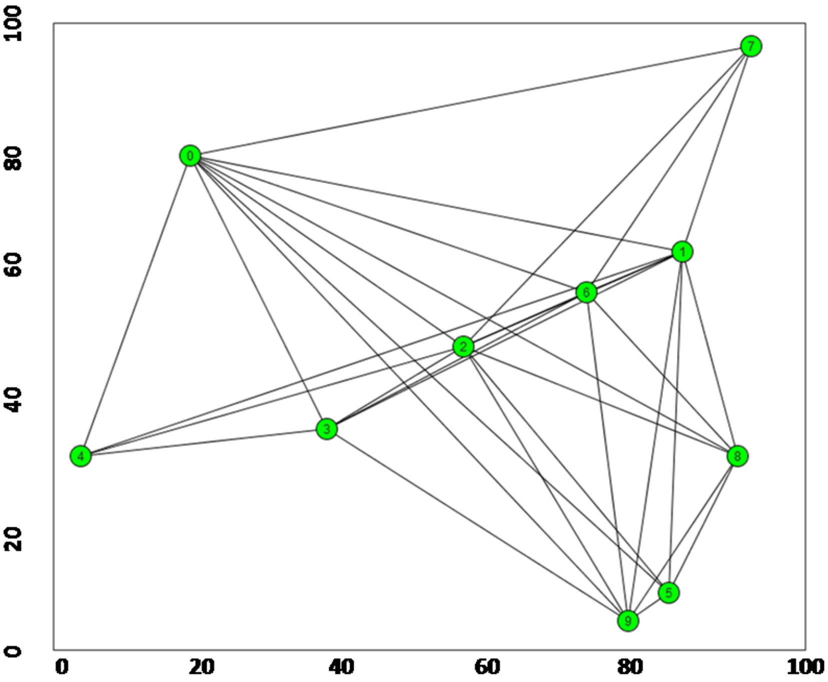


Figure 11.3: The figure shows 10 nodes for a densely connected Scene in the Simulation

Parameter	**Value**
Maximum slot for $Corr^2$	2
Average connectivity	4 nodes
Clock period R	0.03
Sensor frequency variation $\lvert\delta_i\rvert$	≤ 30 %
Round frequency variation $\lvert\delta_k\rvert$	≤ 1 %
$T_i(k)$	$R(1+\delta_i+\delta_k)$
Sensing range variation ϵ	0.5

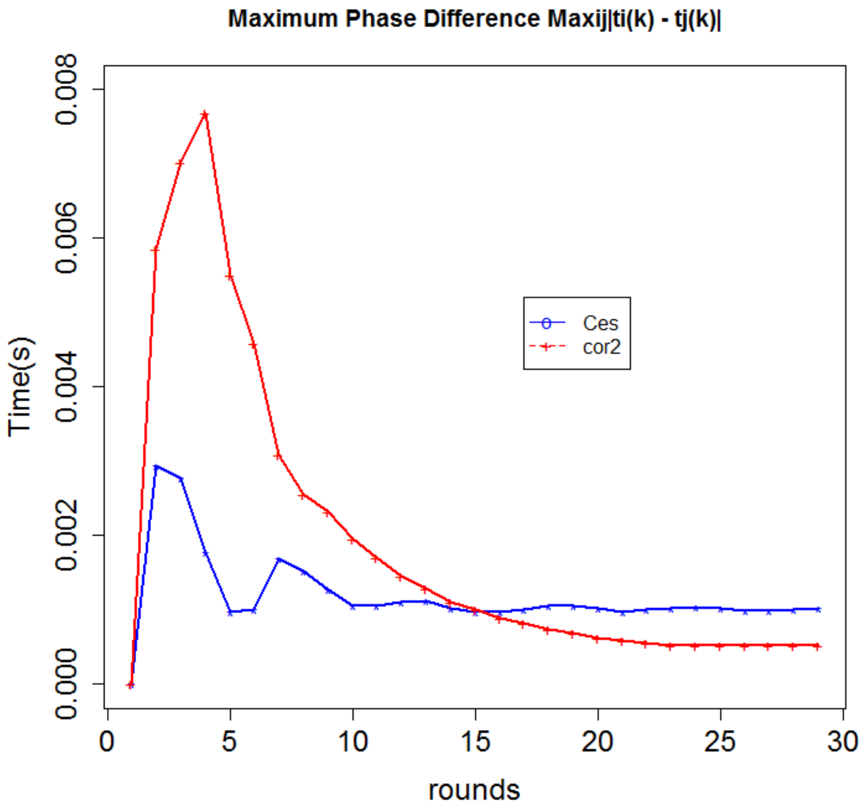


Figure 11.4: Maximum Phase Difference between Rounds for C_{es} and $corr^2$ for a densely connected scene. The $corr^2$ has converged to a common interval after few rounds similar to the C_{es}. After many rounds, $corr^2$ phase differences became smaller than that of C_{es} due to the higher connectivity.

11.2.2 Dense Scenario

In this section, we consider a more connected scene in which sensors have more neighbors. The environment is the same with Section 11.2.1 only that we have changed the average connectivity density $\frac{\|E\|}{\|N\|}$ to six nodes and the average sensing range to $r = 40m$. With these changes in the connectivity density and sensing range, the network becomes more connected than the sparse scene. Figure 11.3 shows the dense scenario with ten nodes scattered in the simulation area. Figure 11.4 shows the comparison between the C_{es} and $Corr^2$ in terms of phase difference between rounds. The $Corr^2$ has dramatically converged its phase difference to a smaller and common interval after few rounds of execution; this behavior is attributed by more number of pulses due to the higher connectivity, unlike in a sparsely connected network. The C_{es} converges its phase difference prior to the $Corr^2$ but with a large phase value. Note that, in a dynamic network, dense connectivity is not always guaranteed, hence the need for an algorithm capable of utilizing limited connectivity without sacrificing the clock precision.

11.3 Dynamic Environment

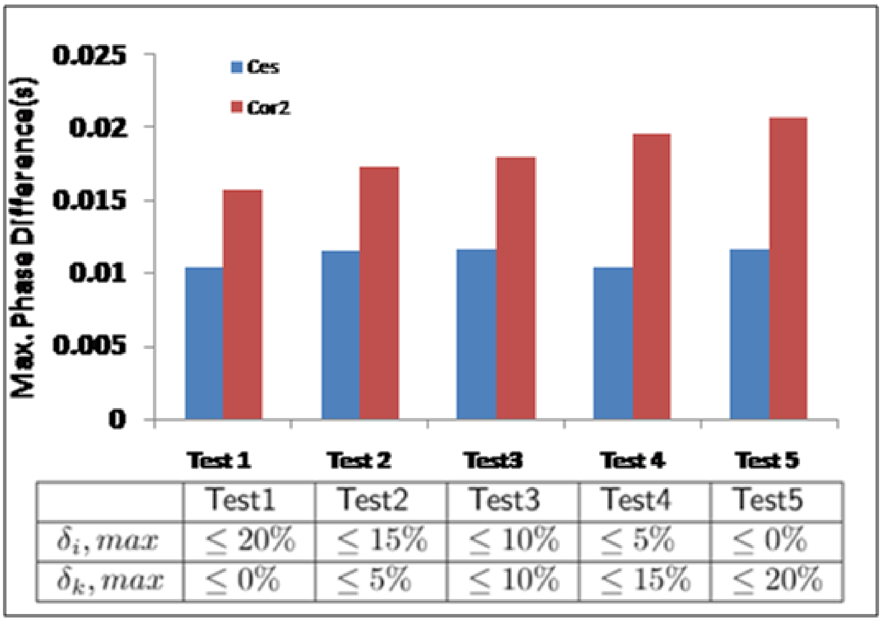
This section considers a dynamic environment with a highly dynamic communication graph and unpredictable changes in clock drift. In the first scenario, we set different values of the clock drift to test the edge cases of both C_{es} and $Corr^2$. In the second scenario, we run both algorithms under different mobile settings: the uncorrelated random walk, and vehicular mobility. Both scenarios reflect a more realistic nature of mobile sensor networks in which a sensor changes neighbors as it changes location, and the network's connectivity may dramatically change over time. A good algorithm should tolerate these uncertainties while maintaining a common notion of time among the participating nodes.

11.3.1 Changes in Clock Drift

In this experiment, we run different simulations using different values of δ_i and δ_k as in Table 11.2. The δ_i and δ_k are frequency varying variables, δ_i is applied once while δ_k is applied at each round. At the start of the simulation, each sensor uses a different value of δ_i; δ_k is changed at the start of each round. We test different cases to verify how each algorithm handle uncertainties of clock drift changes. Figure 11.5 shows the results of different test cases in terms of average phase difference in each case. We can observe that $Corr^2$ errors grow with the increase in clock drift varying variable δ_k unlike C_{es}, which stabilizes within an interval. The figure also shows that the C_{es} has smaller value of clock skew when both When δ_k and δ_i are higher i.e it can tolerate uncertainties in the clock drift with higher value of δ_k and δ_i as well. Therefore, the C_{es} is best suited in an environment with uncertainties in clock drift compared to the $Corr^2$.

	Test1	Test2	Test3	Test4	Test5
δ_i, max	$\leq 20\%$	$\leq 15\%$	$\leq 10\%$	$\leq 5\%$	$\leq 0\%$
δ_k, max	$\leq 0\%$	$\leq 5\%$	$\leq 10\%$	$\leq 15\%$	$\leq 20\%$

Table 11.2: Values of the clock drift under different test scenarios.



	Test1	Test2	Test3	Test4	Test5
δ_i, max	$\leq 20\%$	$\leq 15\%$	$\leq 10\%$	$\leq 5\%$	$\leq 0\%$
δ_k, max	$\leq 0\%$	$\leq 5\%$	$\leq 10\%$	$\leq 15\%$	$\leq 20\%$

Figure 11.5: The figure shows the average phase difference under different values of δ_i and δ_k. The error of $Corr^2$ increases with the increase in δ_k unlike C_{es}

11.3.2 Uncorrelated Random Walk

In this scenario, we simulate random walk movement using a simple four-direction uncorrelated random walk. The environment is similar to Section 11.2.2 only that we add node mobility. Each sensor moves in one of the four directions with a randomly chosen speed in the range $S \in [1, 1.14]m/s$ to reflect the realistic human mobility in an indoor environment. We observe that at some time instants, a sensor may disconnect from its neighbors or have fewer neighbors than in static settings. The result of the simulation is shown in Figure 11.6. The figure shows that C_{es} converges its phase difference to a smaller value and within an interval, but due to the changes in the communication graph, the frequency fluctuates in some rounds. The $Corr^2$ has more phase fluctuations as compared to C_{es} as its compensation technique places equal weight to the previous round estimates, unlike C_{es}, which implements a learning technique to decide on the appropriate weight of the previous and present round estimates for the next pulse prediction. In this regard, the C_{es} adapts to a network with dynamic topology due to mobility.

