TALLINN UNIVERSITY OF TECHNOLOGY
DOCTORAL THESIS
14/2019

Algorithms for Learning and Adaptation Over Networks – Distributed Leader Selection

SANDER ULP

TTÜ PRESS

TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies
Thomas Johann Seebeck Department of Electronics

This dissertation was accepted for the defense of the degree of Doctor of Philosophy in Electronics and Telecommunication on 4 March 2019

Supervisor:
Professor Muhammad Mahtab Alam
Thomas Johann Seebeck Department of Electronics
School of Information Technologies
Tallinn University of Technology
Tallinn, Estonia

Co-Supervisor:
Professor Yannick Le Moullec
Thomas Johann Seebeck Department of Electronics
School of Information Technologies
Tallinn University of Technology
Tallinn, Estonia

Opponents:
Professor Luca Reggiani
Politecnica Di Milano
Milan, Italy

Professor Muhammad Ali Imran
University of Glasgow
Glasgow, The United Kingdom

Defense of the thesis: 12 April 2019, Tallinn

Declaration:
Hereby I declare that this doctoral thesis, my original investigation and achievement, submitted for the doctoral degree at Tallinn University of Technology, has not been submitted for any academic degree elsewhere.

Sander Ulp

---
signature

European Union
European Social Fund

Investing in your future

ISSN 2585-6898 (publication)
ISBN 978-9949-83-409-9 (publication)
ISSN 2585-6901 (PDF)
ISBN 978-9949-83-410-5 (PDF)

TALLINNA TEHNIKAÜLIKOOL
DOKTORITÖÖ
14/2019

Õppimisalgoritmid hajutatud võrkude tarbeks – juhtsõlme hajus valimine

SANDER ULP

To my family.

TABLE OF CONTENTS

LIST OF PUBLICATIONS

The work of this thesis is based on the following publications:

A S. Ulp and T. Trump, “Distributed adaptive network with SNR weighed communication,” in Digital Information, Networking, and Wireless Communications (DINWC), 2015 Third International Conference on. IEEE, 2015, pp. 83–87. [ETIS 3.1]

B S. Ulp and T. Trump, “Leader selection in cooperative network based on MDL subspace algorithm for cognitive radio,” in 2016 50th Asilomar Conference on Signals, Systems and Computers. IEEE, 2016, pp. 704–708. [ETIS 3.1]

C S. Ulp, Y. Le Moullec, and M. M. Alam, “LMS-based leader selection for distributed estimation,” in 2017 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT). IEEE, 2017, pp. 211–215. [ETIS 3.1]

D S. Ulp, Y. Le Moullec, and M. M. Alam, ”Energy-efficient distributed leader selection algorithm for energy-constrained wireless sensor networks.” IEEE Access. 2019, 7, pp. 4410–4421 [ETIS 1.1]

AUTHOR'S CONTRIBUTIONS TO THE PUBLICATIONS

The author's contributions to the publications in this thesis are:

A The author 1) proposed using the MDL subspace algorithm for estimating the SNR at the nodes in the diffusion algorithm and use them as the basis for the weight calculation and 2) developed the theoretical basis for the algorithm. The author performed the computer simulations, analysed and compared the results. The author wrote the paper and the supervisor helped with the revision and by giving feedback.

B The author proposed the distributed leader selection algorithm for distributed estimation and developed the algorithm based on the leader selection by relying on the SNR estimates from the MDL subspace algorithm. The author performed the computer simulations, analysed and compared the results. The author wrote the paper and the supervisor helped with the revision and by giving feedback.

C The author investigated and derived the analytical performance of the distributed leader selection algorithm in a LMS-based (least mean squares) setting and compared it to the analytical performance of the diffusion algorithms. The author performed the computer simulations, analysed and compared the results. The author wrote the paper and the supervisors helped with the revision.

D The author investigated and derived the complexity and energy consumption of the distributed leader selection algorithm and diffusion algorithm and proposed a novel energy-efficient distributed leader selection algorithm. The author performed the computer simulations based on real hardware models, analysed and compared the results. The author wrote the paper and the supervisors helped with the revision and theoretical feedback.

NOMENCLATURE

Abbreviations and Acronyms

ATC	Adapt and Then Combine
CTA	Combine and Then Adapt
CR	Cognitive Radio
DLS	Distributed Leader Selection
EEDLS	Energy-Efficient Distributed Leader Selection
EE	Energy-Efficient
FC	Fusion Centre
IoT	Internet of Things
LMS	Least Mean Squares
MDL	Minimum Description Length
MSE	Mean Squared Error
MSD	Mean Squared Deviation
NP-Hard	Non-deterministic Polynomial-time Hardness
RLS	Recursive Least Squares
SNR	Signal to Noise Ratio
WSN	Wireless Sensor Network

Notation

e, E – Italic letters are used for scalars.

$\mathbf{x}$ – Lower case bold letters are denoting vectors.

All vectors are column vectors except for regression vectors, which are denoted as $\mathbf{u}_{k,i}$.

The operator $E[\cdot]$ stands for mathematical expectation of the subject.

y^* denotes the complex conjugate transpose of y.

Part I

Introduction

1. BACKGROUND AND MOTIVATION

Learning and adaptation over networks is a topic that has been researched in several fields. These fields include, but are not limited to - mathematics [1], computer science [2, 3], economics [4], biology [5], sociology [6, 7], psychology [8]. The topic has evolved rapidly in the past decade and is still receiving attention with the emergence of systems and applications that utilize networks to learn, estimate or organize [9–12]. Classical systems and learning theories in the past have focused on stand-alone systems or learners with success, but recent progress and discoveries in biological sciences, animal behaviour studies and neuroscience of the brain have reshaped the approach in tackling these problems [12].

These findings have revealed how complex systems with self-organizing, adaptation and dynamic behaviour rely on the coordination and cooperation of simpler units in a decentralized manner [13–16]. The units themselves are unable to solve the problems or their performance in solving these problems is greatly poorer in comparison to a network of cooperating units [17]. The interaction between units or agents enables them to share information, creating a system that is more resilient to failure and is able to adapt and learn according to the situation [12, 13, 15, 17]. Several examples of biological networks can be found in the nature that exhibit this kind of sophisticated behaviour of agents with limited abilities [18–20]. Groups of fishes are able to navigate the environment and react to surroundings in a organized manner in a decentralized way. This allows the fishes to respond collectively to predators and other threats [21–23]. Another example is bee swarms which are able to find a way to a new hive only relying on a small number of informed bees [24–26]. Birds that migrate to warmer areas and travel in V formation collectively by cooperating [27, 28].

These discoveries have lead to finding ways to implement these behaviours in solving problems in other disciplines, such as computer science, signal processing, telecommunication, robotics etc. [29–31]. The common goal among these disciplines is to develop theories and tools that enable the design of networks that mimic the sophisticated behaviour and learning processing abilities of the biological networks [32–34]. This avoids using centralized processing units and keeps interaction local among agents in networks [35–37]. Another advantage of these approaches is the simplicity of the agents and low complexity of the tasks needed to be carried out by each agent [18–20].

In the context of this PhD thesis, the work that is of interest is related to research and development of these networks and algorithms to solve problems related to communication systems and signal processing in radio networks. This include optimization, adaptation and learning problems that need to be solved in an efficient and distributed manner using a network of agents with localized interaction. These problems arise in several applications and can be addressed in several ways to either improve the performance, robustness, resilience, privacy and security of the system [38]. Applications include systems that deal with various fields which employ radio networks or wireless sensor networks, such as agriculture, military, environmental studies, etc [39–41].

The motivation behind this work stems from the current situation of increased wireless data traffic which has had an exponential growth due to the increase of wireless devices and their usage [42]. Applications and systems in smart cities and IoT (Internet-of-Things) benefit from decentralized and self-organizing behaviour [43]. These applications are also meant to work in constrained conditions such as having limited amount of energy, which also has motivated to focus on energy-efficiency (EE) in distributed processing [44–46].

More specifically, the focus and interest is on radio networks of agents that cooperate to solve some kind of estimation problem in a distributed manner. Applications that come under consideration include cognitive radio (CR), wireless sensor networks (WSNs) and in a broader sense applications involved with distributed estimation and processing.

Cognitive radio systems are shown to benefit from distributed estimation [47]. CR is a concept that has been developed to combat the scarceness of available spectrum resources [48]. This allows to use the already licensed spectrum when the licensed user (primary user) is not utilizing the spectrum and allows secondary users to use the spectrum in the meantime. In this case, the secondary users have to monitor and estimate whether the primary user is transmitting or not. If the secondary users cooperate, the estimation process can be enhanced and the detection success of the primary user activity is increased [49–52].

Although this work is primarily for radio networks, it is not limited only to the radio networks and can be adapted to solving similar problems in other fields and application.

1.1. Problem statement and research questions

As the learning and adaptation field is evolving rapidly, there are many unresearched problems and approaches to consider tackling these problems [12]. This PhD work mainly focuses on the development of algorithms for distributed estimation and further analysis of the algorithms' performances. The current state-of-the-art algorithms in distributed estimation that are able to learn and adapt in real-time and that have shown good results are known as diffusion algorithms [53–57]. These algorithms have been studied in various situations,

such as under changing topologies and in the presence of link failure [58, 59], under imperfect node to node communication [60, 61] and in other situations and conditions [62, 63].

One of the problems related to the current state-of-the-art diffusion algorithm is selecting appropriate communication weights in the network which has to be based on some qualitative measurement to indicate, which nodes are better at solving the problem in the given application [57]. The work in this PhD started with analysing and proposing a method to estimate the secondary parameter required for the calculations of the weights [Paper A].

The research lead to the conclusion that the diffusion algorithm's performance relies on the accuracy of the secondary parameter estimation [Paper C]. This triggered interest in devising an algorithm whose performance does not heavily rely on the secondary parameter estimation accuracy and led to studying leader selection algorithms. In addition, the leader selection algorithms are specifically suitable for application where there is a need for selecting the best performing node. The currently available works on leader selection algorithms are introduced in Subsection 2.3.5; these are not fully distributed or are unable to adapt in real-time. This led to proposing the distributed leader selection (DLS) algorithm which is fully distributed, adapts and selects the leader node in real-time, and does not heavily rely on the secondary parameter estimation accuracy [Paper B, Paper C]. In addition, the DLS algorithm is able to retain its performance under inaccurate secondary parameter estimation. The theoretical contributions were further extended by moving towards practical realization of the algorithms by evaluating the computational complexity both from algorithmic and hardware view point and proposing a method to improve the energy-efficiency of the DLS algorithm [Paper D].

In more detail this PhD thesis 1) proposes a SNR weighed diffusion algorithm, 2) proposes a novel fully distributed leader selection algorithm, 3) analyses the theoretical performance of the DLS algorithm and the diffusion algorithm, 4) analyses the computational complexity and energy consumption of the DLS algorithm and the diffusion algorithm and 5) proposes an energy-efficient distributed leader selection algorithm.

The research questions raised and answered in this PhD thesis are:

1. Can we estimate the weights for the diffusion algorithm in a blind manner with no *a priori* information?
2. Is it possible to implement a fully distributed leader selection algorithm?
3. What is the performance of the DLS algorithm and how does it compare to the state-of-the-art solutions?
4. What are the computational complexities of the DLS algorithm and the diffusion algorithm?

5. What is the energy consumption of the DLS algorithm and the diffusion algorithm?

6. How can we reduce the energy consumption for the DLS algorithm?

1.2. Contributions of the thesis

To summarize the previous section, the main contributions of the thesis are as follows:

A A novel method for secondary parameter calculation in the diffusion algorithm based on the SNR estimates using the MDL subspace algorithm. The algorithm for calculating the SNR estimates and infusing them in the diffusion algorithm weight calculations. The method is shown to outperform the equally weighed network and non-cooperating network.

B A novel, robust and fully distributed leader selection algorithm that does not require any *a priori* information of the network or architecture in place. The algorithm from Paper A is shown to be used as a basis for the leader selection algorithm. The robustness to the secondary parameter accuracy is described. Results show that the algorithm is able to outperform the equally weighed network and non-cooperating network.

C A LMS-based implementation of the fully distributed leader selection algorithm and analysis of the analytical performance of algorithm in comparison to the diffusion algorithm. The analysis shows that the leader selection algorithm is outperformed in most cases by the diffusion algorithm, but in the presence of a node in a more favourable situation the algorithm has similar performance and even outperforms the diffusion algorithm.

D The theoretical complexities, the computational and communication energy consumption are estimated for the distributed leader selection algorithm and the diffusion algorithm. The results show that the leader selection algorithm is less complex and consumes less computational energy than the diffusion algorithm. However, the communication energy consumption makes the computational energy differences marginal for both algorithms. A novel energy-efficient distributed leader selection algorithm is developed; it is less complex and is able to decrease the energy consumption of the network by $\mathbf{32-53}\%$ and can extend the network lifetime by $\mathbf{14-46}\%$ compared to the DLS algorithm and the diffusion algorithm. The

results are simulated considering real hardware models (MSP430 and RLS10).

1.3. Thesis outline

The rest of this thesis is structured as follows:

Chapter 2: Learning and adaptation over networks. In this chapter the theoretical background and the algorithms that have been used in the learning and adaptation over networks are presented with the relevant state-of-the-art references. The algorithms are presented starting with the non-cooperating network and moving on to fully distributed algorithms. The weight calculations for the algorithms are included as a separate subsection. The chapter includes the contributions of the thesis author to the state-of-the-art theoretical background and the chapter ends with a summary.

Chapter 3: Overview of publications. An overview of the conference and journal articles, which are included in Part II of the thesis, is given in this chapter.

Chapter 4: Conclusions. This chapter provides the conclusions and the research questions that were raised in the beginning of the thesis are answered. The main claims of the thesis are presented and the possible future work is provided.

Part II: Included Publications. Part II of the thesis includes the publications that thesis is based on.

2. LEARNING AND ADAPTATION OVER NETWORKS

This chapter describes the approaches that have been used in the learning and adaptation over networks in the signal processing field and radio networks. The state-of-the-art is presented and then extended by the contributions made in this thesis to the field giving a clear understanding of the impact of the work done. In radio networks we refer to the agents or units in cooperation as nodes. First, the non-cooperating network is described where the nodes work by themselves without any communication with other nodes. The non-cooperating network is a good baseline to understand how a single node works outside of cooperation. Second, an overview of the different approaches with cooperation among nodes is given, which includes centralized and decentralized approaches. Third, the decentralized approaches known as the distributed stochastic-gradient solutions are introduced, which includes algorithms such as incremental, consensus, diffusion and distributed leader selection. The classification and the outline of this chapter can be seen in Figure 2.1. The self-loops of nodes in figures showing the topologies in the following sections have been omitted for clarity. A subsection is dedicated to the weight calculations for the algorithms. This chapter's organization and line of thought (i.e. starting from the non-cooperating network and moving on to distributed algorithms) is inspired by the book titled "Adaptation, Learning, and Optimization Over Networks" by Ali H. Sayed [12].

For the nodes to be able to learn and adapt to the situation, the nodes employ learning algorithms. Stochastic gradient decent algorithms have been used for the learning and adaptation [12]; the algorithms that are discussed in this thesis are based on these algorithms. The algorithms are usually employed as an adaptive filter at each node. For the emergence of a collective sophisticated behaviour, the nodes need to interact with each other. In doing so, the nodes form a network which means there are certain rules to the communication and the formation of the topology of the formed network. In this topology the nodes either share information locally or relay it forward to other nodes or to a centralized processing unit. The nodes or the centralized unit process the information and either distribute the information across the network or the information is forwarded to some other system. The information can be used either by the nodes to make decisions locally or by the centralized unit to make a global decision for the entire network [64].

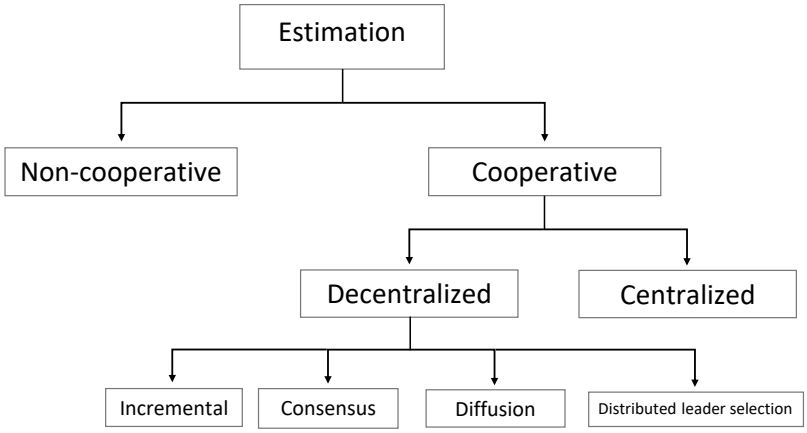


Figure 2.1 Classification of different communication strategies used in estimation.

In this thesis the main interest is in fully distributed solutions which are without any centralized processing unit or predefined architecture and the informations is only shared locally. The problems of not fully distributed solutions are outlined and the restrictions associated with the algorithms are presented. Among other benefits the main advantages of fully distributed algorithms include robustness to failures, lower energy consumption and reduced processing complexity [17, 35].

2.1. Non-cooperating network

A non-cooperating network includes nodes which work on their own to solve the task which is based on estimating a source or a signal. The task is the same for all the nodes in the network. The nodes collect measurements of the source or of the signal and proceed to estimate based on this. A network of K nodes employing an adaptive filter to learn and adapt the signal source is shown in Figure 2.2. The nodes have access to streaming information $d_k(i)$ and $\mathbf{u}_{k,i}$ at each iteration i. Each iteration usually refers to a span of time and can be viewed as a discrete time process. The process can be described as:

$$d_k(i) = \mathbf{u}_{k,i}\mathbf{w}^o + v_k(i), \tag{2.1}$$

where $d_k(i)$ is the measurement at node k, $\mathbf{u}_{k,i}$ is the row regression vector, $\mathbf{w}^o$ is the unknown column vector and $v_k(i)$ is zero-mean white random noise with power $\sigma^2_{v,k}$.

The quantities $v_k(i)$ and $\mathbf{u}_{k,i}$ are assumed to be independent for all k values and $\mathbf{u}_{k,i}$ has a positive-definite covariance matrix, $R_{u,k} = E\ \mathbf{u}^*_{k,i}\mathbf{u}_{k,i} > 0$.

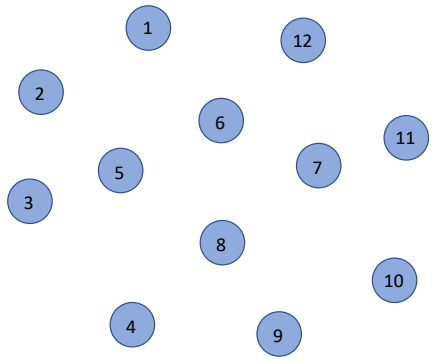


Figure 2.2 An example of a non-cooperating network with $K = 12$ nodes. The absence of edges between the nodes reflects that the nodes do not cooperate.

The task for the nodes is to estimate $\mathbf{w}^o$ by minimizing the global cost function $J_k(w)$.

$$J_k(w) = E \mid d_k(i) - \mathbf{u}_{k,i}\mathbf{w} \mid^2 . \tag{2.2}$$

In this approach the nodes employ the LMS adaptive filters. The nodes can employ some other filter types such as exponential averaging or RLS (recursive least squares) [65]. The LMS adaptation is given as:

$$\mathbf{w}_{k,i} = \mathbf{w}_{k,i-1} + \mu_k \mathbf{u}^*_{k,i}[d_k(i) - \mathbf{u}_{k,i}\mathbf{w}_{k,i-1}], \tag{2.3}$$

where μ_k is a constant positive step-size and $0 < \mu_k < \frac{2}{\lambda_{max}(R_{u,k})}$.

The step-size μ_k can be implemented with a decreasing value, but this inhibits the nodes to learn and adapt because when the step-size is zero the nodes are no longer able to track the source or signal. The nodes estimate and after a certain number of iterations achieve a MSD (mean squared deviation) steady state performance that is dependent on the conditions the node is at. If the nodes share the same conditions, all the nodes achieve at the same steady state performances; however, this is usually not the case [66]. The nodes in better conditions achieve better performance and the nodes in poorer conditions result in poorer performance than the better performing nodes. The advantage of the non-cooperating approach is that the nodes are able to conserve resources which would be spent on communication; however, as will be shown in the next subsections the gains from cooperation usually outweigh the costs of resources spent.