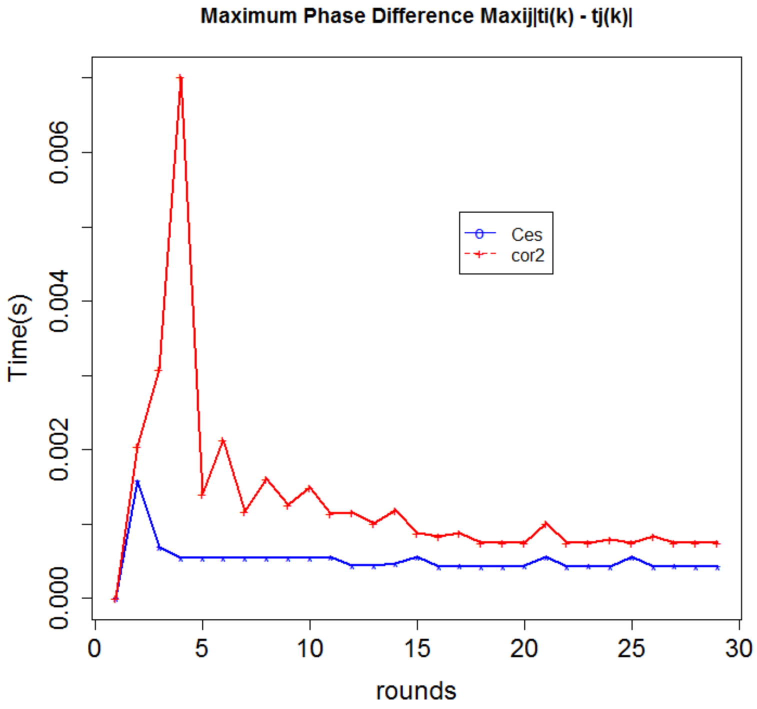


Figure 11.6: The figure shows the Maximum phase difference for the C_{es} and $Corr^2$in each round when nodes move using random walk model. Both algorithms fluctuate due to disconnection of nodes but the C_{es} has less fluctuation and smaller time compared to the $Corr^2$

11.3.3 Vehicular Mobility

In this scenario, we simulate a more realistic environment using a vehicular network. We used the Simulation of Urban MObility (SUMO) simulator for the traffic of an area in the French city of Les Ulis. SUMO is an open-source, highly portable, microscopic road traffic simulation package designed to handle large road networks Nishad and Pandey [2018]. The map of the area was downloaded from Open Street Map OpenStreetMap contributors [2019].

The first scenario considers a static vehicle network of 50 nodes randomly scattered with an average connectivity of 3 neighbors per vehicle as depicted in Figure 11.7. Each vehicle is assigned with a different sensing range of an average 300 m. A communication between a given vehicle and its neighbor is guaranteed only if a neighbor is within the distance $r_i \epsilon \mathbb{R}$, bounded within a range: r(1- ϵ) $\leq r_i \leq$ r(1+ ϵ). We set the time period $T_i(k)$ to R(1+δ_i+δ_k). δ_i and δ_k are frequency varying variables, δ_i is applied once when a vehicle initialized while δ_k is applied at each round. We set $\epsilon = 0.5$ and the number of *timewheel* slot to 2 for $corr^2$. Under this setting, we study the *pulse time generation* to understand further the convergence of different pulses broadcasted in each round. Pulses with different colors are broadcasted by different vehicles. Figure 11.8 presents

Parameter	Value
Number of vehicles	50
Sensing Range	300 m
Area	1000 X 1200 m^2
Sensing range Variation $\in$	0.5
Maximum slot	2
Average connectivity	3
Maximum vehicle speed	14 m/s
Maximum vehicle acceleration	2.6 m/s^2
Vehicle frequency variation $\lvert\delta_i\rvert$	$\leq$ 20%
Round frequency variation $\lvert\delta_k\rvert$	$\leq$ 1%

Table 11.3: Summary of Vehicular network simulation parameters

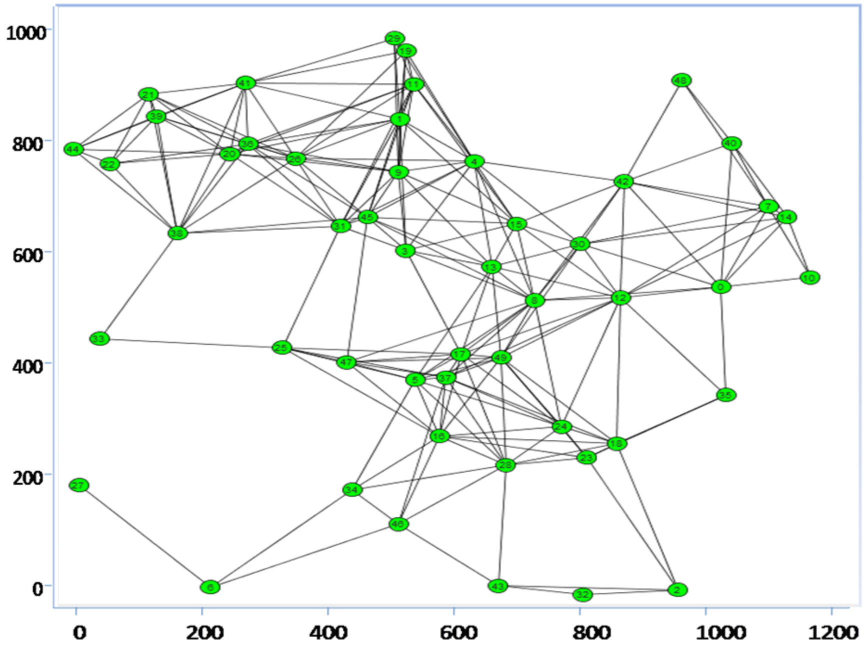


Figure 11.7: Vehicle Network with 50 vehicles

pulses of vehicles if only offset compensation is applied. Observe that, though pulses are trapped within one interval, the interval did not decrease with an increase in rounds. In Figure 11.9, $Corr^2$ is able to converge the pulses toward the middle of the simulation after many rounds of fluctuations as shown in Figure 11.4. However, in Figure 11.10, C_{es} pulses are successfully trapped in one interval throughout the simulation after just a few rounds fluctuations. The discovery phase in our technique helps in choosing an optimal adaptive value of λ that enhances prediction accuracy of C_{es}. This shows the ability of C_{es} to tolerate short-term fluctuations of clock drift and therefore make it a suitable algorithm in a dynamic network with possibly unpredictable node clock drift changes.

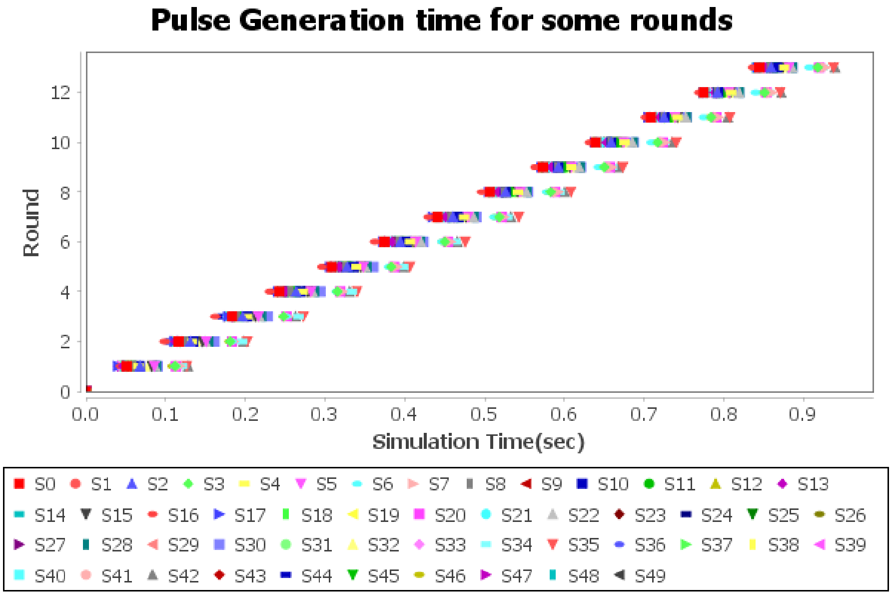


Figure 11.8: Pulse time for each round using $Corr^1$ only. Pulses with different colors belong to different vehicles. The pulses diverged throughout the simulation.

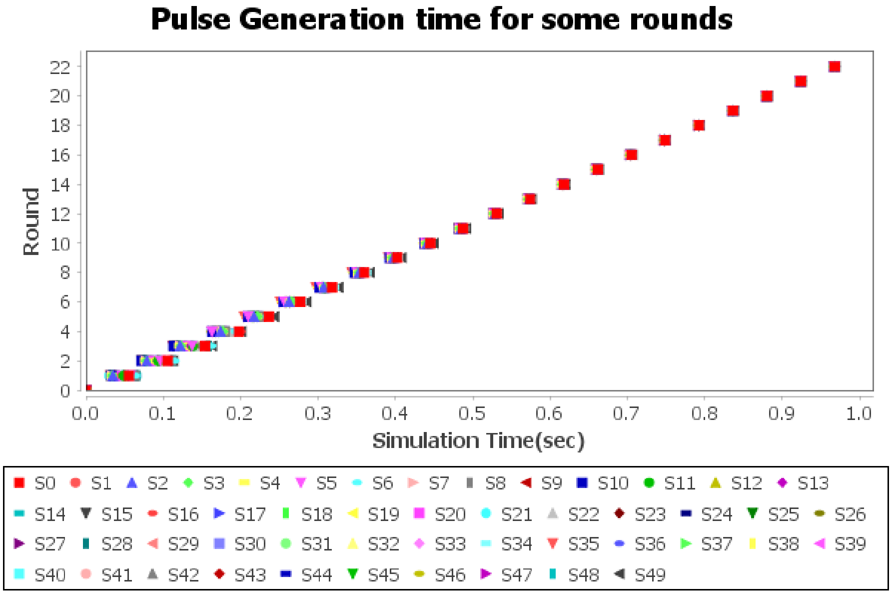


Figure 11.9: Pulse Time for each round using $Corr^2$. The pulses diverged initially but converged to a common value after some rounds.

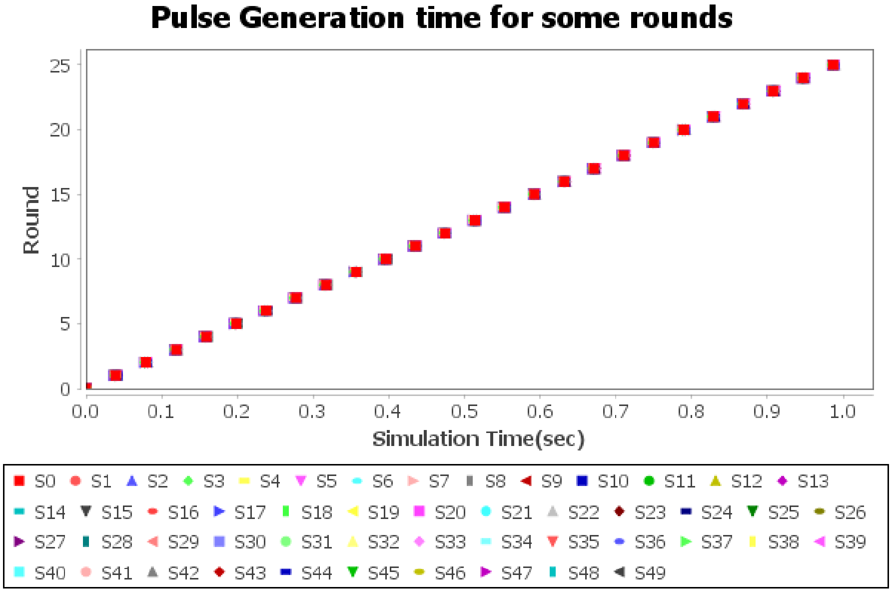


Figure 11.10: Pulse Time for each round using C_{es}. The pulses converged to a common value after few rounds.

In the second scenario, we consider vehicles moving with a maximum speed of 14 m/s and the acceleration of 2.6 m/s^2. All vehicles change routes according to the route file generated using DUAROUTER scripts in SUMO. A vehicle decelerates when turning back or the traffic light is red. At the end of the simulation, we generate a Floating Car Data (FCD) trace file with attributes like the vehicle id, speed, and position used for the evaluation. In Figure 11.11, we compare $corr^1$, $corr^2$ and C_{es} algorithms in terms of Maximum Phase Difference in each round. Observe that $corr^2$ converges to a stable common frequency after almost 16 rounds, indicating its inability to handle random uncertainty of clock drift in this type of setting. $corr^1$ is able to minimize round fluctuations but with a high value of phase difference. The C_{es} stabilizes to a common stable frequency with small deviation throughout the simulation with a small value of phase difference and tolerates inertia's effect even after a long run. This property account for low skew of C_{es} as compared to $corr^1$ or $corr^2$ in a more dynamic settings. Moreover, toward the end of the simulation, the convergence of the $corr^2$ to a smaller time value is a worthy consideration. Better performance is achieved after many rounds

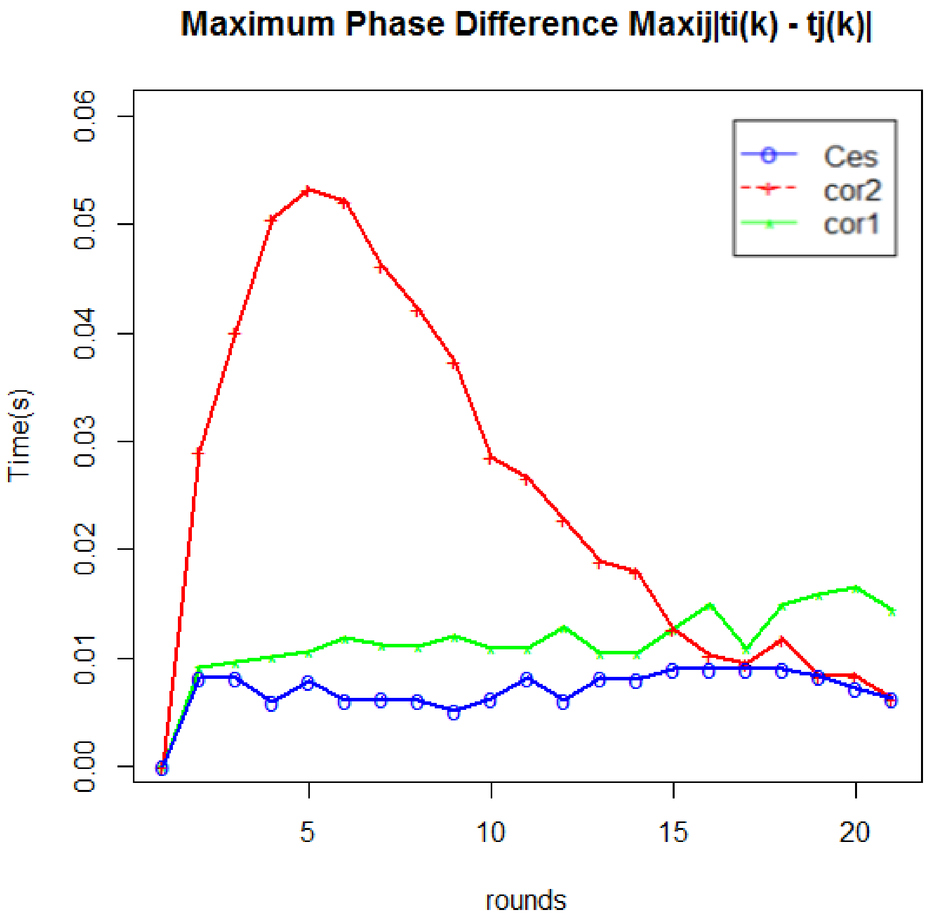


Figure 11.11: The figure shows a Comparison of Maximum Phase Difference for Corr1, Corr2 and C_{es} for a vehicular network. The C_{es} has stabilized to a smaller time value, but at some rounds, it experiences short fluctuations due to vehicle disconnections or change in ambient conditions. The $corr^2$ only stabilizes toward the end of the simulation.

when the clock skew is significantly reduced. The most interesting feature of the C_{es} is quick synchronization, the $corr^2$ synchronizes slowly until the skew is low with a minimal phase difference. Figure 11.12 shows the round duration for $corr^1$, $corr^2$ and C_{es}. Observe that C_{es} maintains a stable round duration after few rounds of fluctuation, unlike $corr^2$ which takes many rounds before stabilizing to common round duration. The instability of the $corr^1$ at some intervals is not surprising because $corr^1$ has only offset correction, making it vulnerable to drift changes. After 15 rounds, we change vehicles speed and the round-varying variable δ_k, thereby increasing the chances of a topology

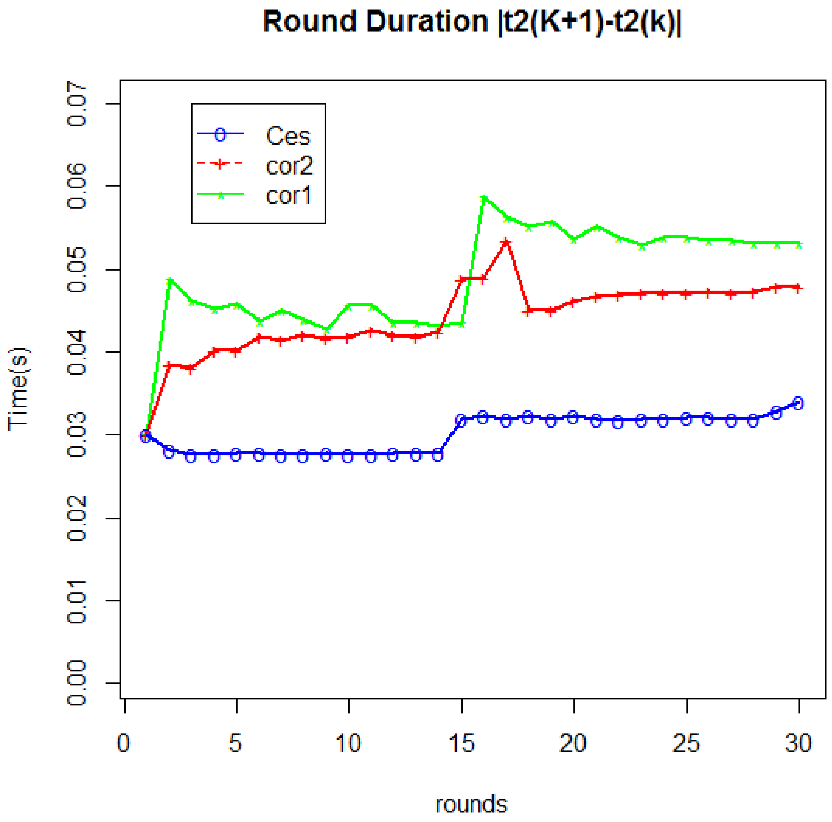


Figure 11.12: The figure shows a comparison of Round Duration for moving vehicles using Ces", Corr2 and Corr1. It shows how each algorithm tolerates sudden changes in the communication graph. The C_{es} maintains a common time value for all rounds even when the network is shaken by changing vehicles speed and the round-varying variable δ_k.

change and frequency variations. Even after these modifications, C_{es} is able to maintain common round duration after small deviation better than $corr^1$ and $corr^2$.

11.3.4 Impacts of the exponential decay factor λ on the performance of C_{es}

We investigate the impact of different values of the exponential decay factor λ on the performance of the proposed algorithm using a static vehicle setting. Figure 11.13 shows the maximum phase difference of the proposed algorithm using different values of an

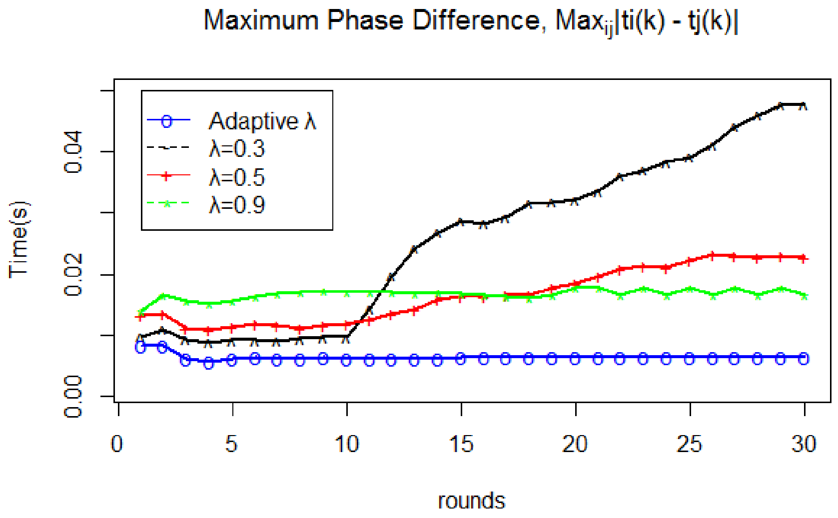


Figure 11.13: Pulse time for each round using corr1

exponential decay factor (λ).

We can observe that with a small value of λ, the previous estimate has more impacts on the predicted clock value, causing a negative impact on the clock-skew as in the case of a $\lambda = 0.3$; the time for the skew increases as the rounds increase. On the other hand, a higher value of λ influences recent estimates which helps to reduce the clock skew and keep phase difference at a steady-state; this can be seen with $\lambda = 0.9$. So a higher value of a λ gives a moderately stabilized skew but with random fluctuations of phases in each round and a higher value time difference. We need to keep the time for the phase difference minimal without fluctuations. Our adaptive approach solves the problem as shown in the figure. The adaptive λ maintains a smaller and stable common value for the phase difference after just a few rounds because it uses the best λ value in the prediction phase at each round to estimate the next pulse time, hence improves the prediction accuracy of the algorithm.

11.4 Conclusion

In this chapter, we validate the proposed algorithm under different environmental settings and investigate its performance. The proposed algorithm is compared with the previous work of Han [2018] and Fugger et al. [2015]. The results have shown that the proposed algorithm has reduced the clock skew compared to the previous techniques. The proposed algorithm has been shown to tolerate higher uncertainties in clock drift and handles unexpected topological changes in a dynamic environment by using a sensor's local history to predicts the next round pulse. The algorithm adapts to both static and dynamic settings, making it the best candidate for an efficient data fusion algorithm in both cases.