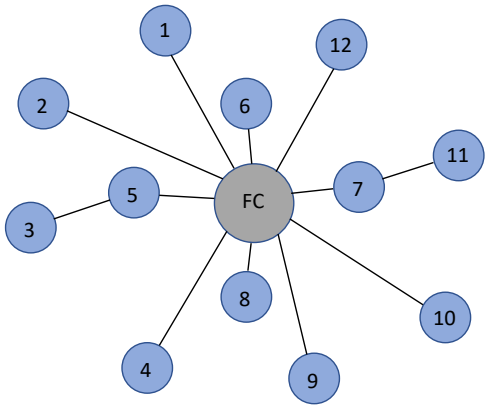


Figure 2.3 Centralized strategy. A network with $K = 12$ nodes and a fusion centre (FC). Nodes 5 and 7 act as transit nodes for node 3 and 11.

2.2. Centralized strategy

Centralized solutions solve the estimation problem by employing a central processing unit that is sometimes referred to as a "fusion centre" (FC) [12]. The FC is connected to all of the nodes either directly or through other nodes (Figure 2.3). This enables the FC to receive the information from all the nodes in the network, see Figure 2.3. The FC processes all of the information and forms a decision or estimate based on this and either passes it on to some other system as input or the information is fed back to the nodes in the network [67]. Centralized solutions have been developed and expanded in several works and have shown good performance in comparion to non-cooperating network as well as able to match the performance of the diffusion algorithms [57]. However, this method has several drawbacks.

Having a FC simplifies the exchange of information in the network as it is all passed on to one location; however, for larger networks the amount of energy required to communicate or pass the information to the centre node is not suitable for energy-constrained application [39, 65]. Processing all the information at one location also requires a node that has high processing power [39]. This means that the central node itself would also need a large amount of energy to receive and process all the information. The FC poses also a single point of failure problem as in the event of the failure of the FC the whole network is rendered useless, furthermore, in information sensitive applications the centralized solutions might not be suitable because of privacy and security considerations [12]. Based on the LMS single agent network (2.3) the centralized solution is given as:

$$\mathbf{w}_i = \mathbf{w}_{i-1} + \mu_k \left(\frac{1}{K} \sum_{k=1}^{K} \mathbf{u}_{k,i}^{*} [d_k(i) - \mathbf{u}_{k,i} \mathbf{w}_{k,i-1}] \right) \tag{2.4}$$

The central node processes the information from all the K nodes in the network and averages their estimates. The centralized solution is able to outperform the non-cooperating network on the network level [57]. However, all nodes might not benefit from the cooperation and additional weighing of the estimates would improve the network performance, which will be discussed in subsection 2.3.4.

2.3. Distributed estimation

In this section and the following subsections we delve into the distributed estimation algorithms from the perspective of radio networks, although the algorithms are not limited to radio networks and can be implemented in other types of networks [30, 31]. Distributed estimation is a method to enhance the performance of the learning and adaptation process in networks. Since the nodes are solving a common task or problem, their efforts can be combined by cooperation [13]. This cooperation among the nodes allows the network of nodes to exhibit sophisticated behaviour. The performance or the behaviour itself might be impossible to achieve using a single node. For example, if the task is to estimate the position of an object or track it, a single node is unable to accomplish this [68]. While cooperating the nodes communicate; the manner of communication depends on the application and the goal that is being accomplished.

Different methods and algorithms have been proposed for distributed estimation. These approaches include strategies such as incremental, consensus, diffusion and leader selection [12]. Some of these algorithms are more restricted and require some kind of architecture or *a priori* knowledge of the network. The advantages and drawbacks of the algorithms are introduced in Subsections 2.3.1, 2.3.2, 2.3.3, 2.3.5.

In addition to the different algorithms and their performances, there are several other aspects required to be assessed and considered. It is important to define what information to share between the nodes and how frequently this should be done. The information shared can be in different forms. The nodes can share their estimates, measurement data or decisions [13]. Sharing more information and more frequently might lead to better performance and accuracy, but sharing information with other nodes is especially costly in resource constrained situations, for example with limited energy availability [69].

The nodes form a graph and the communication is restricted to the paths defined therein. Together they form a topology. The connection can be undirected or directed i.e. information can pass through both ways of the connection or only in one defined way. The topology might be dynamic or static depending on whether the nodes move or are deployed stationary [69]. If required, the nodes should also be able to adapt, which means that if the situation or the input of the network is changed the network will adapt to it and the nodes are able to continue learning this new situation. It is also important to note that the links

between nodes can become unavailable or disconnected leading to changes in the topology and information flow. Nodes might also fail or become unavailable.

The restriction for distributed estimation is that the topology of the network should be strongly connected for meaningful information exchange [12]. Although there exists research on diffusion algorithms in weakly connected networks, it is usually assumed that the network is strongly connected [70].

2.3.1. Incremental strategy

Incremental strategies share and combine information across the network by defining a cyclic path in the network (Figure 2.4) [71]. The incremental strategies have been studied in numerous works [71–79]. The cyclic path visits each node in certain succession. The nodes receive the information and combine the received information with its own information and pass it on to a designated neighbour in the cyclic path. Each node receives the $\mathbf{w}_{k-1,i}$ from the previous node $k-1$ and performs the update until the cycle is complete in the network:

$$\mathbf{w}_{k,i} = \mathbf{w}_{k-1,i} + \frac{\mu_k}{K}\left(\mathbf{u}_{k,i}^{*}[d_k(i) - \mathbf{u}_{k,i}\mathbf{w}_{k,i-1}]\right) \tag{2.5}$$

When all the nodes in the network have been cycled through the estimate is the same as the estimate in the centralized solution i.e. Eq. (2.4). The difference is that the estimate is calculated and the processing is done in a distributed manner which negates the need for a FC and the constraints related to it.

However, the incremental solution has several drawbacks [12]. Firstly, defining a cyclic route that cycles through then all the nodes in the network requires solving an NP-Hard (Non-deterministic Polynomial-time Hardness) problem [80]. Secondly, once the cyclic path is defined, any kind of link failure,

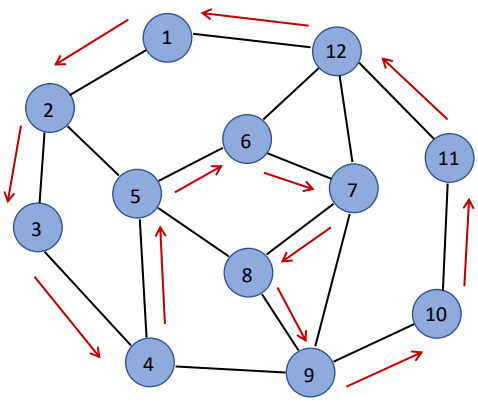


Figure 2.4 Incremental strategy. A cyclic path has been formed through the network that passes all the nodes in the network.

node failure or any topology change will lead to failure of the incremental strategy as the information flow is interrupted [81]. Thirdly, the information sharing in the network is limited as the nodes only receive from one fixed node and share their information with one other defined node in the cyclic path and to reach the final estimate the network has to go through K steps to reach the final estimate which is only available at the node that finished the cycle [12]. The first and the third drawback make the incremental strategy not fully distributed and pose restrictions in the implementations of the algorithm.

2.3.2. Consensus strategy

In contrast to the incremental strategy, in the consensus strategy no cyclic path is required and the communication is not limited to only two neighbours [82]. Further, at each iteration the nodes communicate and estimate without waiting. Neighbours are free to share information among all their neighbours in their vicinity, which means that the nodes processes information that is only available from the neighbouring nodes. Nodes in the vicinity of the node k and that node itself form the neighbourhood $\mathbb{N}_k$. Being connected to several nodes enables consensus strategies to be robust to link failures as well as adapt to topology changes [83]. The consensus strategy consists of two steps, see Algorithm (1) and below.

Algorithm 1 Consensus

for each time instant $i > 0$:
    each node $k = 1, 2, ..., K$ performs the update:
    $\boldsymbol{\psi}_{k,i-1} = \sum_{l \in \mathbb{N}_k} \alpha_{l,k} \mathbf{w}_{l,i-1}$
    $\mathbf{w}_{k,i} = \boldsymbol{\psi}_{k,i-1} + \mu_k \mathbf{u}_{k,i}^{*} [d_k(i) - \mathbf{u}_{k,i} \mathbf{w}_{k,i-1}]$
end

The first step consists of the convex combination of the estimates from the node itself and the nodes from the neighbourhood $\mathbb{N}_k$. $\alpha_{l,k}$ is the combination coefficient between nodes l and k and is calculated based on the selected weighing algorithm which will be discussed in Subsection 2.3.4 and is shown in Figure 2.5. The weights $\alpha_{l,k}$ form the $\mathbf{A}$ matrix where zero entries signify that the connection between nodes does not exist and non-zero entries signify the weight that is used to weigh the information exchange (Table 2.1). The second step is the LMS adaptation step at each node.

The consensus strategies originally relied on the use of two time-scales and decreasing step-sizes [56]. One time-scale is used for the collection of measurements across the nodes and the second one is used for iterating the collected information to achieve consensus. However, this approach is unsuitable to use in real-time applications where the network has to learn and adapt over

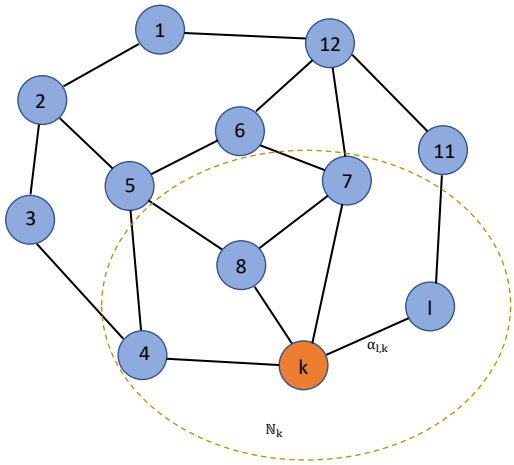


Figure 2.5 Neighbourhood $\mathbb{N}_k$ of node k.

time based on measurement data that is continuously streamed [56]. Single time-scale consensus strategies are able to perform well, but they have stability issues under infinitesimally small step-sizes [56].

2.3.3. Diffusion strategy

Diffusion strategies were specifically developed as distributed solution for being able to respond in real-time to continuous streaming of information over single time-scale [57] and were originally proposed in [84]. The diffusion algorithms are an improvement over the consensus algorithms as they are able to outperform the consensus strategies as well as remain stable while having an exact computational complexity [56]. Diffusion algorithms are the state-of-the-art algorithms in the context of distributed estimation and have shown good performance as well as robustness [12, 53].

There are different variations of the distributed diffusion strategy. The two most known are the Combine-then-Adapt (CTA) and the Adapt-then-Combine

Table 2.1 An example weight matrix $\mathbf{A}$ with $K = 4$ nodes.

$\alpha_{l,k}$	1	2	3	4
1	$\alpha_{1,1}$	$\alpha_{1,2}$	0	0
2	$\alpha_{2,1}$	$\alpha_{2,2}$	$\alpha_{2,3}$	$\alpha_{2,4}$
3	0	$\alpha_{3,2}$	$\alpha_{3,3}$	$\alpha_{3,4}$
4	0	$\alpha_{4,2}$	$\alpha_{4,3}$	$\alpha_{4,4}$

(ATC) algorithms [85]. Similarly to the single time-scale consensus strategy, the CTA algorithm can be divided into two steps, see Algorithm (2). The first step is the combination step, where the estimates from different nodes are combined using convex combination. The second step is the local LMS adaptations step where each node calculates the estimate for the next iteration. The difference between consensus and CTA diffusion lies in the second step where the gradient vector is evaluated based on the diffused value $\boldsymbol{\psi}_{k,i-1}$ instead of the $\mathbf{w}_{k,i-1}$.

Algorithm 2 CTA Diffusion

1: **for** each time instant $i > 0$:
2: each node $k = 1, 2, ..., K$ performs the update:
3: $\boldsymbol{\psi}_{k,i-1} = \sum_{l \in \mathbb{N}_k} \alpha_{l,k} \mathbf{w}_{l,i-1}$
4: $\mathbf{w}_{k,i} = \boldsymbol{\psi}_{k,i-1} + \mu_k \mathbf{u}_{k,i}^* [d_k(i) - \mathbf{u}_{k,i} \boldsymbol{\psi}_{k,i-1}]$
5: **end**

The ATC algorithm is given Algorithm 3 [86]. Each of the nodes at first does the LMS adaptation step and continues by sharing the calculated estimates with its neighbourhood $\mathbb{N}_k$. The nodes receive estimates from their neighbours and in the second step the estimates are combined using convex combination. It has been studied and shown that the ATC diffusion network outperforms the CTA diffusion network [56] and thus, the ATC diffusion network is more often used in the literature as per examples in [87–89].

Algorithm 3 ATC Diffusion

1: **for** each time instant $i > 0$:
2: each node $k = 1, 2, ..., K$ performs the update:
3: $\boldsymbol{\psi}_{k,i} = \mathbf{w}_{k,i-1} + \mu_k \mathbf{u}_{k,i}^* [d_k(i) - \mathbf{u}_{k,i} \mathbf{w}_{k,i-1}]$
4: $\mathbf{w}_{k,i} = \sum_{l \in \mathbb{N}_k} \alpha_{l,k} \boldsymbol{\psi}_{l,i}$
5: **end**

In Algorithms 2 and 3 the nodes share only their estimates with neighbouring nodes. Another option is to extend the cooperation in the diffusion network by exchanging measurements between nodes in addition to the shared estimates [54]. Additional weighing matrix $\mathbf{C}$ is required which will include values $c_{l,k}$ that are the measurement combination coefficients between node l and k. The ATC and CTA diffusion algorithms with measurement exchange are given in Algorithms 4 and 5, respectively.

The measurement exchanges can improve the performance of the diffusion algorithm, but will increase the number of radio communications per iteration in the network [54]. The increased information exchanges might introduce additional interference along with increased energy consumption and reduce the lifetime of the network [90].

Algorithm 4 CTA Diffusion with measurement exchange

1: **for** each time instant $i > 0$:
2: each node $k = 1, 2, ..., K$ performs the update:
3: $\boldsymbol{\psi}_{k,i-1} = \sum_{l \in \mathbb{N}_k} \alpha_{l,k} \mathbf{w}_{l,i-1}$
4: $\mathbf{w}_{k,i} = \boldsymbol{\psi}_{k,i-1} + \mu_k \sum_{l \in \mathbb{N}_k} c_{l,k} \mathbf{u}_{k,i}^* [d_k(i) - \mathbf{u}_{k,i} \boldsymbol{\psi}_{k,i-1}]$
5: **end**

Algorithm 5 ATC Diffusion with measurement exchange

1: **for** each time instant $i > 0$:
2: each node $k = 1, 2, ..., K$ performs the update:
3: $\boldsymbol{\psi}_{k,i} = \mathbf{w}_{k,i-1} + \mu_k \sum_{l \in \mathbb{N}_k} c_{l,k} \mathbf{u}_{k,i}^* [d_k(i) - \mathbf{u}_{k,i} \mathbf{w}_{k,i-1}]$
4: $\mathbf{w}_{k,i} = \sum_{l \in \mathbb{N}_k} \alpha_{l,k}(i) \boldsymbol{\psi}_{l,i}$
5: **end**

2.3.4. Weight calculation

Nodes in the cooperating network are able to communicatate with nodes in its neibourhood $\mathbb{N}_k$. The nodes are operating under different conditions due to the noise profiles and channel gains, which means the performance from node to node differs. Employing the consensus or diffusion algorithms, one has to select how to calculate the weights $\alpha_{l,k}$ in the $\mathbf{A}$ matrix and also in the case of sharing measurements the weights $c_{l,k}$ in the $\mathbf{C}$ matrix. There are several methods and approaches to calculating these weights, such as Metropolis, uniform, relative degree, etc. [54, 55, 83]. The weights and combination rules are not limited to only distributed algorithms and can be also applied to centralized solutions [12]. The most common combination rules are given in Table 2.2 [12, 55, 56].

Using weights during cooperation improves the network performance; however, on the node level the cooperating process is not beneficial for some of the nodes as their performance will degrade [91]. This problem can be mitigated and the network performance can be further improved by adding additional information to the weights in the form of a secondary parameter [54]. The secondary parameter should be selected according to the application and to the situation to reflect the quality of the estimate of the node. This way, higher weights are assigned to estimates from nodes in better conditions and lower weights to nodes which are in worse conditions.

In radio networks and particulalry in the CR spectrum sensing application, the noise power $\sigma_{v,k}^2$ or the signal-to-noise ratio of the received primary user signal can be used as a secondary parameter [Paper A]. These secondary parameters are not usually known beforehand and have to be estimated. Different secondary parameter estimation methods are given in these examples [66, 92]. Examples of

using additional information in the calculation of combination rules can be found in [12]. In [Paper A] the author of this thesis explored using the MDL subspace algorithm to estimate the SNR values at the different nodes. Two examples based on the noise powers at different nodes and the relative degree rule (2.6) and relative degree-variance rule (2.7) are given as:

$$\alpha_{l,k} = \begin{cases} \frac{\sigma_{v,l}^{-2}}{\sum\limits_{l\in\mathbb{N}_k} \sigma_{v,l}^{-2}}, \text{if } l \in \mathbb{N}_k \\ 0, \text{if } l \notin \mathbb{N}_k, \end{cases} \tag{2.6}$$

$$\alpha_{l,k} = \begin{cases} \frac{n_l\sigma_{v,l}^{-2}}{\sum\limits_{l\in\mathbb{N}_k} n_l\sigma_{v,l}^{-2}}, \text{if } l \in \mathbb{N}_k \\ 0, \text{if } l \notin \mathbb{N}_k, \end{cases} \tag{2.7}$$

where $\alpha_{l,k}$ is the weight calculated at node k for neighbouring node l, $\sigma_{v,k}^2$ is the noise power at the node k, $\mathbb{N}_k$ is the neighbourhood of node k and n_k is the size of the neighbourhood.

Since the weights are directly calculated based on the secondary parameter, the estimation accuracy contributes into the accuracy of the weights. If the estimation is inaccurate the performance of the network degrades [Paper C]. Therefore, acquiring reliable or accurate combination weights might be problematic if the secondary parameter estimation is unreliable.

2.3.5. Distributed leader selection

Leader selection is a method that has been used widely in different applications in networks [93–95]. The basic concept is that a node in the network is selected

Table 2.2 Different combinations rules for calculating weights [54, 55, 83]

Name	Rule
Uniform	$\frac{1}{n_k}$
Laplacian	$\frac{1}{n_{max}}$
Maximum Degree	$\frac{1}{N}$
Metropolis	$\frac{1}{\max(n_k,n_l)}$
Relative degree	$\frac{n_l}{\sum\limits_{l\in\mathbb{N}_k} n_k}$
Relative degree-variance	$\frac{\sigma_{v,l}^{-2}}{\sum\limits_{l\in\mathbb{N}_k} \sigma_{v,l}^{-2}}$

to fulfill some task in the network. The other nodes will either listen to this node or follow its lead, becoming follower nodes [96]. There are several works on algorithms employing leader selection in networks that work in a distributed manner; however, they are not fully distributed and they rely on different constraints or *a priori* knowledge [93, 95–100]. For example, in [95, 99, 100] the leader is selected from predefined suitable nodes. These predefined nodes have additional information or better sensors compared to the followers in the network. There are also works [97] that require the nodes to know the topology or to be able to estimate the paths in the network that lead to the leader node. These works can not be directly compared to the algorithms previously outlined in Subsections 2.3.2, 2.3.3 as they have prerequisites and conditions that make them more susceptible to failures and they are unable to learn and adapt to changing conditions.

Therefore, the author developed an algorithm for leader selection that is able to select a leader node in the network in a fully distributed manner and learn and adapt in real-time to streaming information. The proposed algorithm is comparable with cooperating algorithms present in previous Subsections 2.3.1, 2.3.2, 2.3.3. The DLS algorithm has been proposed in our paper [Paper B] and further analysed in [Paper C]. The LMS-based distributed leader selection algorithm is given in Algorithm 6.

Algorithm 6 Distributed leader selection

for each time instant $i > 0$:
  each node $k = 1, 2, ..., K$ performs the update:
  $\mathbf{w}_{k,i} = \mathbf{w}_{k,i-1} + \mu_k \mathbf{u}_{k,i}^{*}[d_k(i) - \mathbf{u}_{k,i}\mathbf{w}_{k,i-1}]$
  if $\alpha_k(i-1) \geq \max_{k \in \mathbb{N}_k}(\alpha_k(i-1))$
    $\mathbf{e}_{k,i} = \mathbf{w}_{k,i}$
    $\alpha_k(i) = (1-\mu_k)\alpha_k(i-1) + \mu_k \sigma_{v,k}^{-2}$
  else
    $c_k = \underset{k \in \mathbb{N}_k}{\operatorname{argmax}}(\alpha_k(i-1))$
    $\mathbf{e}_{k,i} = \mathbf{e}_{c_k,i}$
    $\alpha_k(i) = (1-\mu_k)\alpha_{c_k}(i-1) + \mu_k \sigma_{v,k}^{-2}$
  end
end

The node k exchanges information with other nodes from its neighbourhood $\mathbb{N}_k$. The estimate $\mathbf{e}_{k,i}$ and the corresponding weight $\alpha_k(i)$ that is assigned to the estimate are shared between the nodes. In contrast to the other cooperating algorithms the estimates are weighed instead of the communication paths or nodes. The nodes compare and choose the estimate that has the highest weight in their neighbourhood $\mathbb{N}_k$ and if the estimate belongs to the node k it becomes the leader node. Otherwise the node becomes a follower node and starts to

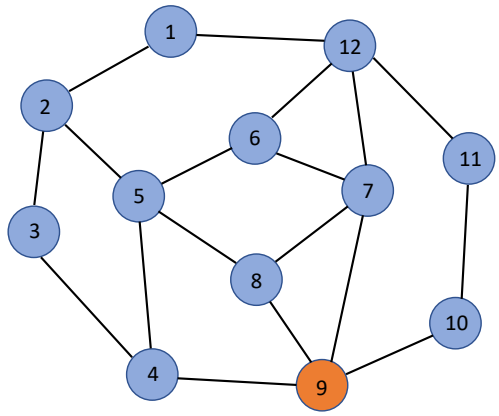


Figure 2.6 The leader in the network has been selected as node 9 and all the other nodes are following its lead and the network attains the performance of node 9.

follow their best neighbour c_k. The node c_k is the local leader for node k in the neighbourhood $\mathbb{N}_k$ and the estimates from the leader node are distributed along with the corresponding weight. In this manner local leaders emerge in different neighbourhoods and after certain number of iterations a global leader is chosen. A secondary parameter is needed to compute the weights similarly to the combination rules that are used in diffusion algorithms to combine the estimates. An example of a network with a leader selected can be seen in Figure 2.6.

The proposed DLS algorithm, while not always able to outperform the diffusion algorithms, has several interesting properties [Paper C]. The performance of the algorithm is more easily deducted as the performance of the network is the same as that of the best performing node in the network [Paper B, Paper C]. The algorithm is simpler and more robust in terms of the secondary parameter estimation accuracy [Paper C]. The algorithm is less complex in terms of computational complexity as well in terms of operations required, especially in applications with longer filter lengths and more densely connected networks [Paper D]. Furthermore, an energy-efficient implementation of the algorithm is possible while reducing the complexity and retaining the performance of the DLS algorithm [Paper D].

2.4. Chapter summary

In this chapter the algorithms for learning and adaptation over networks in the signal processing field were presented. The weight calculations and the author's contributions were included. The summary of the chapter is given as:

1. Centralized solutions and distributed solutions are able to obtain identical performance, but distributed solutions are more robust and adaptive. Therefore, in this thesis the focus is on distributed algorithms.

2. Different models for distributed estimation were introduced – incremental, consensus, diffusion and leader selection. Incremental and consensus solutions are outperformed by the diffusion algorithm. Incremental solutions suffer from NP-hard problem and consensus strategies have stability issues. The current state-of-the-art algorithm is the diffusion algorithm which is used in this thesis as the reference to evaluate the performance of the DLS algorithm.

3. Distributed algorithms as well as centralized solutions can obtain better performance by using secondary parameter estimation to estimate the quality of estimates at different nodes and use weights to weigh the estimates accordingly. Obtaining accurate weight estimates is important to ensure good performance of the diffusion algorithm, which might be problematic in practical applications. In [Paper A] the SNR estimated weights based on MDL subspace algorithm are introduced as a possible method.

4. The DLS algorithm is able to outperform the non-cooperating network as well as the diffusion network using equal weights. In [Paper B] the DLS algorithm is introduced.

5. DLS algorithm is able to obtain similar performance in certain conditions, but overall is outperformed by the diffusion algorithm with optimal weights. In [Paper C] the performance of the DLS algorithm is compared to the diffusion algorithm.

6. The DLS algorithm is more robust to weight inaccuracies and is a less complex algorithm in terms of computations and required energy. In [Paper D] The DLS algorithm is modified to reduce to energy consumption and complexity.

3. OVERVIEW OF PUBLICATIONS

This chapter gives an overview of the publications; the work presented therein provides answers to the research questions raised in Section 1.1.

A In this paper, the performance of an adaptive network of nodes that use the signal to noise ratio (SNR) estimates of the received signal as the basis to improve the performance of the network is analysed. The signal to noise estimates are used to calculate the diffusion algorithms' weights. The communication is weighed according to these estimates, giving nodes with lower SNR smaller weights and larger weights to nodes with higher SNR. The SNR estimates are acquired by using the MDL subspace algorithm. The signal and noise subspace can be estimated by using the MDL information criterion on the calculated eigenvalues of the sample covariance matrix of the received signal. The algorithm uses ATC diffusion strategy and it is shown that the proposed SNR weighted diffusion algorithm is able to outperform the non-cooperating network as well as the equally weighed diffusion algorithm.

B In this paper, a simple and robust algorithm for distributed leader selection is proposed. The leader selection algorithm is able to work in a fully distributed manner. The leader selection is based on a secondary parameter and the performance of the network is not heavily connected to the estimate accuracy. The secondary parameter calculation is based on the SNR estimate of the received signal which is estimated using the MDL subspace algorithm. The proposed algorithm is able to outperform both the non-cooperating network and the diffusion network with equally weighed communication.

C This work investigates the distributed leader selection algorithm in an LMS-based adaptive network. The algorithm is modified to use an LMS adaptive filter. The analytical MSD performance of the algorithm is derived. Simulation results are compared to that of diffusion algorithms using relative variance rule and relative degree variance rule. The analytical results show that the diffusion algorithm outperforms the proposed distributed leader selection

algorithm in most cases; however, in the presence of a significantly better node, the performance is comparable that of the diffusion algorithms. Moreover, the simulation results show that in the case of a significantly better node in the network, the distributed leader selection is able to outperform the diffusion algorithm.

D In this paper, the distributed leader selection algorithm and the diffusion algorithm are analysed from the point of view of computational complexity and energy-efficiency. The analysis shows that the distributed leader selection algorithm is less complex and more energy-efficient than the diffusion algorithm. Furthermore, in applications with densely connected networks and long filter lengths, the computational operations required by the algorithms makes the distributed leader selection algorithm more suitable. However, factoring in the radio communication energy consumption, the differences between the algorithms are marginal. A novel energy-efficient distributed leader selection algorithm is introduced that retains the performance of the algorithm and reduces both the radio communication and the computational energy consumptions. In the simulations the algorithms are mapped to widely used wireless sensor network hardware architectures (MSP430 and RSL10). The proposed algorithm is able to decrease the energy consumption of the network by $32 - 53\%$ and can extend its lifetime by $14 - 46\%$ in comparison to the diffusion algorithm and the distributed leader selection algorithm.

4. CONCLUSIONS

This PhD thesis focused on the analysis and development of algorithms for distributed estimation and adaptation and learning over networks. The thesis was divided into two parts. The first part introduced the motivation, background, literature and the theoretical basis as well as the contributions of the author and the contributions of the publications. The second part included the publications that the thesis consists of and which the conclusions and claims are based on. The thesis proposed a blind secondary parameter estimation based on MDL subspace algorithm which allows to estimate the SNR of the primary signal, incorporate it into the diffusion algorithm weight calculation, and improve its performance over the equally weighed diffusion algorithm. During the thesis a novel distributed leader selection algorithm was proposed and its performance was analysed and compared to the state-of-the-art diffusion algorithm's performance. The distributed leader selection algorithm was modified to improve the algorithm's EE and to enable the network to extend its lifetime. Research questions were formulated in Section 1.1 and the answers are given below:

1. Can we estimate the weights for the diffusion algorithm in a blind manner with no *a priori* information?
   Answer: Yes, by using the MDL subspace algorithm we are able to estimate the SNR at different nodes and base the weight calculation on this. This was investigated in [Paper A].

2. Is it possible to implement a fully distributed leader selection algorithm?
   Answer: Yes, by locally choosing the best performing node in the neighbourhood and sharing its estimate with its corresponding weight, the network will distribute the best performing node's information across the network. The DLS algorithm was proposed in [Paper B].

3. What is the performance of the DLS algorithm and how does it compare to the state-of-the-art solutions?
   Answer: The DLS algorithm is able to outperform the non-cooperating and equally weighed diffusion algorithm. However, the theoretical performance shows that the diffusion algorithm with optimal weights will outperform the DLS algorithm, and the DLS algorithm will achieve similar performance if there is a node in the network with significantly better conditions. The

DLS algorithm is more robust to weight inaccuracies and is able to achieve similar performance or even outperform the diffusion algorithm. The comparison of the diffusion and DLS algorithm is given in [Paper C].

4. What are the computational complexities of the DLS algorithm and the diffusion algorithm?
   Answer: The diffusion algorithm is more complex than the DLS algorithm. The complexity of the diffusion algorithm increases at a larger rate with longer filter lengths and larger neighbourhoods compared to the DLS algorithm. The analysis and the formulation were done in [Paper D].

5. What is the energy consumption of the DLS algorithm and the diffusion algorithm?
   Answer: The DLS algorithms consumes less computational energy than the diffusion algorithm. The communication energy consumed by both algorithms is identical. These results were presented in [Paper D].

6. How can we reduce the energy consumption for the DLS algorithm?
   Answer: Comparison of the computational and the communication energy shows that the computational energy consumption is marginal in comparison to the communication energy consumption. Therefore, the largest gain is obtained by reducing the communication energy. The communication energy is reduced by identifying the redundant connections in the topology that can be disconnected to save energy without losing in estimation performance. In this thesis the energy-efficient DLS algorithm (EEDLS) was proposed in [Paper D].

Based on the publications and the research questions, the main claims of thesis are as follows:

- A novel method for secondary parameter calculation in the diffusion algorithm based on the SNR estimates of the MDL subspace algorithm which is able to improve the performance of the diffusion algorithm using equal weights.
- A novel fully distributed leader selection algorithm for distributed estimation.
  - The DLS algorithm is more robust to secondary parameter estimation than the diffusion algorithm.
  - The DLS is able to perform similarly or even outperform the diffusion algorithm in the presence of a node that is in a more favourable condition.
  - The DLS is less complex and consumes less computational energy than the diffusion algorithm.

- A novel energy-efficient distributed leader selection algorithm that is able to decrease the energy consumption of the network by $\mathbf{32 - 53}\%$ and can extend the network lifetime by $\mathbf{14 - 46}\%$.

4.1. Future work

The current work on the DLS algorithm allows several directions to extend the work. First, the DLS algorithm could be analysed in a dynamic topology and under link failures, i.e., to what extent would these affect the leader selection process and the learning and adaptation performance in the network?

Secondly, the performance of the DLS algorithm and the diffusion algorithm performance with imperfect secondary parameter estimation could be investigated more thoroughly. If some or all the nodes have access to inaccurate information about the quality of their estimates, how much would this affect the MSD performance of the algorithms in comparison?

Thirdly, the energy-efficient distributed leader selection (EEDLS) algorithm could be further extended to adapt to the situation and switch from power saving mode to normal mode according to the changing conditions in the network. For example, what is the impact of accuracy of selecting the leader node if the topology in the network has been decreased in the power saving mode?

REFERENCES

[1] A. Barrat, M. Barthelemy, and A. Vespignani, *Dynamical processes on complex networks*. Cambridge university press, 2008.

[2] T. G. Lewis, *Network science: Theory and applications*. John Wiley & Sons, 2011.

[3] D. Easley and J. Kleinberg, *Networks, crowds, and markets: Reasoning about a highly connected world*. Cambridge University Press, 2010.

[4] A.-L. Barabasi, *Linked: How everything is connected to everything else and what it means*. Plume, 2003.

[5] M. Newman, *Networks: an introduction*. Oxford university press, 2010.

[6] N. A. Christakis and J. H. Fowler, *Connected: The surprising power of our social networks and how they shape our lives*. Little, Brown, 2009.

[7] M. O. Jackson, *Social and economic networks*. Princeton university press, 2010.

[8] J. Surowiecki, M. P. Silverman *et al.*, "The wisdom of crowds," *American Journal of Physics*, vol. 75, no. 2, pp. 190–192, 2007.

[9] Y. Shan, *Applications of complex adaptive systems*. IGI Global, 2008.

[10] P. Angelov, D. P. Filev, and N. Kasabov, *Evolving intelligent systems: methodology and applications*. John Wiley & Sons, 2010, vol. 12.

[11] W. Brenner, R. Zarnekow, and H. Wittig, *Intelligent software agents: foundations and applications*. Springer Science & Business Media, 2012.

[12] A. H. Sayed *et al.*, "Adaptation, learning, and optimization over networks," *Foundations and Trends® in Machine Learning*, vol. 7, no. 4-5, pp. 311–801, 2014.

[13] F. L. Lewis, H. Zhang, K. Hengster-Movric, and A. Das, *Cooperative control of multi-agent systems: optimal and adaptive design approaches*. Springer Science & Business Media, 2013.

[14] J. Shamma, *Cooperative control of distributed multi-agent systems*. John Wiley & Sons, 2008.

[15] Z. Li and Z. Duan, *Cooperative control of multi-agent systems: a consensus region approach*. CRC Press, 2014.

[16] M. Yokoo, *Distributed constraint satisfaction: foundations of cooperation in multi-agent systems*. Springer Science & Business Media, 2012.

[17] C. Prehofer and C. Bettstetter, "Self-organization in communication networks: principles and design paradigms," *IEEE Communications magazine*, vol. 43, no. 7, pp. 78–85, 2005.

[18] A.-L. Barabasi and Z. N. Oltvai, "Network biology: understanding the cell's functional organization," *Nature reviews genetics*, vol. 5, no. 2, p. 101, 2004.

[19] B. H. Junker and F. Schreiber, *Analysis of biological networks*. John Wiley & Sons, 2011, vol. 2.

[20] F. Kepes, *Biological networks*. World Scientific, 2007, vol. 3.

[21] B. L. Partridge, "The structure and function of fish schools," *Scientific american*, vol. 246, no. 6, pp. 114–123, 1982.

[22] M. Milinski and R. Heller, "Influence of a predator on the optimal foraging behaviour of sticklebacks (gasterosteus aculeatus l.)," *Nature*, vol. 275, no. 5681, pp. 642–644, 1978.

[23] W. D. Hamilton, "Geometry for the selfish herd," *Journal of theoretical Biology*, vol. 31, no. 2, pp. 295–311, 1971.

[24] M. Beekman, R. L. Fathke, and T. D. Seeley, "How does an informed minority of scouts guide a honeybee swarm as it flies to its new home?" *Animal Behaviour*, vol. 71, no. 1, pp. 161–171, 2006.

[25] A. Avitabile, R. Morse, and R. Boch, "Swarming honey bees guided by pheromones," *Annals of the Entomological Society of America*, vol. 68, no. 6, pp. 1079–1082, 1975.

[26] T. D. Seeley, R. A. Morse, and P. K. Visscher, "The natural history of the flight of honey bee swarms," *Psyche: A Journal of Entomology*, vol. 86, no. 2-3, pp. 103–113, 1979.

[27] D. Hummel, "Aerodynamic aspects of formation flight in birds," *Journal of theoretical biology*, vol. 104, no. 3, pp. 321–347, 1983.

[28] F. H. Heppner, "Avian flight formations," *Bird-Banding*, vol. 45, no. 2, pp. 160–169, 1974.

[29] F. S. Cattivelli and A. H. Sayed, "Modeling bird flight formations using diffusion adaptation," *IEEE Transactions on Signal Processing*, vol. 59, no. 5, pp. 2038–2051, 2011.

[30] J. Li and A. H. Sayed, "Modeling bee swarming behavior through diffusion adaptation with asymmetric information sharing," *EURASIP Journal on Advances in Signal Processing*, vol. 2012, no. 1, p. 18, 2012.

[31] S.-Y. Tu and A. H. Sayed, "Cooperative prey herding based on diffusion adaptation," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*. IEEE, 2011, pp. 3752–3755.

[32] G.-Z. Yang and G. Yang, *Body sensor networks*. Springer, 2006, vol. 1.

[33] M. Farooq and G. A. Di Caro, "Routing protocols for next-generation networks inspired by collective behaviors of insect societies: An overview," in *Swarm intelligence*. Springer, 2008, pp. 101–160.

[34] A. Zengin, S. Tuncel *et al.*, "A survey on swarm intelligence based routing protocols in wireless sensor networks," *International Journal of Physical Sciences*, vol. 5, no. 14, pp. 2118–2126, 2010.

[35] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, "Next century challenges: Scalable coordination in sensor networks," in *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*. ACM, 1999, pp. 263–270.

[36] F. Dressler, *Self-organization in sensor and actor networks*. John Wiley & Sons, 2008.

[37] F. Dressler and I. Carreras, *Advances in Biologically Inspired Information Systems*. Springer, 2007.

[38] R. W. Thomas, D. H. Friend, L. A. Dasilva, and A. B. Mackenzie, "Cognitive networks: adaptation and learning to achieve end-to-end performance objectives," *IEEE Communications Magazine*, vol. 44, no. 12, pp. 51–57, 2006.

[39] D. Estrin, L. Girod, G. Pottie, and M. Srivastava, "Instrumenting the world with wireless sensor networks," in *icassp*. IEEE, 2001, pp. 2033–2036.

[40] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE communications magazine*, vol. 40, no. 8, pp. 102–114, 2002.

[41] D. Culler, D. Estrin, and M. Srivastava, "Guest editors' introduction: Overview of sensor networks," *Computer*, vol. 37, no. 8, pp. 41–49, 2004.

[42] B. Bangerter, S. Talwar, R. Arefi, and K. Stewart, "Networks and devices for the 5g era," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 90–96, 2014.

[43] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," *Future generation computer systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[44] Z. Sheng, C. Mahapatra, C. Zhu, and V. C. Leung, "Recent advances in industrial wireless sensor networks toward efficient management in IoT," *IEEE Access*, vol. 3, pp. 622–637, 2015.

[45] R. Mahapatra, Y. Nijsure, G. Kaddoum, N. U. Hassan, and C. Yuen, "Energy efficiency tradeoff mechanism towards wireless green communication: A survey." *IEEE Communications Surveys and Tutorials*, vol. 18, no. 1, pp. 686–705, 2016.

[46] K. Wang, Y. Wang, Y. Sun, S. Guo, and J. Wu, "Green industrial internet of things architecture: an energy-efficient perspective," *IEEE Communications Magazine*, vol. 54, no. 12, pp. 48–54, 2016.

[47] T. Yücek and H. Arslan, "A survey of spectrum sensing algorithms for cognitive radio applications," *Communications Surveys & Tutorials, IEEE*, vol. 11, no. 1, pp. 116–130, 2009.

[48] J. Mitola III and G. Q. Maguire Jr, "Cognitive radio: making software radios more personal," *Personal Communications, IEEE*, vol. 6, no. 4, pp. 13–18, 1999.

[49] F. S. Cattivelli and A. H. Sayed, "Distributed detection over adaptive networks using diffusion adaptation," *IEEE Transactions on Signal Processing*, vol. 59, no. 5, pp. 1917–1932, 2011.

[50] A. Ainomäe, T. Trump, and M. Bengtsson, "Distributed diffusion lms based energy detection," in *Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2014 6th International Congress on*. IEEE, 2014, pp. 176–183.

[51] A. Ainomäe, M. Bengtsson, and T. Trump, "Distributed largest eigenvalue-based spectrum sensing using diffusion lms," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 4, no. 2, pp. 362–377, 2018.

[52] A. Hajihoseini and S. A. Ghorashi, "Distributed spectrum sensing for cognitive radio sensor networks using diffusion adaptation," *IEEE Sensors Letters*, vol. 1, no. 5, pp. 1–4, 2017.