Chapter 12

Conclusion and Future Work

12.1 Introduction

In this chapter, we summarized the contributions of the Second part of the thesis. We suggest further works that could extend our findings and further validate our claims.

12.2 Conclusion and Contributions

We proposed a pulse-coupled clock synchronization algorithm, which uses a learning technique to improve clock synchronization accuracy. The proposed algorithm places more weight on the recent estimates by exploiting the heuristics in the newly proposed method to conduct the offset and drift compensation. The learning method adopts a dynamic adaptive approach for choosing an optimal value of the weight factor λ; the learning phase decides on the optimal value and forwards it to the prediction phase for the subsequent prediction. The proposed algorithm has been compared with the previous work of Han [2018], and Fugger et al. [2015] under static and mobile settings, and the results have shown that it has reduced the clock skew, especially in a dynamic environment with high uncertainty in clock drift and unexpected topological changes like the vehicular network. The robustness of the proposed algorithm in realistic settings like the vehicular network and the indoor random walk scenario has shown that pulse-coupled synchronization techniques adapt to challenging environments; hence the need for more study on the pulse-coupled synchronization approaches.

We intend to evaluate the proposed algorithm using a real-world sensor network. The intended sensor test-bed for the evaluation would simulate static and mobile scenarios to extensively investigate the impact of each setting on the performance of our algorithm. We also intend to experiment with a large-scale sensor network to pave the way for implementing our algorithm in a smart campus sensor network with a large number of static and mobile nodes. We would select appropriate communication hardware, followed by a number of simulated scenarios.

12.3 Publications

1. Bakura, S.A, Lambert, A., and Nowak, T. (2020). *Clock Synchronization with Exponential Smoothing for Dynamic Networks.* 54th Annual Conference on Information Sciences and Systems (CISS), 18-20 March, 2020. Princeton University, USA.

2. Bakura, S.A, Lambert, A., and Nowak, T. (2020). *Clock Synchronization with Adaptive Weight Factor for Mobile Networks.* 28th Mediterranean Conference on Control and Automation (MED'2020), 16-18 September, 2020. Saint Raphael, FRANCE

Chapter 13

Conclusion for the Thesis

13.1 General Conclusion

This thesis deals with two problems in the wireless network: Mobility modeling and clock synchronization in the wireless sensor network. In the first part of the thesis, We conduct an in-depth analysis of the human mobility features. Consequently, we discover a new feature characterized by temporal dynamic movement clusters. Motivated by the finding, we develop a synthetic mobility model, used as a tool for the design and evaluation of wireless network protocols. The proposed model is evaluated using the real traces Rhee et al. [2009]; Scott et al. [2009]. It shows matching characteristics in terms of the distribution of the inter-cluster travel distance, intra-cluster travel distance, intra-cluster angle of movement, pause time, inter-contact time, and contact time with that of the studied real traces. In the second part of the thesis, we proposed a distributed pulse-coupled clock synchronization, improving the work by Han [2018]. The proposed algorithm implements a learning technique for offset and drift compensation. Additionally, we proposed an adaptive method for selecting the optimal value of a weight factor λ for efficient prediction. The simulation results have shown that the proposed algorithm has significantly reduced clock skew compared to the previous work in Han [2018] and Fugger et al. [2015]. Our twofold approach has dual benefits: firstly, it developed a realistic tool for the design and performance evaluation of networks with intermittent connections. Secondly, it designs an algorithm that ensures a common notion of time among WSNs node for an efficient implementation of data fusion algorithm, energy management, and real-time safety and infotainment applications.

13.2 Contributions

In this thesis, we achieved the following objectives:

- In Chapter 4, we conducted an in-depth analysis of the spatial, temporal, and connectivity features of human mobility and uncovered a new feature characterized by temporal dynamic movement clusters. The synthetic mobility models can exploit this feature to better predict the next location, stay location, and the opportunistic connection of the mobile nodes.

- We explain the details of the proposed synthetic mobility model that integrate the observed features and re-produce matching characteristics with the real traces Rhee et al. [2009]; Scott et al. [2009] in Chapter 5. An evaluation of the performance of routing protocols was conducted in Chapter 6.

- Chapter 10 proposed a new clock synchronization algorithm that uses a learning technique to predict round pulses. The algorithm performs the offset and drift compensation to ensure a common notion of time among the network nodes and makes each node's clock run at the same frequency with a negligible drift. The algorithm has been adopted in static and dynamic settings.

- The simulation results in Chapter 11 has shown that the proposed technique has reduced the clock skew compared to the previous works in Fugger et al. [2015] and Han [2018], especially in a dynamic setting with drifting clocks and topology changes. The algorithm is able to converge the pulses of the moving vehicles in a vehicular network with better accuracy in the presence of unpredictable changes in the clock drift.

13.3 Future Work

This thesis has made limited contributions regarding the characterization of human mobility features, design of mobility model for network protocols evaluation, and development of an algorithm for efficient data fusion algorithm, real-time safety applications, and resource management protocols in WSNs. There are still many opportunities to improve this work through

13.3.1 Human Mobility Prediction Framework

A good understanding of the human mobility features is an excellent tool for an accurate prediction framework. We intend to extend the proposed model by designing an efficient predicting framework. The new framework would exploit the existing and new uncovered features of human mobility to predict the user's next movement location, stay duration, and possible contact. The framework would be an excellent tool for transport agencies and network designers.

13.3.2 Effects of Obstacles on the Signal Propagation

Our obstacle model is more of trajectory estimation and collision avoidance. Little have been done on the effects of obstacle on the signal propagation; we intend to extensively investigate the effects of obstacles (i.e., the high-rise buildings, long tunnels, tree canopies, and etc.) on the signal propagation.

13.3.3 Experiment on Appropriate Test-bed

The proposed clock synchronization algorithm has been evaluated using simulation; though the result is encouraging, a more realistic evaluation environment is needed. We intend to evaluate the proposed algorithm using a real-world sensor network. The intended sensor test-bed for the evaluation would simulate static and mobile scenarios to extensively investigate the impact of each scene on the performance of our algorithm.

13.3.4 Large-scale Sensor Network

Our use case environment (i.e., smart-campus) is characterized by a large number of static and mobile sensors. In this regard, we intend to experiment with a large-scale sensor network. We would select appropriate communication hardware with different scenarios to simulate.

13.4 Publications

13.4.1 Published Papers

1. Bakura, S.A, Lambert, A., and Nowak, T. (2020). *Clock Synchronization with Exponential Smoothing for Dynamic Networks.* 54th Annual Conference on Information Sciences and Systems (CISS), 18-20 March, 2020. Princeton University, USA.

2. Bakura, S.A, Lambert, A., and Nowak, T. (2020). *An Obstacle-Aware Mobility Model for Campus Delay Tolerant Networks.* 2020 Zooming Innovation in Consumer Technologies Conference, 26-27 May, 2020. Novi Sad, Serbia.

3. Bakura, S.A, Lambert, A., and Nowak, T. (2020). *Clock Synchronization with Adaptive Weight Factor for Mobile Networks.* 28th Mediterranean Conference on

Control and Automation (MED’2020), 16-18 September, 2020. Saint Raphael, FRANCE

13.4.2 Submitted Paper

paper titled: EPOM- Escape Path Obstacle-based Mobility Model for Campus Delay Tolerant Network". Journal name: Journal for Advance Transport. Date Submitted: 21st June, 2021.

Appendix A

Proof of Lemma 1

According to Equation (10.14)

$$\sum_{c=1}^{\tau_j^i(k)-1} \mu_i(c) \leq t_j(k) \leq \sum_{c=1}^{\tau_j^i(k)} \mu_i(c) \tag{A.1}$$

and

$$\sum_{c=1}^{\tau_i(k)-1} \mu_i(c) \leq t_i(k) \leq \sum_{c=1}^{\tau_i(k)} \mu_i(c) \tag{A.2}$$

Combining (A.1) and (A.2) we have

$$\sum_{c=1}^{\tau_j^i(k)-1} \mu_i(c) - \sum_{c=1}^{\tau_i(k)} \mu_i(c) \leq t_j(k) - t_i(k) \leq \sum_{c=1}^{\tau_j^i(k)} \mu_i(c) - \sum_{c=1}^{\tau_i(k)-1} \mu_i(c) \tag{A.3}$$

if $\tau_j^i(k)$ -1 $\geq \tau_i(k)$, the left hand side of Equation (A.3) can be reformulated as

$$\begin{aligned}\sum_{c=1}^{\tau_j^i(k)-1} \mu_i(c) - \sum_{c=1}^{\tau_i(k)} \mu_i(c) &= \sum_{c=\tau_i(k)+1}^{\tau_j^i(k)-1} \mu_i(c) \\ &\geq (\tau_j^i(k) - \tau_i(k) - 1)(1-\varrho)\mu\end{aligned} \tag{A.4}$$

if $\tau_j^i(k)$ -1 $\leq \tau_i(k)$, the left hand side of (A.3) can be derive with similar derivation,

$$\begin{aligned}\sum_{c=1}^{\tau_j^i(k)-1} \mu_i(c) - \sum_{c=1}^{\tau_i(k)} \mu_i(c) &= -\sum_{c=\tau_j^i(k)}^{\tau_i(k)} \mu_i(c) \\ &\geq -(\tau_i(k) - \tau_j^i(k) + 1)(1-\varrho)\mu \\ &= (\tau_j^i(k) - \tau_i(k) - 1)(1-\varrho)\mu\end{aligned} \tag{A.5}$$

Combining Equation (A.4) and (A.5), we have

$$\sum_{c=1}^{\tau_j^i(k)-1} \mu_i(c) - \sum_{c=1}^{\tau_i(k)} \mu_i(c) \geq (\tau_j^i(k) - \tau_i(k) - 1)(1-\varrho)\mu \tag{A.6}$$

with similar derivation as Equation (A.6), the right hand side of Equation (A.3) is

$$\sum_{c=1}^{\tau_j^i(k)} \mu_i(c) - \sum_{c=1}^{\tau_i(k)-1} \mu_i(c) \leq (\tau_j^i(k) - \tau_i(k) + 1)(1+\varrho)\mu \tag{A.7}$$

Combining Equation (A.6), Equation (A.7) with Equation (A.3), we have

$$(\tau_j^i(k) - \tau_i(k) - 1)(1-\varrho)\mu \leq t_j(k) - t_i(k) \leq (\tau_j^i(k) - \tau_i(k) + 1)(1+\varrho)\mu \tag{A.8}$$

From Equation (A.8), we get

$$t_j(k) - t_i(k) = (\tau_j^i(k) - \tau_i(k) - \alpha_j^i(k))/\beta_j^i(k) \tag{A.9}$$

With $\beta_j^i \in [\ \mu(1\text{-}\ \varrho), \mu(1\text{+}\ \varrho)]$ and α_j^i(k) $\in$ [-1, 1].