[53] C. G. Lopes and A. H. Sayed, “Diffusion least-mean squares over adaptive networks: Formulation and performance analysis,” *IEEE Transactions on Signal Processing*, vol. 56, no. 7, pp. 3122–3136, 2008.

[54] F. S. Cattivelli and A. H. Sayed, “Diffusion LMS strategies for distributed estimation,” *IEEE Transactions on Signal Processing*, vol. 58, no. 3, pp. 1035–1048, 2010.

[55] X. Zhao and A. H. Sayed, “Performance limits for distributed estimation over LMS adaptive networks,” *Signal Processing, IEEE Transactions on*, vol. 60, no. 10, pp. 5107–5124, 2012.

[56] S.-Y. Tu and A. H. Sayed, “Diffusion strategies outperform consensus strategies for distributed estimation over adaptive networks,” *IEEE Transactions on Signal Processing*, vol. 60, no. 12, pp. 6217–6234, 2012.

[57] A. H. Sayed, S.-Y. Tu, J. Chen, X. Zhao, and Z. J. Towfic, “Diffusion strategies for adaptation and learning over networks: an examination of distributed strategies and network behavior,” *Signal Processing Magazine, IEEE*, vol. 30, no. 3, pp. 155–171, 2013.

[58] C. G. Lopes and A. H. Sayed, “Diffusion adaptive networks with changing topologies,” in *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*. IEEE, 2008, pp. 3285–3288.

[59] S. Xu, R. C. de Lamare, and H. V. Poor, “Adaptive link selection algorithms for distributed estimation,” *EURASIP Journal on Advances in Signal Processing*, vol. 2015, no. 1, p. 86, 2015.

[60] X. Zhao, S.-Y. Tu, and A. H. Sayed, “Diffusion adaptation over networks under imperfect information exchange and non-stationary data,” *IEEE Transactions on Signal Processing*, vol. 60, no. 7, pp. 3460–3475, 2012.

[61] A. Khalili, M. A. Tinati, A. Rastegarnia, and J. A. Chambers, “Steady-state analysis of diffusion LMS adaptive networks with noisy links,” *IEEE Transactions on Signal Processing*, vol. 60, no. 2, pp. 974–979, 2012.

[62] X. Zhao and A. H. Sayed, “Single-link diffusion strategies over adaptive networks,” in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, 2012, pp. 3749–3752.

[63] Ø. L. Rørtveit, J. H. Husøy, and A. H. Sayed, “Diffusion LMS with communication constraints,” in *Signals, Systems and Computers (ASILOMAR), 2010 Conference Record of the Forty Fourth Asilomar Conference on*. IEEE, 2010, pp. 1645–1649.

[64] P. K. Varshney, *Distributed detection and data fusion*. Springer Science & Business Media, 2012.

[65] F. S. Cattivelli, C. G. Lopes, and A. H. Sayed, "Diffusion recursive least-squares for distributed estimation over adaptive networks," *IEEE Transactions on Signal Processing*, vol. 56, no. 5, pp. 1865–1877, 2008.

[66] N. Takahashi, I. Yamada, and A. H. Sayed, "Diffusion least-mean squares with adaptive combiners: Formulation and performance analysis," *IEEE Transactions on Signal Processing*, vol. 58, no. 9, pp. 4795–4810, 2010.

[67] A. Abdelgawad and M. Bayoumi, "Data fusion in wsn," in *Resource-aware data fusion algorithms for wireless sensor networks*. Springer, 2012, pp. 17–35.

[68] D. Li, K. D. Wong, Y. H. Hu, and A. M. Sayeed, "Detection, classification, and tracking of targets," *IEEE signal processing magazine*, vol. 19, no. 2, pp. 17–29, 2002.

[69] F. H. Fitzek and M. D. Katz, *Cooperation in wireless networks: principles and applications*. Springer, 2006.

[70] B. Ying and A. H. Sayed, "Information exchange and learning dynamics over weakly connected adaptive networks," *IEEE Transactions on Information Theory*, vol. 62, no. 3, pp. 1396–1414, 2016.

[71] D. P. Bertsekas, "A new class of incremental gradient methods for least squares problems," *SIAM Journal on Optimization*, vol. 7, no. 4, pp. 913–926, 1997.

[72] A. Nedic and D. P. Bertsekas, "Incremental subgradient methods for nondifferentiable optimization," *SIAM Journal on Optimization*, vol. 12, no. 1, pp. 109–138, 2001.

[73] M. G. Rabbat and R. D. Nowak, "Quantized incremental algorithms for distributed optimization," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 4, pp. 798–808, 2005.

[74] C. G. Lopes and A. H. Sayed, "Incremental adaptive strategies over distributed networks," *IEEE Transactions on Signal Processing*, vol. 55, no. 8, pp. 4064–4077, 2007.

[75] D. Blatt, A. O. Hero, and H. Gauchman, "A convergent incremental gradient method with a constant step size," *SIAM Journal on Optimization*, vol. 18, no. 1, pp. 29–51, 2007.

[76] F. S. Cattivelli and A. H. Sayed, "Analysis of spatial and incremental lms processing for distributed estimation," *IEEE Transactions on Signal Processing*, vol. 59, no. 4, pp. 1465–1480, 2011.

[77] E. S. H. Neto and Á. R. De Pierro, "Incremental subgradients for constrained convex optimization: a unified framework and new methods," *SIAM Journal on Optimization*, vol. 20, no. 3, pp. 1547–1572, 2009.

[78] B. Johansson, M. Rabi, and M. Johansson, "A randomized incremental subgradient method for distributed optimization in networked systems," *SIAM Journal on Optimization*, vol. 20, no. 3, pp. 1157–1170, 2009.

[79] L. Li, J. A. Chambers, C. G. Lopes, and A. H. Sayed, "Distributed estimation over an adaptive incremental network based on the affine projection algorithm," *IEEE Transactions on Signal Processing*, vol. 58, no. 1, pp. 151–164, 2010.

[80] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of computer computations*. Springer, 1972, pp. 85–103.

[81] J. Chen and A. H. Sayed, "Diffusion adaptation strategies for distributed optimization and learning over networks," *Signal Processing, IEEE Transactions on*, vol. 60, no. 8, pp. 4289–4305, 2012.

[82] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, 2009.

[83] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *Proceedings of the 4th international symposium on Information processing in sensor networks*. IEEE Press, 2005, p. 9.

[84] A. H. Sayed and C. G. Lopes, "Distributed processing over adaptive networks," in *Signal Processing and Its Applications, 2007. ISSPA 2007. 9th International Symposium on*. IEEE, 2007, pp. 1–3.

[85] F. Cattivelli and A. H. Sayed, "Diffusion lms-based distributed detection over adaptive networks," in *Signals, Systems and Computers, 2009 Conference Record of the Forty-Third Asilomar Conference on*. IEEE, 2009, pp. 171–175.

[86] S. Haykin and K. R. Liu, *Handbook on array processing and sensor networks*. John Wiley & Sons, 2010, vol. 63.

[87] J. Ni, "Diffusion sign subband adaptive filtering algorithm for distributed estimation," *IEEE Signal Processing Letters*, vol. 22, no. 11, pp. 2029–2033, 2015.

[88] S.-H. Yim, H.-S. Lee, and W.-J. Song, "A proportionate diffusion lms algorithm for sparse distributed estimation," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 62, no. 10, pp. 992–996, 2015.

[89] J.-W. Lee, S.-E. Kim, and W.-J. Song, "Data-selective diffusion lms for reducing communication overhead," *Signal Processing*, vol. 113, pp. 211–217, 2015.

[90] T. Rault, A. Bouabdallah, and Y. Challal, "Energy efficiency in wireless sensor networks: A top-down survey," *Computer Networks*, vol. 67, pp. 104–122, 2014.

[91] A. H. Sayed, "Diffusion adaptation over networks," in *Academic Press Library in Signal Processing*. Elsevier, 2014, vol. 3, pp. 323–453.

[92] S.-Y. Tu and A. H. Sayed, "Optimal combination rules for adaptation and learning over networks," in *2011 4th IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*. IEEE, 2011, pp. 317–320.

[93] Y. Hong, G. Chen, and L. Bushnell, "Distributed observers design for leader-following control of multi-agent networks," *Automatica*, vol. 44, no. 3, pp. 846–850, 2008.

[94] F. Sen, Q. Bing, and T. Liangrui, "An improved energy-efficient pegasis-based protocol in wireless sensor networks," in *Fuzzy Systems and Knowledge Discovery (FSKD), 2011 Eighth International Conference On*, vol. 4. IEEE, 2011, pp. 2230–2233.

[95] F. Lin, M. Fardad, and M. R. Jovanovic, "Algorithms for leader selection in stochastically forced consensus networks," *IEEE Transactions on Automatic Control*, vol. 59, no. 7, pp. 1789–1802, 2014.

[96] S. Patterson and B. Bamieh, "Leader selection for optimal network coherence," in *2010 49th IEEE Conference on Decision and Control (CDC)*. IEEE, 2010, pp. 2692–2697.

[97] I. Shames, A. M. Teixeira, H. Sandberg, and K. H. Johansson, "Distributed leader selection without direct inter-agent communication," *IFAC Proceedings Volumes*, vol. 43, no. 19, pp. 221–226, 2010.

[98] S. Pequito, V. Preciado, and G. J. Pappas, "Distributed leader selection," in *2015 IEEE 54th Annual Conference on Decision and Control (CDC)*. IEEE, 2015, pp. 962–967.

[99] K. Fitch and N. E. Leonard, "Information centrality and optimal leader selection in noisy networks," in *2013 IEEE 52nd Annual Conference on Decision and Control (CDC)*. IEEE, 2013, pp. 7510–7515.

[100] F. Lin, M. Fardad, and M. R. Jovanović, "Algorithms for leader selection in large dynamical networks: Noise-corrupted leaders," in *2011 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*. IEEE, 2011, pp. 2932–2937.

ACKNOWLEDGEMENTS

I would like to thank the former Department of Radio- and Communication Engineering and Thomas Johann Seebeck Department of Electronics at Tallinn University of Technology for giving me the opportunity to pursue a PhD degree. I would like to thank my supervisors, Professor Tõnu Trump who was my supervisor for the first two years and Professor Muhammed Mahtab Alam and Professor Yannick Le Moullec my current supervisors who supervised me for the past two years. I would like to thank my colleagues at Tallinn University of Technology and in particular Maksim Butsenko, Ahti Ainomäe, Egon Astra, Julia Berdnikova and Hip Kõiv for all the help and support they provided me. I thank my family and friends for all the support and encouragement I received during my studies.

I express my gratitude to the following entities for their financial support during my PhD studies:

- This work has received funding from the European Union′s Horizon 2020 research and innovation program under grant agreement No 668995 and partly supported by European Union Regional Development Fund in the framework of the Tallinn University of Technology Development Program 2016-2022. This material reflects only the authors view and the EC Research Executive Agency is not responsible for any use that may be made of the information it contains;
- ICT Doctoral School at Tallinn University of Technology;
- Information Technology Foundation for Education;
- DoRa Programme.

ABSTRACT
Algorithms for Learning and Adaptation Over Networks – Distributed Leader Selection

Learning and adaptation over networks is a rapidly evolving topic with many possible applications and fields. There are several unanswered and unresearched aspects of these algorithms and their applications. In particular, analysing the performance and developing algorithms for distributed estimation are essential for future smart and self-organizing networks. Indeed, enabling complex and sophisticated behaviour through cooperation of simpler units in the network allows to accomplish more demanding tasks that are unattainable to single units and current solutions.

Estimating from different sensors and cooperating to achieve better performance is a challenging task when taking into account the limitations and constraints presented by applications. The algorithms should enable the networks to learn and adapt to different situations under constraints, such as limited bandwidth, limited battery capacity, physical and communication restrictions, security etc. In this thesis the motivation for and the background to the learning and adaptation are presented and an overview of different methods for distributed estimation as well as the author's contributions to the existing work are given.

This PhD thesis proposes 1) an improvement to the existing diffusion algorithm weight calculation as well as 2) a novel algorithm for distributed leader selection (DLS). The method using MDL (minimum description length) subspace algorithm to estimate the SNR (signal to noise ratio) of the estimated signal allows the weight calculation to incorporate the values and improve the performance of the diffusion algorithm in comparison to the equally weighed diffusion algorithm.

The proposed DLS algorithm is able to select the best performing node as the leader node in a fully distributed manner and achieve its performance across the network. The algorithm outperforms both the equally weighed diffusion algorithm and the non-cooperating network. The theoretical and simulation results show that the DLS algorithm is also able to attain similar performance than that of the diffusion algorithm using optimal weights under certain conditions and overall is less complex and more robust compared to the diffusion algorithm.

The thesis also analyses the energy consumption and computational complexity of the diffusion algorithm and the DLS algorithm. The DLS algorithm is modified to improve its energy efficiency. The proposed energy-efficient distributed leader selection algorithm is able to reduce the energy consumption of the network by $32 - 53\%$ and extend its lifetime by $14 - 46\%$.

KOKKUVÕTE
Õppimisalgoritmid hajutatud võrkude tarbeks – juhtsõlme hajus valimine

Võrkudes toimuv õppimine ja adapteerumine on kiirelt arenev teadusala, mille tulemused on rakendatavad paljudes eri valdkondades. Samas ei ole õppimisalgoritmid ning nende rakendusted tänaseni piisavalt läbi uuritud. Näiteks vajab jätkuvat tööd parameetrite hindamine hajusas süsteemis. Hajusad õppimisalgoritmid on hädavajalikud tuleviku tarkade ja iseorganiseeruvate võrkude loomisel, sest nad võimaldavad luua komplekssseid ja intelligentseid käitumismustreid lihtsamate võrguelementide abil. Lisaks võimaldavad iseorganiseeruvad võrgud lahendada ka ülesandeid, mis poleks võimalikud üksikutel võrguelementidel.

Eri sensorite sisendite kombineerimine ning koostöö võrguelementide vahel osutub keerulisemaks ülesandeks, kui arvesse võtta praktilised piirangud. Välja töötatud algoritmid peaksid võimaldama võrkudel õppida ning adapteeruda erinevates olukordades ning piirangute olemasolul. Piirangd võivad tuleneda näiteks etteantud ribalaiusest, energia tarbimisest, kommunikatsioonist või turvalisusest. Lisaks autori panusele on töös esitatud ülevaade õppimise ja adapteerumise valdkonna taustast ja ülevaade hajutatud hindamise algoritmidest.

Doktoritöös esitletakse täiendust hetkel eksisteeriva difusiooni algoritmi kaalude arvutamisele ning ka uudset algoritmi juhtsõlme hajusalt valimiseks. MDL alamruumi meetodil põhinev SNR hindamise algoritm võimaldab huvipakkuva signaali signaali ja müra suhte väärtused ühildada kaalude arvutamisega ja parandada difusiooni algoritmi toimimist võrreldes ühtlaste kaaludega difusiooni algoritmiga.

Doktoritöös välja pakutud algoritm suudab leida kogu võrgu parima parameetri hinnangu saavutava võrguelemendi täiesti hajusal viisil. Algoritm saavutab paremad tulemused võrreldes koostööd mitte tegevatest võrguelementidest koosneva võrguga ning ühtlaste kaaludega difusioonivõrguga. Teoreetilised tulemused näitavad, et välja töötatud algoritmi tulemus on teatud tingimustel ligilähedane optimaalsete kaaludega difusiooni algoritmiga. Võrreldes difusiooni algoritmiga on juhtsõlme hajusalt valimise algoritm lihtsam ning robustsem.

Doktoritöös analüüsitakse ka juhtsõlme hajusalt valimise algoritmi ning difusiooni algoritmi arvutuslikku keerukust ning energia tarbimist. Analüüside põhjal on juhtsõlme hajusalt valimise algoritmi modifitseeritud, et vähendada

selle energiatarvet ning muuta algoritm energiaefektiivsemaks. Välja töötatud energiaefektiivse juhtsõlme hajutatult valimise algoritmi energiatarve on vähenenud $32 - 53\%$ ning võrgu eluiga on pikenenud $14 - 46\%$ võrra võrreldes juhtsõlme hajusalt valimise algoritmiga ja difusiooni algoritmiga.

Part II

Included Publications

Paper A – Distributed Adaptive Network with SNR Weighed Communication

Distributed Adaptive Network with SNR Weighed Communication

Sander Ulp and Tõnu Trump
Institute of Radio and Telecommunication Engineering
Tallinn University of Technology
Tallinn, Estonia
Email: sander.ulp@ttu.ee, tonu.trump@lr.ttu.ee
Telephone: +372 56 986 324

***Abstract*—Recent studies of self-organizing adaptation in nature, have indicated that similar concepts can be implemented in modern network systems. There have been studies that show how to use self-organizing multi agent adaptive networks for signal estimation, detection and network optimization. A possible application for signal detection based adaptive networks can be implemented in cognitive radio where a number of secondary users need to detect the presence of the primary user activity. In this work we analyse the performance of an adaptive network of nodes with a static topology that estimates the signal to noise ratio (SNR) of the received signal in each node and weighs the communication according to the estimates. The nodes compute the autocorrelation function of the primary signal for further cognitive radio detection application.**

I. Introduction

Cognitive radio networks allow to use the spectrum more efficiently, compared to the traditional systems based on spectrum licensing [1], by allowing secondary users to transmit when the primary user at that frequency band is not transmitting. In order to improve the performance of sensing the primary signal adaptive network algorithms have been proposed to enhance the detection probability [2]. This is achieved by exchanging estimation information between secondary users that communicate with each other. Each secondary user computes its own estimate and improves it by comparing and adding estimation results from other users. The communication follows certain rules and a certain topology [3].

In many works the estimation problem is solved using a central node referred to as "fusion centre" that is connected to all of the nodes and receives all the estimates that the nodes compute. The common estimate computed by the fusion centre is then fed back to all the nodes in the network [4]. This method would require a node that has high processing power and connections with all of the nodes. However, in practical applications a method with low-complexity processing is desirable and the fusion centre would pose a single point of failure problem. In case of failure of the fusion centre the whole network is rendered useless. Therefore, decentralized solutions have been proposed such as consensus strategies, incremental strategies and diffusion strategies [5], [3].

Incremental strategies define a cyclic path through the nodes and data is optimized over this path. However, the incremental strategy suffers from NP-hard problem and is receptive to link and node failures [3]. There have been works on consensus strategies that relied on the use of two time-scales which are unsuitable to use in real-time applications where measurement data continuously streams [5]. Consensus strategies that rely on single time-scale are able to perform well under certain conditions, but tend to suffer from instability. It is also shown that diffusion strategies with constant step size converge better than consensus strategies [5]. Therefore, we will in this paper focus on diffusion strategies.

Diffusion strategies allow the nodes to cooperate and diffuse information in real-time. There have been proposed two schemes for diffusion - Adapt-then-combine (ATC) and Combine-then-Adapt (CTA). Diffusion methods are robust to link and node failures and are able to continue learning even when the cost function changes with time [3]. Although using a diffusion strategy in a distributed adaptive network can enhance the performance of the adaptive network compared to a network with no cooperation between nodes, it shown that on the single node level the better performing nodes do not benefit from the cooperation [6]. In order to improve the performance the estimates that are communicated between nodes should be weighed according to some secondary parameter [6].

In this paper we propose to use the estimate of the signal to noise ratio (SNR) of the primary signal at the secondary user as a secondary parameter to weigh the communication between nodes. The performance of estimating the autocorrelation function of the primary signal in an adaptive network using an ATC diffusion strategy with SNR weighed communication between nodes will be presented. Estimating the autocorrelation function of the primary signal is useful for detection applications in cognitive radio [7]. We are going to compare the performance of the proposed algorithm with a cooperating network using equal combination weights and with a non-cooperating network.

In this paper italic letters are used for scalars (e, E), lower case bold letters are denoting vectors ($\mathbf{x}$) and capital bold letters are denoting matrices ($\mathbf{G}$). (*) denotes the Hermitian transpose of the subject, for example ($\mathbf{G}^*$) denotes the Hermitian transpose of the matrix ($\mathbf{G}$). Operator $E[\cdot]$ stands for mathematical expectation of the subject.

The remainder of the paper is organized as follows. Section 2 states the problem and presents the theoretical basis of the

ISBN: 978-1-4799-6376-8/15/$31.00 ©2015 IEEE

cooperating network with equal combination weights. Section 3 of the paper describes the method of estimating the SNR of the primary signal and the proposed algorithm of weighing the communication between nodes. In Section 4 we present the results of our simulation study. The final section concludes the paper and summarizes the results of the simulations.

II. Problem Statement

Assume that there is a primary signal $\mathbf{s}(n)$ and a network of secondary users, who can communicate with each other. We will view the collective of the secondary users as a network of nodes. The communication of the nodes is possible within a certain range so that each node is connected to a subset of nodes. We assume that the topology of the network is strongly connected [6]. The communication between the nodes is lossless and noiseless. The nodes are identical in terms of processing power and physical attributes.

Complex amplitude of the signal received from the primary user at a location of k–th secondary user is given by

$$\mathbf{y}_k(n) = \beta_k \mathbf{s}(n) + \mathbf{v}_k(n), \tag{1}$$

where β_k is the random channel amplification of the k–th channel with Gaussian distribution. $\mathbf{v}_k(n)$ is the additive white Gaussian noise (AWGN) at the location of k–th network node. The channel amplification β_k and noise $\mathbf{v}_k(n)$ are independent of the signal $\mathbf{s}(n)$.

Each node in the network estimates the autocorrelation function of the primary signal, l–th lag of which is given by

$$\begin{aligned}\mathbf{r}_k &= E[\mathbf{y}_k(n)\mathbf{y}_k^*(n-l)] \\ &= E[(\beta_k \mathbf{s}(n) + \mathbf{v}_k(n)) \\ &\cdot (\beta_k \mathbf{s}^*(n-l) + \mathbf{v}_k^*(n-l)] \\ &= \beta_k^2 E[\mathbf{s}(n)\mathbf{s}^*(n-l)] + E[\mathbf{v}_k(n)\mathbf{v}_k^*(n-l)].\end{aligned} \tag{2}$$

Normalizing the autocorrelation function with the zero lag element we arrive at

$$\mathbf{r}_k = \begin{cases} 1, & l = 0 \\ \frac{\beta_k^2 E[\mathbf{s}(n)\mathbf{s}^*(n-l)] + E[\mathbf{v}_k(n)\mathbf{v}_k^*(n-l)]}{\beta_k^2 E[\mathbf{s}(n)\mathbf{s}^*(n)] + E[\mathbf{v}_k(n)\mathbf{v}_k^*(n)]}, & l > 0. \end{cases} \tag{3}$$

The estimate in each node k is computed at each iteration using exponential averaging

$$\mathbf{r}_k(n) = (1-\mu)\mathbf{r}_k(n-1) + \mu \mathbf{y}_k(n)\mathbf{y}_k^*(n-l), \tag{4}$$

where μ is a constant step size and $0 < \mu < 1$.

Each node in the network computes its own normalized autocorrelation function estimate at every iteration. The nodes will also receive the normalized autocorrelation function estimates from the adjacent nodes in the network that are connected with the node. The weights of the communication paths are calculated based on the number adjacent nodes that are connected (5) and the weights of the communications form a combination matrix $\mathbf{W}$. The equation (5) results in that the autocorrelation function estimates are weighed at each node equally which also results in a doubly stochastic combination matrix [6]. The combination weight $w_{f,k}$ between the nodes f and k is given by:

$$w_{f,k} = \frac{p_{f,k}}{\sum_{k=1}^{K} p_{f,k}}, \tag{5}$$

where

$\sum_{k=1}^{K} p_{f,k}$ is the number of connected adjacent nodes,

$p_{f,k} = \begin{cases} 1, \text{if a connection between the nodes } f \text{ and } k \text{ is present} \\ 0, \text{if a connection between the nodes } f \text{ and } k \text{ is not present,} \end{cases}$

K is the number of nodes in the network,

and f signifies the node that assigns the combination weights. For example, $w_{2,7}$ holds the combination weight between node 2 and node 7.

At each node the estimates are then combined according to (6). Thus, the average estimate $\bar{\mathbf{r}}_k(n)$ at iteration n is calculated as follows:

$$\bar{\mathbf{r}}_k(n) = \sum_{k=1}^{K} \mathbf{r}_k(n) \cdot w_{f,k}. \tag{6}$$

Substituting the average estimate (6) into the adaptive algorithm (4) we arrive at

$$\mathbf{r}_k(n) = (1-\mu)\bar{\mathbf{r}}_k(n-1) + \mu \mathbf{y}_k(n)\mathbf{y}_k^*(n-l). \tag{7}$$

Equation (7) expresses the adaptive algorithm at node k. At each iteration each node will thus compute their estimate based on the previous averaged estimate $\bar{\mathbf{r}}_k(n-1)$ and communicate and receive the computed estimate $\mathbf{r}_k(n)$ to and from their adjacent nodes to compute the new averaged estimate $\bar{\mathbf{r}}_k(n)$ (6). The algorithm is expressed as follows:

Algorithm 1: Adaptive algorithm with equal combination weights

for each time instant $n \geq 0$:

each node k = 1, 2, . . . , K performs the update:

$$\mathbf{r}_k(n) = (1-\mu)\bar{\mathbf{r}}_k(n-1) + \mu \mathbf{y}_k(n)\mathbf{y}_k^*(n-l)$$

$$w_{f,k} = \frac{p_{f,k}}{\sum_{k=1}^{K} p_{f,k}}$$

$$\bar{\mathbf{r}}_k(n) = \sum_{k=1}^{K} \mathbf{r}_k(n) \cdot w_{f,k}$$

end

III. Proposed algorithm

In order to improve the performance of the network a second local parameter is introduced to the network. Each node estimates the signal to noise ratio of the received signal and communicates the value in real-time to the adjacent nodes that are connected to it.

The SNR estimate is computed by calculating the sample covariance matrix of the received signal, computing the eigenvalues of the covariance matrix and determining the subspace of the signal and noise [8].

The k–th user sample covariance matrix $\hat{\mathbf{R}}_k$ can be computed as follows:

$$\hat{\mathbf{R}}_k = \frac{1}{j}\sum_{m=1}^{j} \mathbf{y}_k(m)\mathbf{y}_k^*(m), \tag{8}$$

where
j is the number of recent observation vectors
m is the length of $\mathbf{y}_k(m)$.

Assume that the eigenvalues of the $m \times m$ sample covariance matrix b_i are ranked in descending order $b_1 \geq b_2 \geq b_3 \geq ... \geq b_m$. The signal subspace is determined by using Minimum Description Length (MDL) criterion [8] which is defined as

$$MDL_k(N) = -\log\left[\frac{\prod_{i=N+1}^{m} b_i^{\frac{1}{(m-N)}}}{\frac{1}{m-N}\sum_{i=N+1}^{m} b_i}\right]^{(m-N)j} + \frac{1}{2}N(2m-N)\log j, \tag{9}$$

where
$N \in [0, 1, 2, 3, 4, ..., m-1]$.

The signal subspace dimension at a location of k–th secondary user is N that minimizes (9). Hence:

$$h_k = \underset{N}{argmin}\, MDL_k(N). \tag{10}$$

From the signal subspace h_k we are able to estimate the power of the noise and the signal at the location of k–th secondary user (11), (12).

$$\sigma_N^2 = \frac{1}{m-h_k}\sum_{i=h_k+1}^{m} b_i, \tag{11}$$

$$P_k = \sum_{i=1}^{h_k}\left(b_i - \sigma_N^2\right). \tag{12}$$

Hence the signal to noise ratio estimate at the k–th secondary user is computed as:

$$SNR_k = \left(\frac{P_k}{m\sigma_N^2}\right). \tag{13}$$

The nodes will take into account the SNR estimate and weight the communication based on the values. The weights are calculated as a linear combinations of the SNR estimates (13) using the rule given in (5). This way we ensure that the information from nodes with lower signal to noise ratio have lower weights and information from nodes with higher signal to noise ratio have larger weights. At the k–th secondary user the weights in the combination matrix add up to 1 row-and column-wise retaining the doubly stochastic properties of the combination matrix as it was for the cooperating network with equal combination weights. Thus, the combination weights are computed as:

$$w_{f,k} = \frac{p_{f,k}\cdot SNR_k}{\sum_{k=1}^{K} p_{f,k}\cdot SNR_k}. \tag{14}$$

The weighed estimate $\hat{\mathbf{r}}_k(n)$ at iteration n is given as:

$$\hat{\mathbf{r}}_k(n) = \sum_{k=1}^{K} \mathbf{r}_k(n)\cdot w_{f,k}. \tag{15}$$

Substituting the (15) into (4) we will arrive at

$$\mathbf{r}_k(n) = (1-\mu)\hat{\mathbf{r}}_k(n-1) + \mu\mathbf{y}_k(n)\mathbf{y}_k^*(n-l). \tag{16}$$

To conclude: At each iteration a node will estimate the SNR of the received signal and compute their estimate based on the previous weighed estimate $\hat{\mathbf{r}}_k(n-1)$. The node communicates and receives the computed estimate $\mathbf{r}_k(n)$ and the SNR estimation SNR_k to and from their adjacent nodes to compute the new weighed estimate $\hat{\mathbf{r}}_k(n)$ (16). The algorithm is expressed as follows:

Algorithm 2: Adaptive algorithm with SNR combination weights

for each time instant $n \geq 0$:
each node k = 1, 2, . . . , K performs the update:

$$\mathbf{r}_k(n) = (1-\mu)\bar{\mathbf{r}}_k(n-1) + \mu\mathbf{y}_k(n)\mathbf{y}_k^*(n-l)$$

$$w_{f,k} = \frac{p_{f,k}\cdot SNR_k}{\sum_{k=1}^{K} p_{f,k}\cdot SNR_k}$$

$$\bar{\mathbf{r}}_k(n) = \sum_{k=1}^{K} \mathbf{r}_k(n)\cdot w_{f,k}$$

end

IV. Simulation Results

Let us assume a network of 9 nodes with a static topology which is shown in Fig. 1.

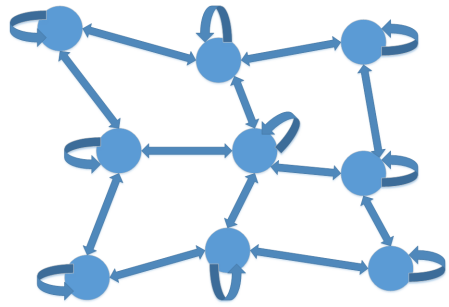


Fig. 1. Topology of the adaptive network

ISBN: 978-1-4799-6376-8/15/$31.00 ©2015 IEEE

TABLE I
TABLE OF CONNECTIONS

$p_{f,k}$		k								
		1	2	3	4	5	6	7	8	9
f	1	1	1	0	1	0	0	0	0	0
	2	1	1	1	0	1	0	0	0	0
	3	0	1	1	0	0	1	0	0	0
	4	1	0	0	1	1	0	1	0	0
	5	0	1	0	1	1	1	0	1	0
	6	0	0	1	0	1	1	0	0	1
	7	0	0	0	1	0	0	1	1	0
	8	0	0	0	0	1	0	1	1	1
	9	0	0	0	0	0	1	0	1	1

From the topology it is seen that all of the nodes share their estimate with themselves and communicate with their adjacent neighbours. The communication between the nodes, when it exists, is full duplex. The table of connections is expressed in Table I. The value "1" reflects a connection between nodes and the value "0" reflects the absence of a connection between nodes (5). The f and k values signify the connections between certain nodes. For example, $p_{2,7}$ shows if there is a connection between node 2 and node 7.

In order to compare the performance of the algorithm, the performance of the non-cooperating network is given in comparison. Assume the same condition and the same topology without any connection between adjacent nodes. The non-cooperating network is shown in Fig. 2. A small constant step size $\mu = 0.01$ is selected for the simulations. For simplicity, it is assumed that the noise power level at each node is $\delta_v^2 = 1$ and the primary signal power is $P_k = 1$. At each node the channel amplification β_k is generated and is assumed to be slow fading during the observation interval. Since the noise power level at each node is constant the SNR estimates reflect the channel attenuation at each node, hence for observation purposes the channel attenuations are selected decreasingly from Node 1 to Node 9. The results are averaged over 100 independent observations.

Fig. 3 expresses the mean squared error (MSE) convergence averaged over all of the autocorrelation values and across all

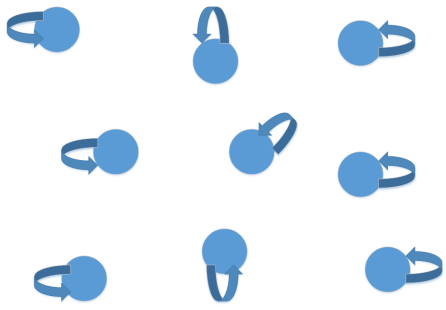


Fig. 2. Topology of the non-cooperating network

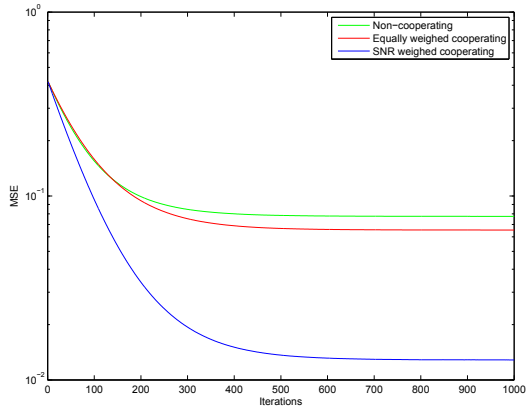


Fig. 3. Network convergence MSE

nodes, where $\beta_k \in \{2.25 + 2.25i; 2 + 2i; 1.75 + 1.75i; 1.5 + 1.5i; 1.25 + 1.25i; 1 + 1i; 0.75 + 0.75i; 0.5 + 0.5i; 0.25 + 0.25i\}$. The learning curves of the non-cooperating network, the cooperating network and the proposed algorithm convergences are presented in the figure.

It can be deducted that proposed the algorithm outperforms the non-cooperating and the cooperating network using equally weighed communication.

In Fig. 4 and in Fig. 5 we present the learning curves for node 6 and node 7 respectively. It is shown in Fig. 4 that the performance of the node 6 in the non-cooperating adaptive network outperforms the cooperating network with equal combination weights. This is due to the fact that the nodes in question have adjacent nodes whose signal to noise ratios are poorer than the observed node and averaged across the nodes the performance degrades. In Fig. 5 it is shown that the cooperating network with equal combination weights at node 7 is able to outperform the non-cooperating network.

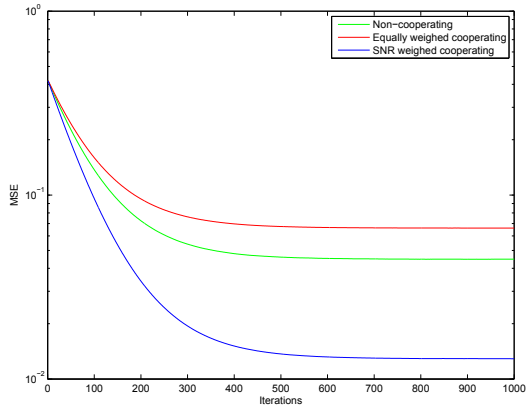


Fig. 4. Convergence at node 6

ISBN: 978-1-4799-6376-8/15/$31.00 ©2015 IEEE

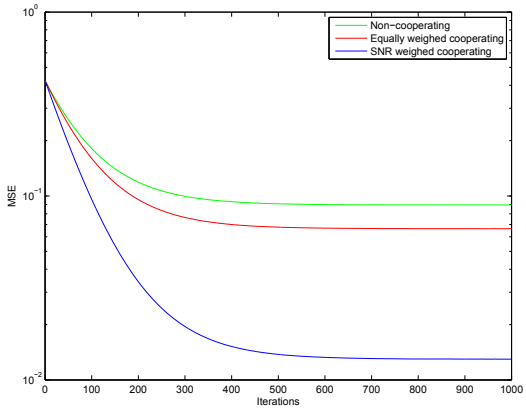


Fig. 5. Convergence at node 7

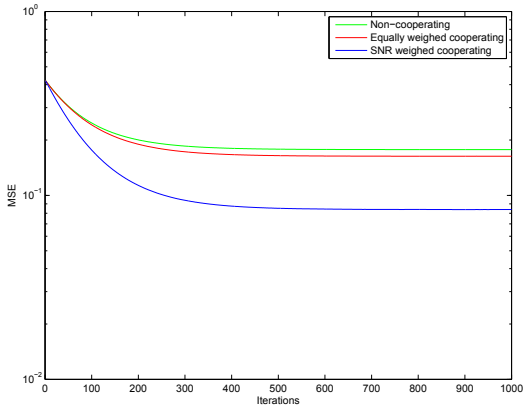


Fig. 6. Network convergence MSE with Rayleigh fading

Nodes 7-9 have the lowest signal to noise ratio and benefit from averaging the estimations with their adjacent nodes. Dependent on the channel attenuation β_k the cooperating network with equal combination nodes will result in better estimates for the nodes that have adjacent nodes with better than average signal to noise ratio and worse estimates for nodes that have adjacent nodes with worse signal to noise ratio than average. The performance of the SNR weighed combination algorithm does not suffer from this problem and is able to outperform the non-cooperating and the cooperating network with equal combination weights.

Fig. 6 expresses the mean squared error (MSE) convergence averaged over all of the autocorrelation values and across all nodes where the channel amplification β_k values are randomly generated Rayleigh fading values. Weighing the communication with SNR estimates has improved the cooperating network performance and thus the proposed algorithm outperforms the non-cooperating network and the cooperating network with equal combination weights.

V. Conclusion

In this paper we have investigated the performance of estimating the autocorrelation function of the primary signal in an adaptive cooperating network using combination weights that base on the linear combinations of the SNR values estimated by the network nodes. As an example we used a static topology with 9 nodes and the nodes computed their estimates using the exponential averaging. The results were compared to an identical network with non-cooperating nodes and to an identical cooperating network using equal combination weights. It is shown that the estimation performance of the nodes in cooperating network is dependent on signal to noise ratios of the node and its adjacent nodes. Nodes with better signal to noise ratio performed worse than the non-cooperating nodes and nodes with worse signal to noise ratio performed better than the non-cooperating nodes. The SNR weighed algorithm is shown to improve the performance of the cooperating network by giving a higher level of confidence to estimations from nodes with higher signal to noise ratio conditions. Overall the algorithm improves the performance of the network on average and achieves better performance at a node level compared to the cooperating network using equal combination weights.

Acknowledgment

This research was supported by the Estonian Doctoral School in Information and Communication Technology.

References

[1] K. Seshukumar, R. Saravanan, and M. Suraj, "Spectrum sensing review in cognitive radio," in *Emerging Trends in VLSI, Embedded System, Nano Electronics and Telecommunication System (ICEVENT), 2013 International Conference on*. IEEE, 2013, pp. 1–4.

[2] Z. Quan, S. Cui, A. H. Sayed, and H. V. Poor, "Optimal multiband joint detection for spectrum sensing in cognitive radio networks," *Signal Processing, IEEE Transactions on*, vol. 57, no. 3, pp. 1128–1140, 2009.

[3] J. Chen and A. H. Sayed, "Diffusion adaptation strategies for distributed optimization and learning over networks," *Signal Processing, IEEE Transactions on*, vol. 60, no. 8, pp. 4289–4305, 2012.

[4] X. Zhao and A. H. Sayed, "Performance limits for distributed estimation over lms adaptive networks," *Signal Processing, IEEE Transactions on*, vol. 60, no. 10, pp. 5107–5124, 2012.

[5] S.-Y. Tu and A. H. Sayed, "Diffusion strategies outperform consensus strategies for distributed estimation over adaptive networks," *Signal Processing, IEEE Transactions on*, vol. 60, no. 12, pp. 6217–6234, 2012.

[6] A. H. Sayed, "Adaptation, learning, and optimization over networks," *Foundations and Trends in Machine Learning*, vol. 7, no. 4–5, pp. 311–801, 2014.

[7] V. Turunen, M. Kosunen, A. Huttunen, S. Kallioinen, P. Ikonen, A. Parssinen, and J. Ryynanen, "Implementation of cyclostationary feature detector for cognitive radios," in *Cognitive Radio Oriented Wireless Networks and Communications, 2009. CROWNCOM'09. 4th International Conference on*. IEEE, 2009, pp. 1–4.

[8] M. Wax and T. Kailath, "Detection of signals by information theoretic criteria," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 33, no. 2, pp. 387–392, 1985.