$$\tau_i^j(k) - \tau_i(k) = (t_j(k) - t_i(k)))/\beta_i^j(k) + \alpha_i^j(k), \tag{A.10}$$

Hence Proof

□

Appendix B

proof of Lemma 2

The timing of the pulse from sensor j to i is received as discrete time event. Using Equation (A.1), we get:

$$\sum_{c=1}^{\tau_i(k+1)-1} \mu_i(c) \leq t_i(k+1) \leq \sum_{c=1}^{\tau_i(k+1)} \mu_i(c) \tag{B.1}$$

and

$$\sum_{c=1}^{\tau_i(k)-1} \mu_i(c) \leq t_i(k) \leq \sum_{c=1}^{\tau_i(k)} \mu_i(c) \tag{B.2}$$

Combining Equation (B.1) and Equation (B.2) we have

$$\sum_{c=1}^{\tau_i(k+1)-1} \mu_i(c) - \sum_{c=1}^{\tau_i(k)} \mu_i(c) \leq t_i(k+1) - t_i(k) \leq \sum_{c=1}^{\tau_i(k+1)} \mu_i(c) - \sum_{c=1}^{\tau_i(k)-1} \mu_i(c) \tag{B.3}$$

if τ_i(k+1) $\geq$ τ_i(k)+1, the left hand side of Equation (B.3) is

$$\begin{aligned} \sum_{c=1}^{\tau_i(k+1)-1} \mu_i(c) - \sum_{c=1}^{\tau_i(k)} \mu_i(c) &= \sum_{c=\tau_i(k)+1}^{\tau_i(k+1)-1} \mu_i(c) \\ &\geq (\tau_i(k+1) - \tau_i(k) - 1)(1-\varrho)\mu \end{aligned} \tag{B.4}$$

if τ_i(k+1) -1 $\leq$ τ_i(k), the left hand side of Equation (B.3) is

$$\begin{aligned}\sum_{c=1}^{\tau_i(k+1)-1}\mu_i(c)-\sum_{c=1}^{\tau_i(k)}\mu_i(c) &= -\sum_{c=\tau_i(k+1)}^{\tau_i(k)}\mu_i(c)\\ &\geq -(\tau_i(k)-\tau_i(k+1)+1)(1-\varrho)\mu\\ &\geq (\tau_i(k+1)-\tau_i(k)-1)(1-\varrho)\mu\end{aligned} \tag{B.5}$$

With Equation (B.4) and Equation (B.5) we have

$$\sum_{c=1}^{\tau_i(k+1)-1}\mu_i(c)-\sum_{c=1}^{\tau_i(k)}\mu_i(c) \geq (\tau_i(k+1)-\tau_i(k)-1)(1-\varrho)\mu \tag{B.6}$$

With Similar derivation, the right hand side of Equation (B.3) is

$$\sum_{c=1}^{\tau_i(k+1)}\mu_i(c)-\sum_{c=1}^{\tau_i(k)-1}\mu_i(c) \geq (\tau_i(k+1)-\tau_i(k)+1)(1+\varrho)\mu \tag{B.7}$$

Combining Equation (B.6) and Equation (B.7) with Equation (B.3), we get.

$$(\tau_i(k+1)-\tau_i(k)-1)(1-\varrho)\mu \leq t_i(k+1)-t_i(k) \leq (\tau_i(k+1)-\tau_i(k)+1)(1+\varrho)\mu \tag{B.8}$$

From Equation (B.8), we get

$$t_i(k+1)-t_i(k) = (\tau_i(k+1)-\tau_i(k)-\alpha_i(k))\beta_i(k) \tag{B.9}$$

With $\beta_i \in$ [μ(1- ϱ), μ(1+ ϱ)] and α_i(k) $\in$ [-1, 1].

hence proof.

□

Appendix C

Derivation for Equation (10.7)

let $\mathrm{C}'_{\mathrm{es_i}}(\mathrm{k})$ be the value of exponentially weighted average for sensor i at round k, where $0< \lambda <1$. λ is called the weighting factor. We need a starting value for $\mathrm{C}'_{\mathrm{es_i}}(\mathrm{k}-1)$ which can be

- a specified target value C, $\mathrm{C}'_{\mathrm{es_i}}(\mathrm{k}-1) = 0$. or

- The first estimate of $\mathrm{C}_{\mathrm{avg_i}}(\mathrm{k})$, $\mathrm{C}'_{\mathrm{es_i}}(\mathrm{k}-1) = \mathrm{C}_{\mathrm{avg_i}}(\mathrm{k})$

Observe that $\mathrm{C}'_{\mathrm{es_i}}(\mathrm{k})$ is a weighted average of all previous estimates, for example using Equation (10.4), if

K = 1:

$$\begin{aligned} \mathrm{C}'_{\mathrm{es_i}}(1) &= \lambda \mathrm{C}_{\mathrm{avg_i}}(1) + (1-\lambda)(\mathrm{C}'_{\mathrm{es_i}}(0)) \\ &= \lambda \mathrm{C}_{\mathrm{avg_i}}(\mathrm{k}) + (1-\lambda)\mathrm{C}'_{\mathrm{es_i}}(\mathrm{k}-1) \end{aligned} \tag{C.1}$$

K = 2:

$$
\begin{aligned}
\mathrm{C}'_{\mathrm{es_i}}(2) &= \lambda \mathrm{C}_{\mathrm{avg_i}}(2) + (1-\lambda)(\mathrm{C}'_{\mathrm{es_i}}(1)) \\
&= \lambda(1-\lambda)^{(0)}\mathrm{C}_{\mathrm{avg_i}}(2) + \lambda(1-\lambda)^{(1)}\mathrm{C}_{\mathrm{avg_i}}(1) + (1-\lambda)^{(2)}\mathrm{C}'_{\mathrm{es_i}}(0) \\
&= \lambda(1-\lambda)^{(k-2)}\mathrm{C}_{\mathrm{avg_i}}(\mathrm{k}) + \lambda(1-\lambda)^{(k-1)}\mathrm{C}_{\mathrm{avg_i}}(\mathrm{k}-1) + (1-\lambda)^{(k)}\mathrm{C}'_{\mathrm{es_i}}(\mathrm{k}-2)
\end{aligned}
$$

K = 3:

$$
\begin{aligned}
\mathrm{C}'_{\mathrm{es_i}}(3) &= \lambda \mathrm{C}'_{\mathrm{es_i}}(3) + (1-\lambda)(\mathrm{C}'_{\mathrm{es_i}}(2)) \\
&= \lambda(1-\lambda)^{(0)}\mathrm{C}'_{\mathrm{es_i}}(3) + \lambda(1-\lambda)^{(1)}\mathrm{C}_{\mathrm{avg_i}}(2) + \lambda(1-\lambda)^{(2)}\mathrm{C}_{\mathrm{avg_i}}(1) + (1-\lambda)^{(3)}\mathrm{C}'_{\mathrm{es_i}}(0)) \\
&= \lambda(1-\lambda)^{(k-3)}\mathrm{C}_{\mathrm{avg_i}}(\mathrm{k}) + \lambda(1-\lambda)^{(k-2)}\mathrm{C}_{\mathrm{avg_i}}(\mathrm{k}-1) + \lambda(1-\lambda)^{(k-1)}\mathrm{C}_{\mathrm{avg_i}}(\mathrm{k}-2) + \\
&(1-\lambda)^{(k)}\mathrm{C}'_{\mathrm{es_i}}(\mathrm{k}-3)
\end{aligned}
\tag{C.2}
$$

We can recursively write the general form of $\mathrm{C}'_{\mathrm{es_i}}(\mathrm{k})$ for 0< λ <1 as

$$
\mathrm{C}'_{\mathrm{es_i}}(\mathrm{k}) = \lambda \sum_{n=0}^{k-1} \left((1-\lambda)^n \mathrm{C}_{\mathrm{avg_i}}(\mathrm{k}-\mathrm{n}) \right) + (1-\lambda)^k \mathrm{C}'_{\mathrm{es_i}}(\mathrm{k}-\mathrm{k}) + (1-\lambda)^k \mathrm{C}'_{\mathrm{es_i}}(\mathrm{k}-\mathrm{k}) \tag{C.3}
$$

□

Appendix D

proof of Equation (10.7) by induction

1. Proposition:

$$\begin{aligned}\tau_i(k+1) &= \Big(\tau_i(k) + \mathrm{C}'_{\mathrm{es_i}}(\mathrm{k})\Big) + R \\ &= \tau_i(k) + \lambda \sum_{n=0}^{k-1} \left((1-\lambda)^n \left(\frac{1}{|In_i(k+1-n)|} \sum_{j \in In_i(k+1-n)} (\tau_j(k-n) - \tau_i(k-n)) \right) \right) + \\ &(1-\lambda)^k \mathrm{C}'_{\mathrm{es_i}}(\mathrm{k}-\mathrm{k}) + R\end{aligned} \tag{D.1}$$

for all k ≥ 0

2. Base case:

Let k=1.

L.H.S. $\tau_i(k+1) = \tau_i(2)$

$\tau_i(2)$=($\tau_i(1)$+$\mathrm{C}'_{\mathrm{es_i}}(1)$) +R. From Equation (D.1)

R.H.S. substituting 1 for k in Equation (D.1)

$$= \tau_i(1) + \lambda \sum_{n=0}^{1-1} \left((1-\lambda)^0 \left(\frac{1}{|In_i(1+1-0)|} \sum_{j \in In_i(1+1-0)} (\tau_j(1-0) - \tau_i(1-0)) \right) \right) +$$

$(1-\lambda)^1 \mathrm{C}'_{\mathrm{es_i}}(1-1) + R$

$$= \tau_i(1) + \lambda \sum_{n=0}^{0} \left((1-\lambda)^0 \left(\frac{1}{|In_i(1+1-0)|} \sum_{j \in In_i(1+1-0)} (\tau_j(1-0) - \tau_i(1-0)) \right) \right) +$$

$(1-\lambda)^1 \mathrm{C}'_{\mathrm{es_i}}(1-1) + R$

$$= \tau_i(1) + \lambda \left(1. \frac{1}{|In_i(2)|} \sum_{j \in In_i(2)} (\tau_j(1) - \tau_i(1)) \right) + (1-\lambda) \mathrm{C}'_{\mathrm{es_i}}(0) + R$$

$$= \tau_i(1) + \left(\lambda \mathrm{C}_{\mathrm{avg_i}}(1) + (1-\lambda) \mathrm{C}'_{\mathrm{es_i}}(0) \right) + R$$

$$= (\tau_i(1) + \mathrm{C}'_{\mathrm{es_i}}(1)) + R \tag{D.2}$$

Note that from Equation (10.6), the expression ($\frac{1}{|In_i(2)|} \sum_{j \in In_i(2)}$ ($\tau_j(1)$ - $\tau_i(1)$)) is equivelent to $\mathrm{C}'_{\mathrm{es_i}}$(k) if k = 1. Similarly from Equation (C.1) the expression (λ $\mathrm{C}_{\mathrm{avg_i}}(1)$ + (1- λ)$\mathrm{C}'_{\mathrm{es_i}}(0)$) is equivelent to $\mathrm{C}'_{\mathrm{es_i}}(1)$ when k = 1. Since L.H.S. and R.H.S. are the same, then base case holds, therefore the formula is true for k=1.