Paper B – Leader Selection in Cooperative Network Based on MDL Subspace Algorithm for Cognitive Radio

Leader Selection in Cooperative Network Based on MDL Subspace Algorithm for Cognitive Radio

Sander Ulp and Tõnu Trump
Department of Radio and Telecommunication Engineering
Tallinn University of Technology
Tallinn, Estonia
Email: sander.ulp@ttu.ee, tonu.trump@lr.ttu.ee
Telephone: +372 56 986 324

***Abstract*—This paper presents a simple and robust algorithm for determining a leader node in a cooperative network based on MDL (Minimum Description Length) subspace algorithm. The algorithm aims to improve the performance of the cooperating network in a spectrum sensing problem for cognitive radio. The outline of the communication and selection process is described and the SNR (signal to noise ratio) estimation algorithm is given. Simulation results show that the algorithm outperforms the non-cooperating network and a cooperating diffusion network with uniform combination weights.**

I. Introduction

With emergence of high data rate communication systems the need for more efficient usage of the frequency spectrum is increasing. An innovative method of more efficient usage of already allocated spectrum is cognitive radio [1]. Cognitive radio is a concept that allows non-licensed users (secondary users) to utilize the radio spectrum when the licensed user (primary user) is not transmitting. In this setting the there arises the problem of detecting the primary user activity. In order to improve the performance of the detection effort it has been proposed that the secondary users should cooperate and share their estimates in an adaptive and learning manner [2].

Many cooperating strategies and algorithms have been proposed in the past [3], [4]. Strategies such as incremental and centralized that rely on certain topology or network architecture which make them less flexible in terms of cognitive radio situations [3], [5]. It has been shown that fully distributed networks such as diffusion networks are a more suited for the cognitive radio problem [3].

Each node in the cooperating network is able to communication with a set of nodes, but are not in close proximity, hence operating in different situation in terms of performance due to the noise profiles and channel gains. Using the cooperative network with uniform weights the overall performance of the network is improved, but on the single node level the best performing nodes sacrifice their performance and the cooperating process is not beneficial for them [3], [5]. Methods for determining and weighing the nodes accordingly to their conditions have been proposed [6]–[8]. Diffusion networks have been shown to perform well they require additional information for non-equal combination weights which is not usually available to network in real-time [3]. While there are works on methods of introducing secondary information to the network they require certain prerequisites and the performance of the network is heavily reliant on the secondary parameter [9]. Therefore, acquiring reliable or accurate combination weights poses a problem.

We propose an algorithm that works in a fully distributed manner and the performance of the network does not heavily rely on combination weights. Each of the nodes determines the best performing neighbour, learns its estimates and distributes them. In this manner each of the nodes in the network would in time achieve the performance of the best performing node. A secondary parameter is introduced to determine the best performing however the accuracy of the secondary parameter estimate is not heavily connected to the performance of the algorithm. Thus, we propose a simple and robust algorithm that uses a secondary parameter to determine the best performing node and cooperate based on this. It is shown that the algorithm is able to outperform the diffusion algorithm with uniform combination weights and the non-cooperating network.

In this paper italic letters are used for scalars (e, E), lower case bold letters are denoting vectors $(\mathbf{x})$ and capital bold letters are denoting matrices $(\mathbf{G})$. All vectors are assumed to be column vectors and for convenience are presented as the transpose of the row vector. (y^*) denotes the complex conjugate of (y) and $(\mathbf{G}^H)$ is the hermitian transpose of the matrix $(\mathbf{G})$. Operator $E[\cdot]$ stands for mathematical expectation of the subject.

The remainder of the paper is organized as follows. In section 2 the problem formulation is given. Section 3 of the paper describes the proposed algorithm of communication leader selection and the MDL subspace algorithm is introduced. Section 4 of the paper describes the performance of the proposed algorithm. In Section 5 we present the results of our simulations. The final section concludes the paper and summarizes the results of the paper.

This research was supported by the European Regional Development Fund, the Information Technology Foundation for Education and the HITSA project for the development and research of the laboratory for the telecommunication services curriculum at Institute of Radio and Telecommunication Engineering, Tallinn University of Technology.

II. Problem Statement

Assume that there is a primary user signal $s(n)$ and a network of K secondary users, who are sensing the primary user activity. We will view the collective of the secondary users as a network of nodes. The nodes communicate with each other and the communication of the nodes is possible within a certain range so that each node is connected to a subset of nodes. We assume that the topology of the network is strongly connected [3]. The communication between the nodes is modelled as lossless and noiseless. The nodes are identical in terms of processing power and physical attributes.

Complex amplitude of the signal received from the primary user at a location of k–th secondary user is given by

$$y_k(n) = \beta_k s(n) + v_k(n), \tag{1}$$

where β_k is the random channel amplification of the k–th channel with Gaussian distribution. $v_k(n)$ is the additive white Gaussian noise (AWGN) at the location of k–th network node. The channel amplification β_k and noise $v_k(n)$ are independent of the signal $s(n)$.

Each node in the network estimates the autocorrelation function of the primary user signal $y(n)$, the l–th lag of the autocorrelation function is given as

$$\begin{aligned} r_k(l) &= E[y_k(n)y_k^*(n-l)] \\ &= E[(\beta_k s(n) + v_k(n)) \\ &\cdot (\beta_k s^*(n-l) + v_k^*(n-l)] \\ &= \beta_k^2 E[s(n)s^*(n-l)] + E[v_k(n)v_k^*(n-l)]. \end{aligned} \tag{2}$$

Normalizing the autocorrelation function with the zero lag element we arrive at

$$r_k(l) = \begin{cases} 1, & l = 0 \\ \frac{\beta_k^2 E[s(n)s^*(n-l)] + E[v_k(n)v_k^*(n-l)]}{\beta_k^2 E[s(n)s^*(n)] + E[v_k(n)v_k^*(n)]}, & l > 0. \end{cases} \tag{3}$$

In practice, we of course use the estimate of the autocorrelation function. Thus we can write the estimate of the normalized autocorrelation function at the l–th lag as

$$r_k(l) = \frac{\sum_{n=0}^{N-1} y_k(n)y_k^*(n-l)}{\sum_{n=0}^{N-1} y_k(n)y_k^*(n)}. \tag{4}$$

The estimated normalized autocorrelation function vector can be written as

$$\mathbf{r}_k(i) = [r_k(0), r_k(1), r_k(2), r_k(3), ..., r_k(L-1)]^T. \tag{5}$$

The averaged estimate at each node k is computed at each iteration using exponential averaging:

$$\mathbf{w}_k(i) = (1-\mu)\mathbf{w}_k(i-1) + \mu \mathbf{r}_k(i), \tag{6}$$

where μ is a constant step size and $0 < \mu < 1$.

This outlines the single agent network where every node works independently. In the cooperative network the nodes will also receive the normalized autocorrelation function estimates from the adjacent nodes in the network that are connected with the node.

III. Proposed algorithm

Each of the node in the cooperating network has different situation in terms of performance due to the noise profiles and channel gains. For the cooperative network with equal weights the overall performance of the network is improved, but on the single node level the best performing nodes sacrifice their performance and the cooperating process is not beneficial for them [3]. If the nodes would cooperate in such manner that they would determine the best node to listen to the network would achieve the performance of the best performing node in the network. Thus our idea is to introduce a secondary parameter to select the best performing node. Each node estimates the signal to noise ratio of the received primary user signal $\mathbf{y}_k$.

The signal to noise ratio (SNR) of the received signal $\mathbf{y}_k$ is computed using the Minimum Description Length (MDL) subspace algorithm [10]–[12]. By calculating the sample covariance matrix of the received signal $\mathbf{y}_k$, computing the eigenvalues of the covariance matrix and determining the subspace of the signal and noise we can obtain the SNR estimate at the k–th secondary user.

The k–th user sample covariance matrix $\hat{\mathbf{R}}_k$ can be computed as follows:

$$\hat{\mathbf{R}}_k = \frac{1}{j}\sum_{i=1}^{j} \mathbf{y}_k(i)\mathbf{y}_k^H(i), \tag{7}$$

where j is the number of recent observation vectors and i is the time instance of when $\mathbf{y}_k(i)$ is received.

Ranking the eigenvalues of the $n \times n$ sample covariance matrix b_t in descending order $b_1 \geq b_2 \geq b_3 \geq ... \geq b_n$. The signal subspace is determined by using MDL criterion which is defined as

$$\begin{aligned} \mathrm{MDL}_k(N) = &-\log\left[\frac{\prod_{t=N+1}^{n} b_t^{\frac{1}{(n-N)}}}{\frac{1}{n-N}\sum_{t=N+1}^{m} b_t}\right]^{(m-N)j} \\ &+ \frac{1}{2}N(2n-N)\log j, \end{aligned} \tag{8}$$

where $N \in [0, 1, 2, 3, 4, ..., n-1]$.

The signal subspace dimension at a location of k–th secondary user is N that minimizes (8). Hence,

$$h_k = \underset{N}{\operatorname{argmin}}\ \mathrm{MDL}_k(N). \tag{9}$$

From the signal subspace h_k we are able to estimate the power of the noise at k–th secondary user:

$$\sigma_N^2 = \frac{1}{m - h_k} \sum_{t=h_k+1}^{m} b_t. \tag{10}$$

The estimate of the power of the signal at the location of k–th secondary user is given as

$$P_k = \sum_{t=1}^{h_k} \left(b_t - \sigma_N^2 \right). \tag{11}$$

Hence, the signal to noise ratio estimate at the k–th secondary user is computed as

$$\text{SNR}_k = \left(\frac{P_k}{\sigma_k^2} \right), \tag{12}$$

where the $P_k = \beta_k^2 P$ is the power of the signal at node k, P is the primary signal power, β_k is the channel gain for node k and σ_k^2 is the noise power.

The estimated signal to noise value is exponentially averaged over time and is defined as the weight at node k:

$$\alpha_k(i) = (1 - \mu)\alpha_k(i-1) + \mu \text{SNR}_k. \tag{13}$$

The nodes will proceed to exchange autocorrelation estimates between nodes together with the corresponding weight $\alpha_k(i)$ that is assigned to the estimate at iteration i. This ensures that each of the nodes distributes the best information across the network that is available to them. The algorithm at a node level is explained as following:

At each iteration the each of the nodes determines whether their weight is the largest of the neighbouring nodes:

$$\alpha_k(i-1) \geq \max_{1<k<K} (\alpha_k(i-1) \cdot p_{f,k}), \tag{14}$$

where $\alpha_k(i-1)$ is the calculated weight given in (13) and $p_{f,k} \in [0, 1]$ signifies whether there is a connection between the nodes or not.

If this statement holds true the node will use its own adaptation result as the best available information. The best available information at node k is given as

$$\mathbf{e}_k(i) = \mathbf{w}_k(i). \tag{15}$$

The node will also calculate a new weight value based on the estimated SNR value:

$$\alpha_k(i) = (1 - \mu)\alpha_k(i-1) + \mu \text{SNR}_k. \tag{16}$$

If the statement in (14) is false and the weight at node k is smaller than any of the weights from adjacent nodes the node will determine the best neighbouring source of information from the adjacent nodes. This corresponds to the neighbouring node with the largest weight in the comparison which is given as

$$c_k = \underset{k}{\text{argmax}}(\alpha_k(i-1) \cdot p_{f,k}) \tag{17}$$

and use the neighbours estimate instead:

$$\mathbf{e}_k(i) = \mathbf{e}_{c_k}(i). \tag{18}$$

At the end of iteration each of the nodes will use their SNR estimate to calculate the weight for the next iteration. This ensures that if the signal to noise ratio of the nodes change the node with the best information will have their information distributed during the next iteration.

$$\alpha_k(i) = (1 - \mu)\alpha_{c_k}(i-1) + \mu \text{SNR}_k. \tag{19}$$

Algorithm 1: Leader selection in cooperative network

for each time instant $i > 0$:

each node k = 1, 2, . . . , K performs the update:

$$\mathbf{w}_k(i) = (1 - \mu)\mathbf{w}_k(i-1) + \mu \mathbf{r}_k(i)$$

if

$$\alpha_k(i-1) \geq \max_{1<k<K} (\alpha_k(i-1) \cdot p_{f,k})$$

$$\mathbf{e}_k(i) = \mathbf{w}_k(i)$$

$$\alpha_k(i) = (1 - \mu)\alpha_k(i-1) + \mu \text{SNR}_k$$

else

$$c_k = \underset{k}{\text{argmax}}(\alpha_k(i-1) \cdot p_{f,k})$$

$$\mathbf{e}_k(i) = \mathbf{e}_{c_k}(i)$$

$$\alpha_k(i) = (1 - \mu)\alpha_{c_k}(i-1) + \mu \text{SNR}_k$$

end

end

IV. Performance

The MSE (mean squared error) performance of a node in this setting is given as

$$\text{MSE}_k = \sum_{i=1}^{I} [\mathbf{w}_k(i) - \mathbf{q}(i)]^2, \tag{20}$$

where $\mathbf{q}(i)$ is the normalized autocorrelation function vector of the primary user signal $s(n)$:

$$\mathbf{q}(i) = [q(0), q(1), q(2), q(3), ..., q(L-1)]^T. \tag{21}$$

The normalized autocorrelation function at the l–th lag of the primary user signal $s(n)$ can be written as

$$q(l) = \frac{\sum_{n=0}^{N-1} s(n)s^*(n-l)}{\sum_{n=0}^{N-1} s(n)s^*(n)}. \tag{22}$$

The steady state MSE performance of the algorithm is easily deductible. Each of the nodes achieves the MSE performance of the best performing node and if we would average the performance across nodes the global MSE performance would be that of the best performing node:

$$\boxed{\text{MSE} = \min_k \text{MSE}_k}. \tag{23}$$

The gain of the algorithm in comparison to the non-cooperating network largely depends on the situation of individual nodes. If the nodes are in similar conditions in aspect to noise and channel gains the gain is marginal. However, if one or some of the nodes are in more favourable situation than others, the gain per node is the difference between the nodes in worse conditions and the best performing node.

In practice of course we are not aware of the MSE performance of different nodes and a secondary parameter is required to select the leader node. The accuracy of the secondary parameter estimate will only affect the algorithm if the estimate accuracy is poorer than the difference between different nodes. In this work we use the MDL subspace algorithm to estimate the signal to noise ratio of the primary signal. The performance of the MDL subspace algorithm will be left for future work, but the simulations have shown that the accuracy of the estimate is enough to determine the leader node in this setting.

V. Simulation Results

Let us assume a network of nodes that share their estimate with themselves and communicate with their adjacent neighbours. The communication between the nodes, when it exists, is full duplex. The topology of the network is static and the number of nodes in the network is 9 and the topology is shown on Fig. 1 [13]. A small constant step size $\mu = 0.01$ is selected for the simulations. The noise power level δ_k^2 at each node is modelled as AWGN and the primary user signal power is $P = 1$. The channel amplification β_k for each node is generated and the channel is modelled as a slow Rayleigh fading channel. The results are averaged over 100 independent trials.

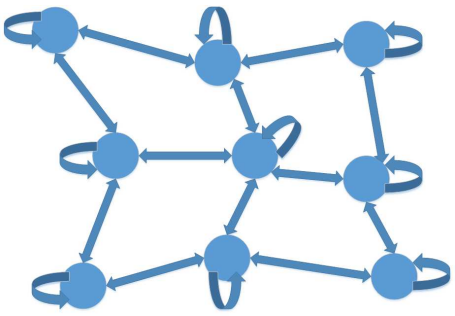


Fig. 1. Topology of the network

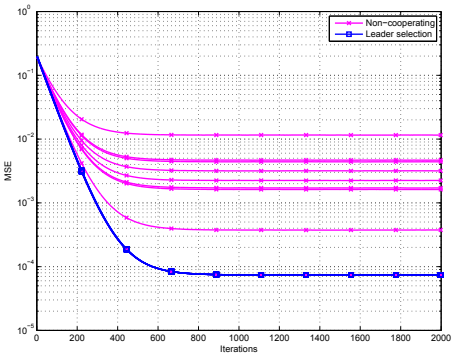


Fig. 2. Non-cooperating node and leader selection node MSE convergences

Fig. 2 shows the MSE convergences for different nodes in the non-cooperating scenario and for the proposed algorithm. It is seen that the performance of individual nodes for the proposed algorithm is that of the best performing node in the steady state and thus, the steady state performance of individual nodes in the proposed algorithm is identical. The gains in this particular scenario are straightforward, each of the nodes has gained in performance as the difference between the best performing node and their performance.

Fig. 3 illustrates that the nodes in the leader selection algorithm converge at different speed which is due to the distribution of the best performing nodes. The magenta lines show the MSE performance of individual nodes in the non-cooperating network. The blue lines show the MSE performance of individuals nodes in a network that switches from the non-cooperating scenario to the leader selection scenario at

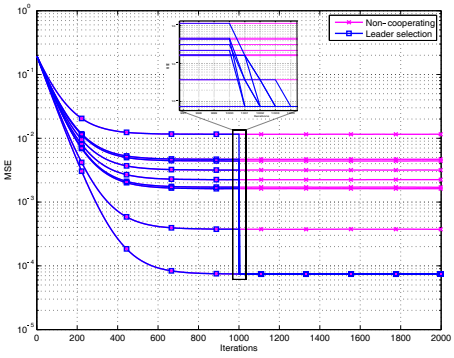


Fig. 3. Switch from non-cooperating to leader selection network

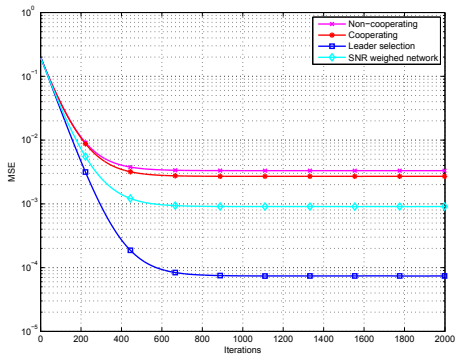


Fig. 4. Network convergence MSE with Rayleigh fading

iteration 1000. From the zoomed in section in Fig. 3 it is seen that nodes in different neighbourhoods will converge into local leaders and over many iterations all of the nodes will follow the global leader node. The speed of which the nodes select the global leader node depends on the topology and amount of nodes in the network. In this particular topology where the amount of nodes is 9 the convergence to a global leader is within 4 iterations which is the longest communication path in the network.

Fig. 4 shows the mean squared error (MSE) convergence averaged over all of the autocorrelation values and across all nodes. The algorithm simulation results are compared to that of the non-cooperating network, a network with ATC cooperation with equal weights, a network with ATC cooperation with SNR based weights [13], [14]. It can be seen that the leader selection algorithm outperforms the algorithms in comparison on the network level.

VI. Conclusion

In this paper we presented a simple and robust method for determining a leader node in a cooperating network. We introduced a secondary parameter based on MDL subspace algorithm to estimate the signal to noise ratio of the nodes in the network. The qualitative estimate was used to determine the best performing node in the network. It was shown that the network followed the leader node in a distributed matter and the network achieved the performance of the best performing node across all nodes. The results were compared to a network with non-cooperating nodes, a cooperating network with ATC diffusion using equal combination weights and a cooperating network with ATC diffusion using SNR weighed combination weights. It is shown that the proposed algorithm outperforms the networks in comparison on the network level and on the single node level.

References

[1] J. Mitola III and G. Q. Maguire Jr, "Cognitive radio: making software radios more personal," *Personal Communications, IEEE*, vol. 6, no. 4, pp. 13–18, 1999.

[2] T. Yücek and H. Arslan, "A survey of spectrum sensing algorithms for cognitive radio applications," *Communications Surveys & Tutorials, IEEE*, vol. 11, no. 1, pp. 116–130, 2009.

[3] A. H. Sayed, "Adaptation, learning, and optimization over networks," *Foundations and Trends in Machine Learning*, vol. 7, no. 4–5, pp. 311–801, 2014.

[4] I. F. Akyildiz, B. F. Lo, and R. Balakrishnan, "Cooperative spectrum sensing in cognitive radio networks: A survey," *Physical communication*, vol. 4, no. 1, pp. 40–62, 2011.

[5] A. H. Sayed, S.-Y. Tu, J. Chen, X. Zhao, and Z. J. Towfic, "Diffusion strategies for adaptation and learning over networks: an examination of distributed strategies and network behavior," *Signal Processing Magazine, IEEE*, vol. 30, no. 3, pp. 155–171, 2013.

[6] V. Blondel, J. M. Hendrickx, A. Olshevsky, J. Tsitsiklis *et al.*, "Convergence in multiagent coordination, consensus, and flocking," in *IEEE Conference on Decision and Control*, vol. 44, no. 3. IEEE; 1998, 2005, p. 2996.