3. Assumption:

Since it's true for k=1, let's sssume that it's true for all numbers up to M.

Assume k = M :

$$\tau_i(M+1) = \left(\tau_i(M) + \mathrm{C}'_{\mathrm{es_i}}(\mathrm{M}) \right) + R$$

Substituting M for K in Equation (D.1) gives

$$= \tau_i(M) + \lambda \sum_{n=0}^{M-1} \left((1-\lambda)^n \left(\frac{1}{|In_i(M+1-n)|} \sum_{j \in In_i(M+1-n)} (\tau_j(M-n) - \tau_i(M-n)) \right) \right) + (1-\lambda)^M \mathrm{C}'_{\mathrm{es_i}}(\mathrm{M}-\mathrm{M}) + R$$

$$
\begin{aligned}
&= \tau_i(M) + \lambda\Big((1-\lambda)^0\Big(\frac{1}{|In_i(M+1-0)|}\sum_{j\in In_i(M+1-0)}(\tau_j(M-0)-\tau_i(M-0))\Big)+\\
&\lambda\Big((1-\lambda)^{M-1}\Big(\frac{1}{|In_i((M+1)-(M-1))|}\sum_{j\in In_i((M+1)-(M-1))}(\tau_j(M-(M-1))-\tau_i(M-(M-1)))\Big)+\\
&(1-\lambda)^M \mathrm{C}'_{\mathrm{es_i}}(\mathrm{M}-\mathrm{M})+R\\
&= \tau_i(M)+\lambda(1-\lambda)^0\Big(\frac{1}{|In_i(M+1)|}\sum_{j\in In_i(M+1)}(\tau_j(M)-\tau_i(M))\Big)+\lambda(1-\lambda)^{M-1}\Big(\frac{1}{|In_i(2)|}\sum_{j\in In_i(2)}(\tau_j(1)-\tau_i(1))\Big)+\\
&(1-\lambda)^M \mathrm{C}'_{\mathrm{es_i}}(0)+R\\
&= \tau_i(M)+\lambda(1-\lambda)^0\mathrm{C}_{\mathrm{avg_i}}(\mathrm{M})+\lambda(1-\lambda)^{M-1}\mathrm{C}_{\mathrm{avg_i}}(1)+(1-\lambda)^M\mathrm{C}'_{\mathrm{es_i}}(0)+R
\end{aligned}
\tag{D.3}
$$

4. Induction step:

if it's true for k = M, then it should be true for k= M+1.

We need to show it also holds true for M+1

$$
\tau_i((M+1)+1) = \Big(\tau_i(M+1)+\mathrm{C}'_{\mathrm{es_i}}(\mathrm{M}+1)\Big)+R \tag{D.4}
$$

Substituting M+1 for K in Equation (D.1) to show that, the result is the same with Equation (D.4)

$$
\begin{aligned}
&= \tau_i(M{+}1)+\lambda\sum_{n=0}^{(M+1)-1}\Bigg((1{-}\lambda)^n\Big(\frac{1}{|In_i((M+1)+1-n)|}\sum_{j\in In_i((M+1)+1-n)}(\tau_j((M{+}1){-}n){-}\tau_i((M{+}1){-}n))\Big)\Bigg)+\\
&(1-\lambda)^{(M+1)}\mathrm{C}'_{\mathrm{es_i}}(\mathrm{M}+1)-(M+1))+R
\end{aligned}
\tag{D.5}
$$

$$
\begin{gathered}
=\tau_i(M+1)+\lambda\Big((1-\lambda)^0\Big(\frac{1}{|In_i((M+1)+1-0)|}\sum_{j\in In_i((M+1)+1-0)}(\tau_j((M+1)-0)-\tau_i((M+1)-0))\Big)+\\
\lambda\Big((1-\lambda)^{M-1}\Big(\frac{1}{|In_i((M+1)+1-(M-1))|}\sum_{j\in In_i((M+1)+1-(M-1))}(\tau_j((M+1)-(M-1))-\tau_i((M+1)-(M-1)))\Big)+\\
\lambda\Big((1-\lambda)^{M}\Big(\frac{1}{|In_i((M+1)+1-M))|}\sum_{j\in In_i((M+1)+1-M))}(\tau_j((M+1)-M)-\tau_i((M+1)-M)))\Big)+\\
(1-\lambda)^{M+1}\mathrm{C}'_{\mathrm{es_i}}(\mathrm{M}+1)-(M+1))+R\\
=\tau_i(M+1)+\lambda(1-\lambda)^0\Big(\frac{1}{|In_i(M+2)|}\sum_{j\in In_i(M+2)}(\tau_j(M+1)-\tau_i(M+1))\Big)+\lambda(1-\lambda)^{M-1}\Big(\frac{1}{|In_i(3)|}\sum_{j\in In_i(3)}(\tau_j(2)-\tau_i(2))\Big)+\\
\lambda(1-\lambda)^{M}\Big(\frac{1}{|In_i(2)|}\sum_{j\in In_i(2)}(\tau_j(1)-\tau_i(1))\Big)+(1-\lambda)^{M+1}\mathrm{C}'_{\mathrm{es_i}}(0)+R\\
=\tau_i(M+1)+\lambda(1-\lambda)^0\mathrm{C}_{\mathrm{avg_i}}(\mathrm{M}+1)+\lambda(1-\lambda)^{M-1}\mathrm{C}_{\mathrm{avg_i}}(2)+\lambda(1-\lambda)^{M}\mathrm{C}_{\mathrm{avg_i}}(1)+(1-\lambda)^{M+1}\mathrm{C}'_{\mathrm{es_i}}(0)+R
\end{gathered}
\tag{D.6}
$$

Considering Equation (C.2), result of Equation (D.6) is a recursive form of $\mathrm{C}'_{\mathrm{es_i}}(\mathrm{k})$ with k = M+1. Therefore Equation (D.6) gives

$$=\Big(\tau_i(M+1)+\mathrm{C}'_{\mathrm{es_i}}(\mathrm{M}+1)\Big)+R \tag{D.7}$$

Note that, the derived result of Equation (D.6) is the same with Equation (D.4)

5. Conclusion:

Since it works for k=1, and if it works for k = M then it also works for k = M+1, the the formula works for all k∈ℝ

□

Appendix E

proof of Lemma 3

We now show that at any round k each row of $\boldsymbol{A}(k)$ sums up to value 1. consider the Figure 1 which shows communication graph for network G with four nodes after 6 rounds of an execution of SAWEW algorithm. each node chooses the weight of 1/3. choosing any arbitrary round, we need to show that the exponentially weighted sums of $\boldsymbol{A}(k)$ row equals 1.

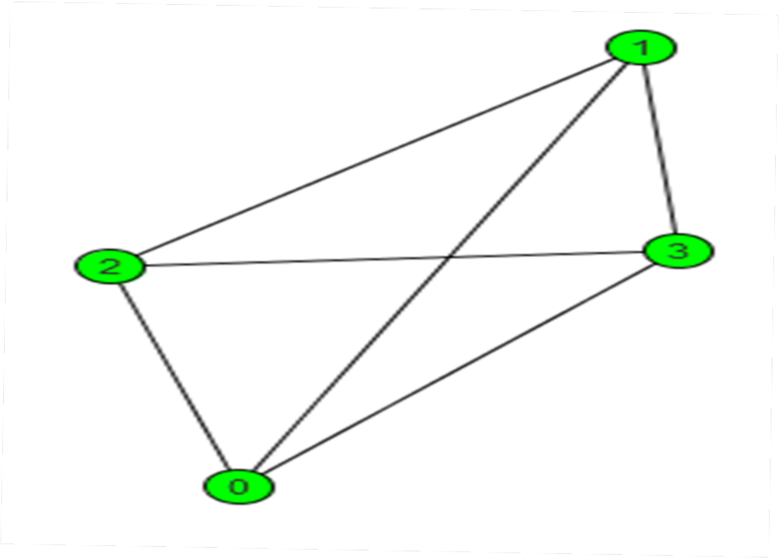


Figure E.1: Communication Graph for Network G with for Nodes let k = 3. $\lambda = \frac{4}{5}$ or 0.8, 1-$\lambda = \frac{1}{5}$ or 0.2

Equation (10.22) becomes

$$\boldsymbol{A}(3)_{i,j} = \begin{cases} \lambda\Big((1-\lambda)^0\frac{1}{|In_i(3)|} + (1-\lambda)^1\frac{1}{|In_i(2)|} + (1-\lambda)^2\frac{1}{|In_i(1)|}\Big) + (1-\lambda)^3\frac{1}{|In_i(0)|} & \text{(i,j)} \in V \\ 0 & \text{else} \end{cases} \tag{E.1}$$

substituting the values for λ and 1-λ in Equation (E.1) gives

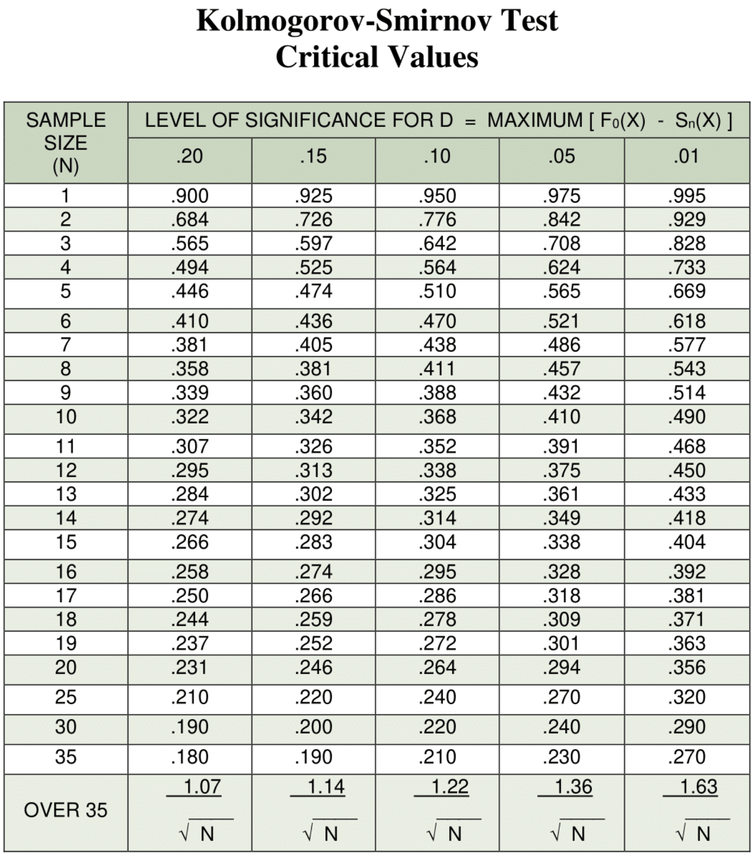
$$\boldsymbol{A}(3)_{i,j} = \begin{cases} \frac{4}{5}\Big(\frac{1}{5}^0\frac{1}{3} + \frac{1}{5}^1\frac{1}{3} + \frac{1}{5}^2\frac{1}{3}\Big) + \frac{1}{5}^3\frac{1}{3} & \text{(i,j)} \in V \\ 0 & \text{else} \end{cases} \tag{E.2}$$