[7] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Systems & Control Letters*, vol. 53, no. 1, pp. 65–78, 2004.

[8] C. G. Lopes and A. H. Sayed, "Diffusion least-mean squares over adaptive networks: Formulation and performance analysis," *IEEE Transactions on Signal Processing*, vol. 56, no. 7, pp. 3122–3136, 2008.

[9] S.-Y. Tu and A. H. Sayed, "Optimal combination rules for adaptation and learning over networks," in *Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), 2011 4th IEEE International Workshop on*. IEEE, 2011, pp. 317–320.

[10] M. Wax and T. Kailath, "Detection of signals by information theoretic criteria," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 33, no. 2, pp. 387–392, 1985.

[11] H. L. Van Trees, *Detection, estimation, and modulation theory, optimum array processing*. John Wiley & Sons, 2004.

[12] D. Sui and L. Ge, "On the blind snr estimation for if signals," in *First International Conference on Innovative Computing, Information and Control-Volume I (ICICIC'06)*, vol. 2. IEEE, 2006, pp. 374–378.

[13] S. Ulp and T. Trump, "Distributed adaptive network with SNR weighed communication," in *Digital Information, Networking, and Wireless Communications (DINWC), 2015 Third International Conference on*. IEEE, 2015, pp. 83–87.

[14] J. Chen and A. H. Sayed, "Diffusion adaptation strategies for distributed optimization and learning over networks," *Signal Processing, IEEE Transactions on*, vol. 60, no. 8, pp. 4289–4305, 2012.

Paper C – LMS-Based Leader Selection for Distributed Estimation

LMS-Based Leader Selection for Distributed Estimation

Sander Ulp, Yannick Le Moullec, and Muhammad Mahtab Alam
Thomas Johann Seebeck Department of Electronics
Tallinn University of Technology
Tallinn, Estonia
Email: sander.ulp@ttu.ee, yannick.lemoullec@ttu.ee, muhammad.alam@ttu.ee

***Abstract*—In this paper we investigate the distributed leader selection algorithm in a LMS-based (least mean squares) adaptive network. We introduce the modified algorithm and express the MSD (mean squared deviation) performance of the algorithm. The outline of the communication and setting of the problem is given and the LMS-based distributed leader selection algorithm is described. We compare the analytical performance of the algorithm to the diffusion algorithms and illustrate that the leader selection has robust performance compared to the diffusion algorithms. Simulation results show that the diffusion algorithms outperform the leader selection algorithm in most cases. However and interestingly, the results also show that leader selection attains the same or even better network performance than the diffusion algorithms in a scenario where one of the nodes is in better conditions (e.g. lower noise power) than the other nodes in the network.**

I. Introduction

Estimation over distributed networks is a method to enhance the performance of nodes that cooperate to solve a common task or problem. Different methods and algorithms have been proposed for estimation over distributed networks. These approaches include strategies such as centralized, diffusion, consensus and incremental [1]. Many of these methods, however, require some kind of architecture or *a priori* knowledge of the network.

It has been shown that diffusion algorithms achieve good steady state MSD performance in distributed networks and they do not require any predefined architecture [2], therefore we focus our interest on diffusion algorithms. It has been also shown that the ATC (adapt and then combine) diffusion network outperforms the CTA (combine and then adapt) diffusion network and thus, we consider the ATC diffusion network in this work [2]. The performance of these diffusion algorithms rely on the calculation of the communication weights which are based on a secondary parameter, for example noise powers at the nodes [1].

There are several methods for calculating the weights for the communication and the in depth analysis of the performance of different weights have been derived [3], [4]. By assuming that the noise powers are not uniform across the network the relative variance and relative degree variance weight calculation methods have been shown to achieve good performance [4] and will be considered in this paper. The secondary parameters are in most of the cases not known beforehand and have to be estimated [1], which directly impacts the weight calculation and therefore influences the performance of the diffusion network. Distributed leader selection offers an interesting alternative to the diffusion algorithms as it is more robust to the secondary parameter estimation [5].

There are many works on leader selection, however, they rely on different constraints or *a priori* knowledge [6]–[11]. In [8]–[11] works the leaders are known beforehand and have extra information or better sensors compared to the followers in the network. Some works also require the nodes to know the topology beforehand or estimate the paths to the leader [6], [12]. Therefore, for the comparison with the diffusion algorithms to be fair we are interested in a fully distributed leader selection solution that is able to adapt and learn in real-time.

The purpose of this work is to investigate the MSD performance of the modified version of our previously proposed distributed leader selection algorithm [5]. The algorithm is modified and implemented using LMS to have fair comparison with the diffusion algorithms proposed in [4]. We expected that the diffusion algorithms are able to outperform the distributed leader selection theoretically. However, the simulation results show that there are situations where the distributed leader selection is able to achieve similar or even better network performance. We analyse the scenarios in which the distributed leader selection performance is able to perform equally or better compete with diffusion strategies.

In this paper, italic letters are used for scalars (e, E), lower case bold letters are denoting vectors $(\mathbf{x})$. All vectors are column vectors except for regression vectors, which are denoted as $\mathbf{u}_{k,i}$. The operator $E[\cdot]$ stands for mathematical expectation of the subject and (y^*) denotes the complex conjugate of (y).

The remainder of the paper is organized as follows. In Section 2 we introduce the problem setting. Section 3 gives an overview of the LMS-based leader selection algorithm.

978-1-5386-4662-5/17/$31.00 ©2017 IEEE

Section 4 describes the performance of the algorithm, the comparison to the diffusion algorithms and presents the simulation results. The final section concludes the paper and summarizes the results of the paper.

II. Problem Formulation

Assume that there are K nodes. Each of the nodes have access to $d_k(i)$ and $\mathbf{u}_{k,i}$ at each time instant i.

$$d_k(i) = \mathbf{u}_{k,i}\mathbf{w}^0 + v_k(i) \tag{1}$$

where $d_k(i)$ is the measurement at node k, $\mathbf{u}_{k,i}$ is the row regression vector, $\mathbf{w}^0$ is the unknown column vector and $v_k(i)$ is zero-mean white random noise with power $\sigma^2_{v,k}$. The nodes estimate the $\mathbf{w}^0$ to minimize the cost function:

$$J_k(w) = E \mid d_k(i) - \mathbf{u}_{k,i}\mathbf{w} \mid^2 \tag{2}$$

Each of the nodes employs the LMS algorithm for adaptation:

$$\mathbf{w}_{k,i} = \mathbf{w}_{k,i-1} + \mu_k \mathbf{u}^*_{k,i}[d_k(i) - \mathbf{u}_{k,i}\mathbf{w}_{k,i-1}] \tag{3}$$

where μ_k is a positive step-size $0 < \mu_k < 1$.

This outlines the non-cooperating network where every node works independently. To improve the performance of the nodes at the network level, the nodes can cooperate by exchanging information between themselves.

The network performance can be further improved by assigning weights based on the situation of nodes which is related to some secondary parameter. The noise power $\sigma^2_{v,k}$ can be used as a secondary parameter to calculate the weights. We assume that noise powers are known *a priori* to the nodes for simplicity and for the sake of a fair comparison to the diffusion algorithms in the performance evaluation section. In practice, the noise powers are not usually known and have to be estimated. There are works on estimating noise powers at different nodes [4], [13]. We have also covered a possible solution in our last work which is based on the MDL (minimum description lenght) subspace algorithm [5].

III. LMS-based distributed leader selection

The goal of the distributed leader selection algorithm is to determine the leader node in a network of nodes in a distributed manner. Whether the noise powers are known or estimated, the nodes are able to use the secondary parameter information to determine which of the nodes is the best performing node and follow its lead. We have previously outlined the algorithm for the distributed leader selection [5]. In this paper, we look at the distributed leader selection algorithm using a LMS-based adaptive network, see Algorithm 1.

The nodes exchange $\mathbf{w}_{k,i}$ estimates between themselves, together with the corresponding weight $\alpha_k(i)$ that is assigned to the estimate at iteration i. The weights are calculated based on the noise powers; the algorithm is given as follows:

Each of the nodes k performs the LMS adaptation of $\mathbf{w}_{k,i}$ at each iteration i as can be seen on Line 3. The nodes exchange their results of the adaptation and the corresponding weight with nodes that they are connected in the neighbourhood. The nodes proceed to determine whether their weight is the largest of the neighbouring nodes (Line 4). If this statement holds true, the node uses its own LMS adaptation result as the best available information (Line 5). The node also calculates adaptively a new weight value for the next iteration based on the noise power (Line 6).

Algorithm 1 LMS-Based Distributed Leader selection

1 **for** each time instant $i > 0$:
2 for each node k = 1, 2, . . . , K performs the update:
3 $\mathbf{w}_{k,i} = \mathbf{w}_{k,i-1} + \mu_k \mathbf{u}^*_{k,i}[d_k(i) - \mathbf{u}_{k,i}\mathbf{w}_{k,i-1}]$
4 **if** $\alpha_k(i-1) \geq \max_{1<k<K}(\alpha_k(i-1) \cdot p_{f,k})$
5 $\mathbf{e}_{k,i} = \mathbf{w}_{k,i}$
6 $\alpha_k(i) = (1-\mu)\alpha_k(i-1) + \mu\sigma^{-2}_{v,k}$
7 **else**
8 $c_k = \underset{k}{\operatorname{argmax}}(\alpha_k(i-1) \cdot p_{f,k})$
9 $\mathbf{e}_{k,i} = \mathbf{e}_{c_k,i}$
10 $\alpha_k(i) = (1-\mu)\alpha_{c_k}(i-1) + \mu\sigma^{-2}_{v,k}$
11 **end**
12 **end**

If the statement on Line 4 is false and the weight at node k is smaller than any of the weights from connected nodes then the node determines the best neighbouring source of information (Line 8). This corresponds to the neighbouring node c_k with the largest weight $\alpha_k(i-1)$. The node uses the LMS adaptation result from the best neighbour as can be seen on Line 9. The node will calculate the next iteration weight based on the their own noise power and the weight that corresponds to the best neighbour, as can be seen on Line 10. This ensures that if the noise powers of the nodes change the node with the best information has its information distributed during the next iterations.

IV. Performance evaluation

The LMS-based distributed leader selection algorithm performance can be obtained from the single node performance from the non-cooperative network. All of the nodes in the cooperating network have different MSD performance due to their respective noise powers $\sigma^2_{v,k}$. The MSD performance of the non-cooperating node (denoted as ncoop) employing LMS is given as [1]:

$$\text{MSD}_{ncoop,k} = \frac{\mu M}{2}\sigma_{v,k}^2 \tag{4}$$

where M is the length of the adaptive filter, μ is the step size of the LMS algorithm, $\sigma_{v,k}^2$ is the noise power at node k.

The performance of the non-cooperating LMS network is given as [1]:

$$\text{MSD}_{ncoop} = \frac{\mu M}{2}\left(\frac{1}{N}\sum_{k=1}^{N}\sigma_{v,k}^2\right) \tag{5}$$

If we employ the leader selection algorithm, the network achieves the best performing nodes MSD performance. In this setting this is the node with the lowest noise power. We can express the network MSD performance of the leader selection algorithm (denoted as ls) as:

$$\text{MSD}_{ls} = \min_k \text{MSD}_{ncoop,k} \tag{6}$$

It is important to note that the network MSD performance of the leader selection algorithm is equal to the individual node performance. Thus, we can express the network and node performance of the leader selection algorithm as:

$$\text{MSD}_{ls} = \text{MSD}_{ls,k} = \frac{\mu M}{2}\min_k \sigma_{v,k}^2 \tag{7}$$

A. Comparison of algorithms

The analytical MSD performance of the diffusion network (denoted as coop) as well as the node performance with optimal weights is given as [1]:

$$\text{MSD}_{coop} = \text{MSD}_{coop,k} = \frac{\mu M}{2}\left(\sum_{k=1}^{N}\frac{1}{\sigma_{v,k}^2}\right)^{-1} \tag{8}$$

If we compare the MSD network performance of the diffusion network with optimal weights (7) and the performance of the leader selection algorithm (8) we can see that the leader selection does not outperform the diffusion network. To demonstrate this we write:

$$\frac{\mu M}{2}\min_k \sigma_{v,k}^2 \geq \frac{\mu M}{2}\left(\sum_{k=1}^{N}\frac{1}{\sigma_{v,k}^2}\right)^{-1} \tag{9}$$

which we can write as:

$$\min_k \sigma_{v,k}^2 \geq \left(\sum_{k=1}^{N}\frac{1}{\sigma_{v,k}^2}\right)^{-1} \tag{10}$$

To further elaborate, if we rewrite (10) and define the node with the lowest noise power as p:

$$p = \underset{k}{\text{argmin}}\,\sigma_{v,k}^2 \tag{11}$$

$$\left(\frac{1}{\sigma_{v,p}^2}\right)^{-1} \geq \left(\frac{1}{\sigma_{v,p}^2} + \sum_{k=1,k\neq p}^{N}\frac{1}{\sigma_{v,k}^2}\right)^{-1} \tag{12}$$

We see that the both equation sides include the term that includes the lowest noise power. From this we can see that the performance difference between the diffusion network and the leader selection algorithm is related to the sum term. We can conclude that the diffusion network outperforms the leader selection algorithm and we can write:

$$\text{MSD}_{coop} \leq \text{MSD}_{ls} \tag{13}$$

Let us consider the case where one of the nodes is in a more favourable situation and has significantly lower noise power compare to the other nodes in the network:

$$\sum_{k=1,k\neq p}^{N}\frac{1}{\sigma_{v,k}^2} \ll \frac{1}{\sigma_{v,p}^2} \tag{14}$$

In this scenario we see that the performance of the cooperating network largely depends on the node with lowest noise power:

$$\text{MSD}_{coop} \approx \frac{\mu M}{2}\left(\frac{1}{\sigma_{v,p}^2}\right)^{-1} \tag{15}$$

The resulting equation is the equation for the performance of the distributed leader selection algorithm (7) and we can write that under these conditions:

$$\text{MSD}_{coop} \approx \text{MSD}_{ls} \tag{16}$$

To summarize, the diffusion network outperforms the leader selection algorithm in most cases and in the presence of a node in more favorable conditions the performance difference between the two algorithms is marginal. It is important to note that the above results (13,16) hold for optimal combinational weights for the diffusion network and are valid if all the nodes obtain the same performance. However, simulations results show that it is not the case and there is variation among the performance of the nodes in the diffusion network as can be seen in [13], [14]. Furthermore, it is also important to note that the combination weight optimality heavily relies on the estimation accuracy of the secondary parameter. Inaccuracies of the estimation leads to the degradation of the diffusion network performance in comparison to the leader selection as the inaccuracy of the secondary parameter does not affect the performance to such large extent. Taking this into account the leader selection becomes an interesting alternative and in the next subsection we explore these scenarios in simulations.

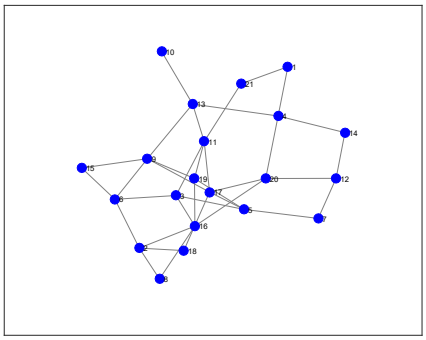
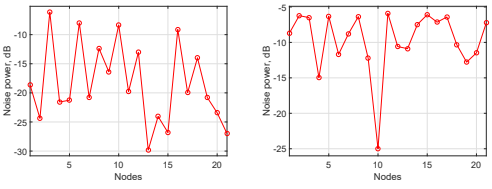
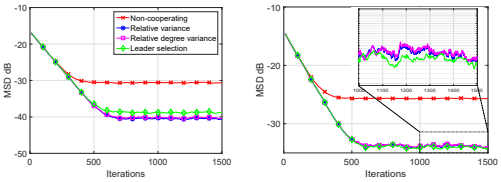


Fig. 1: Topology of the network.



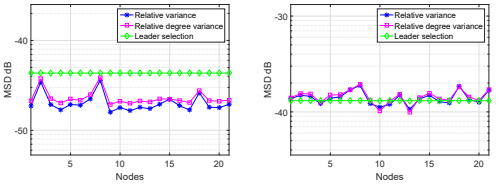
(a) (b)

Fig. 2: Noise powers at nodes.



(a) (b)

Fig. 3: Network MSD performances in both scenarios for the non-cooperating, the relative variance rule, the relative degree variance and the leader selection.



(a) (b)

Fig. 4: Steady state performances at individual nodes in both scenarios for the relative variance rule, the relative degree variance and the leader selection.

B. Simulation results

Let us assume a strongly connected network of nodes with randomly generated topology and the amount of nodes in the generated network is $K = 21$ (Fig. 1).

The nodes share their estimates based on the connections and it is assumed that the nodes have a self-loop which allows them to communicate with themselves. The communication between the nodes is full duplex and is modelled as lossless and noiseless. A constant step size $\mu = 0.01$ is selected for all the nodes in the simulations. The noise power levels are randomly generated between $\sigma^2_{v,k} = [-5 \quad -30]$ dB and are modelled as AWGN (additive white Gaussian noise) for each node. The regression vector power $P_u = 1$ is constant at each node. The results are averaged over 100 independent trials.

We examine two scenarios in our simulations. In the first scenario, the distribution of the noise powers can be seen on Fig. 2a. In the second scenario there is one node that has noticeably lower noise power than the other nodes in the network, as can be seen in Fig. 2b.

For the diffusion network we use the ATC algorithm and for the weight calculation the relative-variance rule and the relative-degree-variance rule [4]. We compare the network level MSD performance of the non-cooperating, the diffusion algorithms and the leader selection algorithm (Fig. 3). In the first scenario, the diffusion network algorithms' performances are close to each other and have better performance than the other algorithms, including the leader selection algorithm (Fig. 3a). However, for the other scenario the leader selection algorithm is able to achieve better network performance than the diffusion algorithms as can be seen in Fig. 3b.

The results can be explained by observing the MSD steady state performance of the individual nodes (Fig. 4). In the first scenario the diffusion network nodes' performances are closer to each other, but for the second scenario there are larger variations among different nodes' performances, as can be seen in Fig. 4a and Fig. 4b. In the first scenario the steady state performance of the nodes in the leader selection algorithm is poorer than the performances of the nodes of the diffusion algorithms. In the second scenario we see that some of the nodes of the diffusion algorithms are unable to achieve better performance than the leader nodes performance in the leader selection algorithm. This results in a better MSD network performance for the leader selection algorithm.

V. Conclusion

In this paper, we investigated the distributed leader selection algorithm in a LMS-based adaptive network. The algorithm analytical MSD performance was given and the performance was compared to the diffusion algorithm performance with optimal weights. The analytical performance shows that the diffusion networks are able to achieve better performance than the leader selection algorithm. It is shown that the leader selection achieves similar performance under the condition where one of the nodes is in significantly better conditions. For the simulations, the relative variance and relative degree variance diffusion algorithms were compared to the leader selection algorithm. The diffusion algorithms were able to achieve better performance than the leader selection in most cases. However, in the presence of one node with significantly better conditions, the leader selection algorithm is able to obtain similar or even better network performance than the algorithms in comparison.

Acknowledgment

This research was supported by the European Regional Development Fund and the Information Technology Foundation for Education. This project has also received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement No. 668995 and the Institutional Research Project IUT19-11 financed by the Estonian Research Council.

References

[1] A. H. Sayed, "Adaptation, learning, and optimization over networks," *Foundations and Trends in Machine Learning*, vol. 7, no. 4–5, pp. 311–801, 2014.

[2] S.-Y. Tu and A. H. Sayed, "Diffusion strategies outperform consensus strategies for distributed estimation over adaptive networks," *IEEE Transactions on Signal Processing*, vol. 60, no. 12, pp. 6217–6234, 2012.

[3] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *Proceedings of the 4th international symposium on Information processing in sensor networks*. IEEE Press, 2005, p. 9.

[4] S.-Y. Tu and A. H. Sayed, "Optimal combination rules for adaptation and learning over networks," in *2011 4th IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*. IEEE, 2011, pp. 317–320.

[5] S. Ulp and T. Trump, "Leader selection in cooperative network based on mdl subspace algorithm for cognitive radio," in *2016 50th Asilomar Conference on Signals, Systems and Computers*. IEEE, 2016, pp. 704–708.