After a little algebra, Matrix $\boldsymbol{A}(k)$ is represented as

$$\left[\begin{array}{cccccc} & 0 & 1 & 2 & 3 & sums \\ 0 & 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 1 \\ 1 & \frac{1}{3} & 0 & \frac{1}{3} & \frac{1}{3} & 1 \\ 2 & \frac{1}{3} & \frac{1}{3} & 0 & \frac{1}{3} & 1 \\ 3 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 & 1 \end{array}\right]$$

Appendix F

KS Table



Kolmogorov-Smirnov Test
Critical Values

SAMPLE SIZE (N)	LEVEL OF SIGNIFICANCE FOR D = MAXIMUM [$F_0(X) - S_n(X)$]				
	.20	.15	.10	.05	.01
1	.900	.925	.950	.975	.995
2	.684	.726	.776	.842	.929
3	.565	.597	.642	.708	.828
4	.494	.525	.564	.624	.733
5	.446	.474	.510	.565	.669
6	.410	.436	.470	.521	.618
7	.381	.405	.438	.486	.577
8	.358	.381	.411	.457	.543
9	.339	.360	.388	.432	.514
10	.322	.342	.368	.410	.490
11	.307	.326	.352	.391	.468
12	.295	.313	.338	.375	.450
13	.284	.302	.325	.361	.433
14	.274	.292	.314	.349	.418
15	.266	.283	.304	.338	.404
16	.258	.274	.295	.328	.392
17	.250	.266	.286	.318	.381
18	.244	.259	.278	.309	.371
19	.237	.252	.272	.301	.363
20	.231	.246	.264	.294	.356
25	.210	.220	.240	.270	.320
30	.190	.200	.220	.240	.290
35	.180	.190	.210	.230	.270
OVER 35	$\frac{1.07}{\sqrt{N}}$	$\frac{1.14}{\sqrt{N}}$	$\frac{1.22}{\sqrt{N}}$	$\frac{1.36}{\sqrt{N}}$	$\frac{1.63}{\sqrt{N}}$

Figure F.1: Kolmogorov-Smirnov test Table

Bibliography

H. Aissaoua, M. Aliouat, A. Bounceur, and R. Euler. A distributed consensus-based clock synchronization protocol for wireless sensor networks. *Wireless Personal Communications*, 95(4):4579–4600, Apr. 2017.

I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38(4):393–422, Mar. 2002.

E. Altman, R. Groenevelt, and P. Nain. Relaying in Mobile Ad Hoc Networks: The Brownian Motion Mobility Model. Technical Report RR-5311, INRIA, Sept. 2004.

N. Aschenbruck, R. Ernst, E. Gerhards-Padilla, and M. Schwamborn. BonnMotion: a mobility scenario generation and analysis tool. In *Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques*. ICST, 2010.

N. Aschenbruck, A. Munjal, and T. Camp. Trace-based mobility modeling for multi-hop wireless networks. *Computer Communications*, 34(6):704–714, May 2011.

F. Bai, N. Sadagopan, and A. Helmy. IMPORTANT: a framework to systematically analyze the impact of mobility on performance of routing protocols for adhoc networks. In *Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No.03CH37428)*, 2003.

S. A. Bakura, A. Lambert, and T. Nowak. Clock synchronization with adaptive weight factor for mobile networks. In *2020 28th Mediterranean Conference on Control and Automation (MED)*, pages 1033–1038, 2020.

C. Bettstetter. Mobility modeling in wireless networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(3):55–66, July 2001.

J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking - MobiCom 98*, 1998.

A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott. Impact of human mobility on opportunistic forwarding algorithms. *IEEE Transactions on Mobile Computing*, 6(6):606–620, June 2007.

B. Charron-Bost, M. Függer, and T. Nowak. Amortized averaging algorithms for approximate consensus. *CoRR*, abs/1512.04222, 2015.

C.-C. Chiang and M. Gerla. On-demand multicast in mobile wireless networks. In *Proceedings Sixth International Conference on Network Protocols (Cat. No.98TB100256)*. IEEE Comput. Soc, 1998.

A. Clauset, C. R. Shalizi, and M. E. J. Newman. Power-law distributions in empirical data. *SIAM Review*, 51(4):661–703, Nov 2009. ISSN 1095-7200.

A. Clementi, A. Monti, F. Pasquale, and R. Silvestri. Information spreading in stationary markovian evolving graphs. *IEEE Transactions on Parallel and Distributed Systems*, 22(9):1425–1432, Sept. 2011.

D. Dolev, J. Y. Halpern, B. Simons, and R. Strong. Dynamic fault-tolerant clock synchronization. *Journal of the ACM*, 42(1):143–185, Jan. 1995.

J. P. Dominguez-Morales, A. Rios-Navarro, M. Dominguez-Morales, R. Tapiador-Morales, D. Gutierrez-Galan, D. Cascado-Caballero, A. Jimenez-Fernandez, and A. Linares-Barranco. Wireless sensor network for wildlife tracking and behavior classification of animals in doñana. *IEEE Communications Letters*, 20(12):2534–2537, Dec. 2016.

F. Ekman, A. Keränen, J. Karvo, and J. Ott. Working day movement model. In *Proceeding of the 1st ACM SIGMOBILE workshop on Mobility models - MobilityModels 08*, 2008.

M. Fugger, T. Nowak, and B. Charron-Bost. Diffusive clock synchronization in highly dynamic networks. In *IEEE 49th Annual Conference on Information Sciences and Systems*, Mar 2015.

A. Galati, K. Djemame, and C. Greenhalgh. Analysis of human mobility patterns for opportunistic forwarding in shopping mall environments. *Social Network Analysis and Mining*, 5(1), Apr. 2015.

S. Ganeriwal, R. Kumar, and M. B. Srivastava. Timing-sync protocol for sensor networks. In *Proceedings of the first international conference on Embedded networked sensor systems - SenSys '03*, 2003.

E. Garone, A. Gasparri, and F. Lamonaca. Clock synchronization protocol for wireless sensor networks with bounded communication delays. *Automatica*, 59:60–72, Sept. 2015.

C. S. Gillespie. Fitting heavy tailed distributions: ThepoweRlawPackage. *Journal of Statistical Software*, 64(2), 2015.

M. C. González, C. A. Hidalgo, and A.-L. Barabási. Understanding individual human mobility patterns. *Nature*, 453(7196):779–782, June 2008.

M. Gorawski and K. Grochla. The real-life mobility model: RLMM. In *Second International Conference on Future Generation Communication Technologies (FGCT 2013)*. IEEE, Nov. 2013.

F. Gustafsson and F. Gunnarsson. Positioning using time-difference of arrival measurements. In *2003 International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03)*. IEEE, 2003.

C. Y. Han. *Clock Synchronization and Localization for Wireless Sensor Network*. PhD thesis, Universite Paris-Saclay, 11 2018.

C.-Y. Han, T. Nowak, and A. Lambert. Pulse synchronization for vehicular networks. In *IEEE Intelligent Vehicles Symposium (IV)*, Jun, 2018.

T. Henderson. Project page of the crawdad, 2020. URL `http://crawdad.org/`.

K. Herrmann. Modeling the sociological aspects of mobility in ad hoc networks. In *Proceedings of the 6th international workshop on Modeling analysis and simulation of wireless and mobile systems - MSWIM 03*. ACM Press, 2003.

D. Hrabčák, M. Matis, L. Doboš, and J. Papaj. Students social based mobility model for MANET-DTN networks. *Mobile Information Systems*, 2017:1–13, 2017.

W. Hsu, K. Merchant, H. wei Shu, C. hsin Hsu, and A. Helmy. Weighted waypoint mobility model and its impact on ad hoc networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, 9(1):59–63, Jan. 2005.

W. Hsu, T. Spyropoulos, K. Psounis, and A. Helmy. Modeling time-variant user mobility in wireless mobile networks. In *26th International Conference on Computer Communications*. IEEE, 2007.

P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot. Pocket switched networks: Real-world mobility and its consequences for opportunistic forwarding. Technical Report UCAM-CL-TR-617, University of Cambridge, August 2005. SRC Research Report 18.

A. Jardosh, E. Belding-Royer, K. Almeroth, and S. Suri. Real-world environment models for mobile network evaluation. *IEEE Journal on Selected Areas in Communications*, 23(3):622–632, Mar. 2005.

D. B. Johnson and D. A. Maltz. Dynamic source routing in ad hoc wireless networks. In *The Kluwer International Series in Engineering and Computer Science*, pages 153–181. Springer US, 1996.

S. C. Johnson. Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254, Sept. 1967.

A. Keränen, J. Ott, and T. Kärkkäinen. The ONE Simulator for DTN Protocol Evaluation. In *SIMUTools '09: Proceedings of the 2nd International Conference on Simulation Tools and Techniques*, New York, NY, USA, 2009. ICST. ISBN 978-963-9799-45-5.

M. Kim, D. Kotz, and S. Kim. Extracting a mobility model from real user traces. In *Proceedings of 25th International Conference on Computer Communications*. IEEE, 2006.

D. Kotz and K. Essien. Analysis of a campus-wide wireless network. In *Proceedings of the 8th annual international conference on Mobile computing and networking - MobiCom 02*. ACM Press, 2002.

L. Lamport and P. M. Melliar-Smith. Synchronizing clocks in the presence of faults. *Journal of the ACM*, 32(1):52–78, Jan. 1985.

G. F. Lawler and V. Limic. *Random Walk: A Modern Introduction.* Cambridge University Press, 2009.

K. Lee, S. Hong, S. Kim, I. Rhee, and S. Chong. Demystifying levy walk patterns in human walks, 2008.

K. Lee, S. Hong, S. J. Kim, I. Rhee, and S. Chong. SLAW: A new mobility model for human walks. In *IEEE INFOCOM 2009 - The 28th Conference on Computer Communications*, Apr. 2009.

W. Li, Z. Wang, G. Wei, L. Ma, J. Hu, and D. Ding. A survey on multisensor fusion and consensus filtering for sensor networks. *Discrete Dynamics in Nature and Society*, 2015:1–12, 2015.

A. Mei and J. Stefa. Swim: A simple model to generate small mobile worlds. In *IEEE INFOCOM 2009*, pages 2106–2113, 2009.

A. Mei, G. Morabito, P. Santi, and J. Stefa. Social-aware stateless forwarding in pocket switched networks. In *2011 Proceedings IEEE INFOCOM*. IEEE, Apr. 2011.

A. Munjal, T. Camp, and W. C. Navidi. SMOOTH. In *Proceedings of the 14th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems - MSWiM 11*, 2011.

A. Munjal, T. Camp, and N. Aschenbruck. Smooth- a simple way to model human mobility. *Journal of Electrical and Computer Engineering*, 2012:1–16, 2012.

M. Musolesi, S. Hailes, and C. Mascolo. An ad hoc mobility model founded on social network theory. In *Proceedings of the 7th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems - MSWiM 04*, 2004.

A. D. Nguyen, P. Sénac, V. Ramiro, and M. Diaz. STEPS - an approach for human mobility modeling. In *NETWORKING 2011*, pages 254–265. Springer Berlin Heidelberg, 2011.

S. Nishad and T. Pandey. Realistic simulation of vehicularnetwork in urban and semi-urban area of india. *International Journal for Research in Applied Science and Engineering Technology (IJRASET)*, 6(9):438–447, Sept. 2018.

B. A. A. Nunes. *Characterizing User Mobility in Wireless Networks.* PhD thesis, University of California Santa Cruz, School of Engineering, 2012.

V. Ogrizovic, J. Gucevic, and S. Delcev. Ambient Temperature Changes and the Impact to Time Measurement Error. *Publications de l'Observatoire Astronomique de Beograd*, 91:105–110, Dec. 2012.

OpenStreetMap contributors. Planet dump retrieved from https://planet.osm.org . https://www.openstreetmap.org, 2019.

C. Papageorgiou, K. Birkos, T. Dagiuklas, and S. Kotsopoulos. An obstacle-aware human mobility model for ad hoc networks. In *IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, Sept. 2009.

P. Pirozmand, G. Wu, B. Jedari, and F. Xia. Human mobility in opportunistic networks: Characteristics, models and prediction methods. *Journal of Network and Computer Applications*, 42:45–58, June 2014.

C. Prandi, L. Monti, C. Ceccarini, and P. Salomoni. Smart campus: Fostering the community awareness through an intelligent environment. *Mobile Networks and Applications*, 25(3):945–952, Feb. 2019.

T. J. P. Project. *OpenJUMP GIS - the free, java-based open source GIS*, 2008. URL http://www.openjump.org.

R. Rastogi, I. Thaniarasu, and S. Chandra. Design implications of walking speed for pedestrian facilities. *Journal of Transportation Engineering*, 137(10):687–696, Oct. 2011.

I. Rhee, M. Shin, S. Hong, K. Lee, and S. Chong. On the levy-walk nature of human mobility. In *INFOCOM 2008 - The 27th Conference on Computer Communications*. IEEE, Apr. 2008.

I. Rhee, M. Shin, S. Hong, K. Lee, S. Kim, and S. Chong. Crawdad dataset ncsu/mobilitymodels (v. 2009-07-23), 2009.

K. Rome. Time synchronization in ad hoc networks. In *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking and computing*, 2001.

M. Sanchez. Mobility models, 1998. URL `http://www.disca.upv.es/misan/mobmodel.htm`. Accessed: 2020-04-8.

P. Santi. *Mobility Models for Next Generation Wireless Networks*. A John Wiley and Sons, Ltd., 2012.

L. Schenato and F. Fiorentin. Average TimeSynch: A consensus-based protocol for clock synchronization in wireless sensor networks. *Automatica*, 47(9):1878–1886, Sept. 2011.

M. Schwamborn and N. Aschenbruck. Introducing geographic restrictions to the SLAW human mobility model. In *21st International Symposium on Modelling, Analysis and Simulation of Computer and Telecommunication Systems*. IEEE, Aug. 2013.

R. Scopigno and H. A. Cozzetti. GNSS synchronization in vanets. In *IEEE 3rd International Conference on New Technologies, Mobility and Security*, Dec. 2009.

J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau. CRAWDAD dataset cambridge/haggle (v. 2009-05-29). Downloaded from `https://crawdad.org/cambridge/haggle/20090529/imote`, May 2009. traceset: imote.

J. Sidorenko, V. Schatz, N. Scherer-Negenborn, M. Arens, and U. Hugentobler. Fusion of time of arrival and time difference of arrival for ultra-wideband indoor localization, 2019.

O. Simeone and U. Spagnolini. Distributed time synchronization in wireless sensor networks with coupled discrete-time oscillators. *EURASIP Journal on Wireless Communications and Networking*, 1(1), June 2007.

P. Sommer and R. Wattenhofer. Symmetric clock synchronization in sensor networks. In *ACM Proceedings of the workshop on Real world wireless sensor networks REALWSN*, 2008.

Tkk/Comnet. Project page of the one simulator, 2013. URL `http://www.netlab.tkk.fi/tutkimus/dtn/theone`.

J. van Greunen and J. Rabaey. Lightweight time synchronization for sensor networks. In *Proceedings of the 2nd ACM international conference on Wireless sensor networks and applications - WSNA '03*, 2003.

W. Wang, J. Wang, M. Wang, B. Wang, and W. Zhang. A realistic mobility model with irregular obstacle constraints for mobile ad hoc networks. *Wireless Networks*, 25(2): 487–506, Sept. 2017.

D. Wu, J. Li, and J. Liu. A random obstacle-based mobility model for delay-tolerant networking. *International Journal of Network Management*, 21(4):326–339, July 2011.

R. Xu and D. C. Wunsch. *Clustering*. Wiley-IEEE Press, 2008.

W. Ye, J. Heidemann, and D. Estrin. An energy-efficient MAC protocol for wireless sensor networks. In *Proceedings of 21st Annual Joint Conference of the Computer and Communications Societies*. IEEE, 2002.

J. Yoon, M. Liu, and B. Noble. Random waypoint considered harmful. In *22nd Annual Joint Conference of the Computer and Communications Societies (IEEE Cat. No.03CH37428)*, 2003.

Q. Zheng, X. Hong, J. Liu, D. Cordes, and W. Huang. Agenda driven mobility modelling. *International Journal of Ad Hoc and Ubiquitous Computing*, 5(1):22, 2010.

Y. Zheng, Q. Li, Y. Chen, X. Xie, and W.-Y. Ma. Understanding mobility based on gps data. In *In Proceedings of ACM conference on Ubiquitous Computing*. ACM, Sept 2008.

M. Zignani and S. Gaito. Extracting human mobility patterns from GPS-based traces. In *2010 IFIP Wireless Days*. IEEE, Oct. 2010.

université
PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies
de l'information et de
la communication (STIC)

Titre: Modèles de mobilité et synchronisation d'horloge dans les réseaux sans fil.

Mots clés: Modèle de mobilité, Synchronisation d'horloge, Couplage par impulsions, Réseaux tolérants aux délais, Réseau de capteurs, Réseau dynamique.

Résumé: La mobilité dans les réseaux ad-hoc conventionnels est un défi en raison de l'invalidation constante des chemins de bout en bout. Nous traitons, dans cette thèse, spécifiquement des réseaux mobiles ad-hoc où les humains sont les principaux porteurs d'appareils mobiles communicants. Une bonne compréhension de la mobilité humaine permet la conception d'un modèle de mobilité réaliste en tant qu'outil d'évaluation des protocoles de réseau.

Les modèles conventionnels d'évaluation des protocoles des premiers réseaux ad hoc (par exemple, random walks, random waypoints, random directions) ne parviennent pas à simuler correctement la mobilité humaine. Des études récentes ont montré que la mobilité humaine est influencée par les habitudes personnelles, les relations sociales, les caractéristiques environnementales et les préférences de localisation. Par conséquent, un modèle réaliste devrait inclure ces caractéristiques.

À cet égard et à l'aide de traces réelles, nous avons développé une heuristique pour définir un modèle de mobilité humaine basée sur des caractéristiques spatiales, temporelles et de connectivité. Nous avons remarqué des clusters de mouvements dynamiques temporels associés à des utilisateurs individuels. Nous avons étudié la distribution de la distance parcourue, du temps de pause, de l'angle de déplacement, de la durée de contact et de la durée d'intercontact. Motivés par nos résultats, nous avons proposé un nouveau modèle de mobilité qui imite de manière réaliste les caractéristiques de la mobilité humaine. Notre modèle a été validé en comparant ses traces synthétiques à des mesures de mobilité réelles.

Dans un environnement de campus intelligent, les réseaux prennent en charge les applications de surveillance environnementale et de positionnement intérieur/extérieur, parfois avec un déploiement important de capteurs. Compte tenu des limitations des capteurs telles qu'autonomie énergétique, capacité de calcul limité, et la dynamique, les horloges des capteurs doivent être synchronisées pour exécuter des algorithmes de fusion de données, mettre en œuvre des protocoles de gestion de l'énergie ou un traitement temps réel des applications où la sécurité est importante.

Compte tenu de cela, nous avons proposé un algorithme de synchronisation d'horloge distribuée à couplage d'impulsions pour des réseaux de capteurs sans fil. Notre algorithme permet de réduire les décalages d'horloge dus aux conditions ambiantes, à la mobilité ou aux défauts de fabrication. Pour ce faire, les capteurs mesurent les différences de temps en échangeant uniquement des impulsions au lieu de paquets.

Par conséquent, notre algorithme est léger et robuste à la défaillance de capteurs du réseau. L'algorithme proposé est comparé aux travaux antérieurs avec des paramètres statiques et mobiles. Les résultats montrent qu'il peut réduire le décalage d'horloge, en particulier dans un environnement dynamique avec une dérive d'horloge importante et des changements topologiques inattendus comme ceux apparaissant dans les réseaux de véhicules.

université
PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies
de l'information et de
la communication (STIC)

Title: Patterns of Mobility and Clock Synchronization in Wireless Networks.

Keywords: Mobility model, Clock synchronization, Pulse-coupled, Delay tolerant network, Sensor network, Dynamic network.

Abstract: Mobility in conventional ad-hoc networks is a challenge due to the constant invalidation of end-to-end paths. We deal with mobile ad-hoc networks where humans are the main carriers of mobile devices. A good understanding of human mobility patterns aids the design of a realistic mobility model as a tool for evaluating network protocols.

Conventional models for evaluating network protocols in early ad-hoc networks (e.g., random walks, random waypoints, random directions) fail to properly capture human mobility. In fact, recent studies have shown that human mobility is influenced by personal habits, social relationships, environmental features, and locations preferences. Therefore, a realistic model should be able to include these features.

In this regard, we develop a heuristic to characterize human mobility based on the spatial, temporal, and connectivity features using real traces. Consequently, we uncover temporal dynamic movement clusters associated with individual users. We also study the distribution of the travel distance, pause time, angle of movement, contact duration, and inter-contact duration. Motivated by our findings, we proposed a new synthetic mobility model that mimics realistic features of human mobility. We validate the model by comparing its synthetic traces against real mobility measurements.

Moreover, in a smart-campus environment, networks support applications for environmental monitoring and indoor/outdoor positioning, sometimes with a large deployment of sensors. Considering the limitation of sensors such as battery limitation, dynamicity, and low computing clock rate, sensor clocks need to have a common time to perform information fusion algorithms, implement energy management protocols, or real-time processing for safety applications. Given this, we proposed a pulse-coupled distributed clock synchronization algorithm for wireless sensor networks to reduce the clock skew due to the ambient conditions, mobility, or manufacturing defects. In our algorithm, sensors measure time differences by only exchanging zero-bit pulse instead of packets. Therefore our algorithm is lightweight and robust to the failure of the sensors in the network. The proposed algorithm is compared to previous works under static and mobile settings, and the results show that it can reduce the clock skew, especially in a dynamic environment with high uncertainty in clock drift and unexpected topological changes like vehicular networks.