[6] I. Shames, A. M. Teixeira, H. Sandberg, and K. H. Johansson, "Distributed leader selection without direct inter-agent communication," *IFAC Proceedings Volumes*, vol. 43, no. 19, pp. 221–226, 2010.

[7] Y. Hong, G. Chen, and L. Bushnell, "Distributed observers design for leader-following control of multi-agent networks," *Automatica*, vol. 44, no. 3, pp. 846–850, 2008.

[8] F. Lin, M. Fardad, and M. R. Jovanovic, "Algorithms for leader selection in stochastically forced consensus networks," *IEEE Transactions on Automatic Control*, vol. 59, no. 7, pp. 1789–1802, 2014.

[9] K. Fitch and N. E. Leonard, "Information centrality and optimal leader selection in noisy networks," in *2013 IEEE 52nd Annual Conference on Decision and Control (CDC)*. IEEE, 2013, pp. 7510–7515.

[10] S. Patterson and B. Bamieh, "Leader selection for optimal network coherence," in *2010 49th IEEE Conference on Decision and Control (CDC)*. IEEE, 2010, pp. 2692–2697.

[11] F. Lin, M. Fardad, and M. R. Jovanović, "Algorithms for leader selection in large dynamical networks: Noise-corrupted leaders," in *2011 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*. IEEE, 2011, pp. 2932–2937.

[12] S. Pequito, V. Preciado, and G. J. Pappas, "Distributed leader selection," in *2015 IEEE 54th Annual Conference on Decision and Control (CDC)*. IEEE, 2015, pp. 962–967.

[13] N. Takahashi, I. Yamada, and A. H. Sayed, "Diffusion least-mean squares with adaptive combiners: Formulation and performance analysis," *IEEE Transactions on Signal Processing*, vol. 58, no. 9, pp. 4795–4810, 2010.

[14] F. S. Cattivelli and A. H. Sayed, "Diffusion lms strategies for distributed estimation," *IEEE Transactions on Signal Processing*, vol. 58, no. 3, pp. 1035–1048, 2010.

Paper D – Energy-Efficient Distributed Leader Selection Algorithm for Energy-Constrained Wireless Sensor Networks

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 25, 2018, accepted December 8, 2018, date of publication December 18, 2018,
date of current version January 11, 2019.

Digital Object Identifier 10.1109/ACCESS.2018.2888551

Energy-Efficient Distributed Leader Selection Algorithm for Energy-Constrained Wireless Sensor Networks

SANDER ULP, YANNICK LE MOULLEC, AND MUHAMMAD MAHTAB ALAM
Thomas Johann Seebeck Department of Electronics, Tallinn University of Technology, 19086 Tallinn, Estonia
Corresponding author: Sander Ulp (sander.ulp@taltech.ee)

This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 668995 and partly supported by European Union Regional Development Fund in the framework of the Tallinn University of Technology Development Program 2016-2022. This material reflects only the authors view and the EC Research Executive Agency is not responsible for any use that may be made of the information it contains. This research was supported by the Information Technology Foundation for Education.

ABSTRACT In the context of green communication and energy-efficiency in wireless communication, this paper investigates distributed estimation algorithms in an energy-constrained wireless sensor network and proposes an energy-efficient distributed leader selection algorithm. The existing state-of-the-art diffusion algorithm and the recently introduced distributed leader selection algorithm are investigated. To evaluate the energy consumption of the algorithms, their respective algorithmic complexity, and number of operations and information exchanges are derived and compared. The obtained values are used as a basis to estimate the execution time and energy consumption of the algorithms. We propose and introduce the energy-efficient distributed leader selection algorithm which retains the performance of the existing leader selection algorithm while reducing the complexity and energy consumption. For the simulations, the algorithms are mapped to widely used wireless sensor network hardware architectures (MSP430 and RSL10). The numerical results show that the proposed algorithm is able to decrease the energy consumption of the network by 32%–53% and can extend the network lifetime by 14%–46% as compared with the diffusion and the distributed leader selection algorithms.

INDEX TERMS Energy-efficiency, distributed estimation, wireless sensor networks, distributed leader selection, diffusion.

I. INTRODUCTION

Recently, data traffic has had an exponential growth due to the rapid increase of the usage of wireless devices. With new applications on the horizon in the field of Internet-of-Things (IoT), the amount of energy consumed by wireless applications and systems is on the rise. This has put a lot of focus on energy-efficiency (EE) and green communication (GC) systems to reduce the amount of energy consumed [1], [2]. EE in wireless sensor networks (WSN) is an important topic that has received a lot attention due to the numerous applications that WSNs have in IoT [1]–[4]. Many recent works have aimed at reducing the amount of energy consumed by optimizing protocols and clustering on the physical and the network levels [5]–[7]. Similarly, hardware such as microcontrollers (MCUs) and radio modules are being optimized for low-power systems and applications [8]. The above approaches are especially important when the WSN is energy-constrained and the nodes have limited battery capacity to work with. In this work we focus on distributed estimation as one of the applications of WSNs and on the energy-efficiency of the underlying algorithms.

The idea behind distributed estimation is to solve a common task or to estimate some process in a network and by means of cooperation to enhance the performance of the network and of the nodes. Distributed estimation is a popular research topic in various interdisciplinary fields, such as agriculture, military, environmental studies and signal processing [9], [10]. We are interested in fully distributed solutions without any centralized processing unit. The benefits of fully distributed estimation include low energy consumption, low processing complexity and robustness [10].

Methods proposed for solving distributed estimation problems include incremental, consensus, diffusion and leader selection algorithms [10]–[12]. Incremental algorithms suffer

from a NP hard problem as a cyclic route through the network has to be defined for the algorithm to work [10]. Consensus algorithms are outperformed by diffusion algorithms and also have stability issues under infinitesimally small step-sizes [11]. Therefore, in this work we focus on the diffusion algorithm and the distributed leader selection algorithm. Diffusion algorithms have shown good performance as well as robustness [10], [13]. Distributed leader selection (DLS) has been proposed in our earlier work [12]. DLS, while not always able to outperform the diffusion algorithms, has the benefits of being simpler and more robust in terms of the secondary parameter estimation accuracy [14].

When considering a massive number of devices communicating in a distributed fashion in an energy-constrained situation, it is critical to have an extended lifetime for the network and thus, as a first step, we want to estimate the energy cost of the diffusion algorithm and of the DLS algorithm. The energy costs of the algorithms are obtained by using high-level estimation and are based on the number of operations required for the algorithms to run. Thus, we investigate the complexity of the aforementioned algorithms, as well as the required number of operations for both of the algorithms. It has been shown that by using high-level estimation it is possible to get accurate and meaningful estimations of the energy consumption of the algorithms [15]–[17]. The estimates are a good basis for knowing how the algorithms scale on real hardware architectures. We are also interested in further improving the energy-efficiency of the DLS algorithm. We analyse the energy cost of the DLS algorithm and propose a novel method to reduce its energy consumption. The resulting proposed energy-efficient distributed leader selection (EEDLS) algorithm is described and compared to the diffusion algorithm and the DLS algorithm. The results are verified by simulations. For the simulations, the algorithms are mapped onto widely used WSN hardware architectures (MSP430 and RSL10).

To summarize, the purpose of this work is to investigate and compare the diffusion algorithm and the DLS algorithm from the perspective of energy cost of the computations and radio communication, and propose a novel method to improve the energy-efficiency of the DLS algorithm.

The research contributions are detailed as follows:

- The analysis and comparison of the diffusion algorithm's and DLS algorithm's complexities and number of operations required. Previous works have outlined the number of operations required for the diffusion algorithm [11]; this work extends this by including the additional operations and complexity for the weight calculations and comparing it to the complexity and operations of the DLS algorithm. In addition, the impact of the size of the neighbourhood and of the LMS filter length on the number of operations required are explicitly considered.
- A novel method for reducing the energy consumption of the DLS algorithm by reducing the computations and radio communication. There exist works on reducing the energy consumption of the diffusion algorithm by modifying either the algorithm, the communication frequency, or the topology [18]–[21]. However, in these works additional complexity is introduced [18], [19] or the performance of the diffusion algorithm degrades [20], [21]. We propose a novel EE method for reducing the energy consumption of the DLS algorithm. In contrast to existing works, the proposed EEDLS algorithm is able to reduce the amount of required radio communications to the minimum and to reduce the complexity while retaining the performance of the DLS algorithm.
- A numerical simulation study of the computational and radio communication energy consumption of the diffusion, DLS, and the proposed EEDLS algorithms considering the MSP430 MCU and the RSL10 radio module. Previous works have focused on the diffusion algorithm's analytical side [22], [23] and, to the authors' best knowledge, there are no works that explore the physical energy requirements of the diffusion algorithms and that estimate the energy consumption on real hardware models. Simulation results of the EEDLS algorithm show that the network energy consumption is reduced by $32 - 53\%$ and the network lifetime is extended by $14 - 46\%$ as compared to the DLS algorithm and diffusion algorithm.

The remainder of the paper is organized as follows. In Section II we outline the diffusion algorithm and the DLS algorithm, as well as describe the problem setting. Section III gives an overview of the computational and radio communication energy consumption estimation and the network lifetime estimation. The complexity and number of operations required for the diffusion algorithm and DLS algorithm are given and compared. The computational impact of the neighbourhood and of the LMS adaptive filter length are described. The proposed EEDLS algorithm is introduced in Section IV. Section V presents the simulation setting and the numerical results for the diffusion, DLS, and EEDLS algorithms for different topologies. Section VI concludes the paper.

In this paper, italic letters are used for scalars (e, E) and lower case bold letters denote vectors ($\mathbf{x}$). All vectors are column vectors, except for the regression vector $\mathbf{u}_{k,i}$. The operator $E[\cdot]$ stands for mathematical expectation of the subject and (y^*) denotes the complex conjugate transpose of (y).

II. ENERGY-CONSTRAINED DISTRIBUTED ESTIMATION

In this section, we introduce the problem setting and outline the diffusion and DLS algorithm for the readers' better understanding of the analysis of the algorithms presented later on in Section III.

A. GENERAL MODEL

Assume that there are K nodes that are estimating a common parameter, signal source, or information about some target or object. Each of the nodes have access to $d_k(i)$ and

$\mathbf{u}_{k,i}$ at each time instant i:

$$d_k(i) = \mathbf{u}_{k,i}\mathbf{w}^o + v_k(i), \tag{1}$$

where $d_k(i)$ is the measurement at node k, $\mathbf{u}_{k,i}$ is the row regression vector, $\mathbf{w}^o$ is the unknown column vector and $v_k(i)$ is zero-mean white random noise with power $\sigma^2_{v,k}$.

$v_k(i)$ and $\mathbf{u}_{k,i}$ are assumed to be independent for all k values. The $\mathbf{u}_{k,i}$ has a positive-definite covariance matrix, $R_{u,k} = E\,\mathbf{u}^*_{k,i}\mathbf{u}_{k,i} > 0$. The nodes estimate $\mathbf{w}^o$ by minimizing the global cost function:

$$J_k(w) = E \mid d_k(i) - \mathbf{u}_{k,i}\mathbf{w} \mid^2 . \tag{2}$$

Each of the nodes employs the LMS algorithm for adaptation:

$$\mathbf{w}_{k,i} = \mathbf{w}_{k,i-1} + \mu_k \mathbf{u}^*_{k,i}[d_k(i) - \mathbf{u}_{k,i}\mathbf{w}_{k,i-1}], \tag{3}$$

where μ_k is a constant positive step-size such that, $0 < \mu_k < \frac{2}{\lambda_{max}(R_{u,k})}$.

In this manner the nodes in the network are working independently by employing a LMS filter locally to the data. The performance of the individual nodes and that of the network can be improved if the nodes cooperate and communicate in some manner [10]. As noted earlier, we are looking at fully distributed algorithms and we focus on the diffusion algorithm and the DLS algorithm as they have been shown to have good performance [14].

The K nodes form a WSN. Each of the nodes is able to communicate with a subset of nodes and with itself, which forms the node k-s neighbourhood $\mathbb{N}_k$. In addition, the nodes are energy-constrained which means that the energy available to the nodes is limited; the nodes have a certain amount of battery capacity C. During its lifetime, a node can carry out a certain amount of communications to its neighbours as well as a certain amount of computations, which is reflected by the amount of iterations the node can carry out before running out of energy and dying. The topology of the network is assumed to be strongly connected and the connections between the nodes are lossless and noiseless. The nodes are identical with respect to processing power and physical attributes. The nodes do not possess any *a priori* knowledge about other nodes and the network topology.

B. DIFFUSION

We outline the diffusion algorithm in this subsection. In the following two variants of the diffusion algorithm, each of the nodes calculate the estimates using an LMS filter. The estimates are shared with the node's neighbours from the neighbourhood $\mathbb{N}_k$. The nodes combine the estimates using weights $\alpha_{l,k}(i)$ which form the $\mathbf{A}$ matrix [24]. Since there are limitations to the energy available to the nodes, we do not consider exchanging the measurements between the nodes, i.e. the matrix for measurement weights is $\mathbf{C} = I$ and the measurements are only available to the node itself. Sharing the measurements can improve the performance of the diffusion algorithm, but will double the amount of information exchanges per iteration [24]. The combination step and adaptation step can be performed in different order, resulting in the ATC (adapt and then combine) and the CTA (combine and then adapt) variants of the diffusion algorithm which are given as Algorithms 1 and 2, respectively. The algorithms have identical computational complexity [11], but it has been shown that the ATC algorithm attains better performance and therefore, later on we consider the ATC algorithm [24].

Algorithm 1 ATC Diffusion

1: **for** each time instant $i > 0$:
2: each node $k = 1, 2, ..., K$ performs the update:
3: $\boldsymbol{\psi}_{k,i} = \mathbf{w}_{k,i-1} + \mu_k \mathbf{u}^*_{k,i}[d_k(i) - \mathbf{u}_{k,i}\mathbf{w}_{k,i-1}]$
4: $\mathbf{w}_{k,i} = \sum_{l\in\mathbb{N}_k} \alpha_{l,k}(i)\boldsymbol{\psi}_{l,i}$
5: **end**

Algorithm 2 CTA Diffusion

1: **for** each time instant $i > 0$:
2: each node $k = 1, 2, ..., K$ performs the update:
3: $\boldsymbol{\psi}_{k,i-1} = \sum_{l\in\mathbb{N}_k} \alpha_{l,k}(i)\mathbf{w}_{l,i-1}$
4: $\mathbf{w}_{k,i} = \boldsymbol{\psi}_{k,i-1} + \mu_k \mathbf{u}^*_{k,i}[d_k(i) - \mathbf{u}_{k,i}\boldsymbol{\psi}_{k,i-1}]$
5: **end**

There exist several methods for calculating the weights $\alpha_{l,k}(i)$ in the $\mathbf{A}$ matrix, such as Metropolis, uniform, relative degree etc. [24]–[26]. The network performance can be further improved by assigning weights based on the situation of the nodes, which is related to some secondary parameter [10]. The noise power $\sigma^2_{v,k}$ can be used as a secondary parameter to calculate the weights. In practice, the noise powers are not usually known and have to be estimated. Works on estimating noise powers at different nodes are available in e.g. [12], [27], [28]. In this work, we assume that the nodes have *a priori* knowledge of their noise powers as the calculations for the secondary parameter estimation would be identical for the diffusion algorithm and DLS algorithm. The relative variance rule and relative degree-variance rule weight calculations for the diffusion algorithms have shown good and similar performance and are given in (5) and (5) [27].

$$\alpha_{l,k}(i) = \begin{cases} \dfrac{\sigma^{-2}_{v,l}}{\sum\limits_{l\in\mathbb{N}_k} \sigma^{-2}_{v,l}}, & \text{if } l \in \mathbb{N}_k \\ 0, & \text{if } l \notin \mathbb{N}_k, \end{cases} \tag{4}$$

$$\alpha_{l,k}(i) = \begin{cases} \dfrac{n_l\sigma^{-2}_{v,l}}{\sum\limits_{l\in\mathbb{N}_k} n_l\sigma^{-2}_{v,l}}, & \text{if } l \in \mathbb{N}_k \\ 0, & \text{if } l \notin \mathbb{N}_k, \end{cases} \tag{5}$$

where $\alpha_{l,k}(i)$ is the weight calculated at node k for neighbouring node l, i is the current iteration, $\sigma^2_{v,k}$ is the noise power at node k, $\mathbb{N}_k$ is the neighbourhood of node k and n_k is the number of neighbours of the node plus the node itself.

The diffusion algorithm including the weight calculation based on the relative variance rule is given as Algorithm 3.

Algorithm 3 Diffusion Algorithm

1: **for** each time instant $i > 0$:
2: each node $k = 1, 2, ..., K$ performs the update:
3: $\boldsymbol{\psi}_{k,i} = \mathbf{w}_{k,i-1} + \mu_k \mathbf{u}_{k,i}^{*}[d_k(i) - \mathbf{u}_{k,i}\mathbf{w}_{k,i-1}]$
4: **for** each $l \in \mathbb{N}_k$
5: $\alpha_{l,k}(i) = \frac{\sigma_{v,l}^{-2}}{\sum_{l\in\mathbb{N}_k} \sigma_{v,l}^{-2}}$
6: **end**
7: $\mathbf{w}_{k,i} = \sum_{l\in\mathbb{N}_k} \alpha_{l,k}(i)\boldsymbol{\psi}_{l,i}$
8: **end**

C. DISTRIBUTED LEADER SELECTION

In contrast to the diffusion algorithms, the nodes in the DLS algorithm determine a leader node in the network. The DLS has been previously introduced in [12] and [14]. The nodes only have access to the information from the neighbours from the neighbourhood $\mathbb{N}_k$ and select the node which corresponds to the best performing neighbour c_k. The estimates from this node are used and distributed along with the corresponding weight (Algorithm 4). In this manner, the nodes converge to local leaders in their neighbourhood and then to a global leader. A secondary parameter is required to select the leader. Whether this secondary parameter is known or estimated, the nodes are able to use its information to determine which of the nodes is the best performing one and follow its lead. In this work, as for the diffusion algorithm, the secondary parameter is the noise power at a node and is assumed to be known by the node (see Subsection II-B).

The nodes exchange the estimate $\mathbf{e}_{k,i}$ between themselves, together with the corresponding weight $\alpha_k(i)$ that is assigned to the estimate at iteration i. Unlike the diffusion algorithm where the communication paths are weighted, the weights of the DLS algorithm correspond to the nodes themselves. The nodes compare their weight to that of their neighbouring nodes in $\mathbb{N}_k$ and if they have the highest weight, they become the leader node (Algorithm 4 line 4). Otherwise they become follower nodes, in which case they listen to their best neighbour c_k (Algorithm 4 line 8). $\mathbf{e}_{k,i}$ is the estimation result that is used by the node.

III. ENERGY CONSUMPTION ESTIMATION

In this section, the energy consumption models and estimation method are described. The complexity and the required number of operations for the diffusion algorithm and DLS algorithm are derived.

A. NETWORK ENERGY CONSUMPTION

We estimate the energy consumed by a sensor node by following the model proposed and used in [29] and [30]. The energy E consumed by each sensor is given as:

$$\frac{E}{V} = I_a t_a + I_l t_l + I_t t_t + I_r t_r + \sum I_c t_c, \tag{6}$$

where V is the voltage of the power supply. I_a and t_a are the current drawn and execution time of the MCU (Microprocessor Control Unit) in the active mode. I_l and t_l are the current drawn and execution time of the MCU in the low power mode. I_t, I_r, t_r and t_r are the currents drawn and execution times of the radio in transmit and receive modes, respectively. I_c and t_c are the currents drawn and execution time of other components.

Since we are interested in the comparison and the performance of the algorithms, we simplify the formula by focusing only on the active, transmit and receive modes. The energy consumption for the low power mode and for the other components are assumed to be the same for both algorithms. We also have to take into account that the computational times, transmit and receive times for each node are unique due to the topology and how many neighbours the node communicates with (see Subsection II-A). We note that if the amount of neighbours change during estimation, the value changes according to the iteration i. We then can write for node k at iteration i:

$$\frac{E_k(i)}{V} = I_a t_{a,k}(i) + I_t t_{t,k}(i) + I_r t_{r,k}(i) \tag{7}$$

and the energy consumption for node k at iteration i:

$$\begin{aligned} E_k(i) &= V(I_a t_{a,k}(i) + I_r t_{r,k}(i) + I_t t_{t,k}(i)) \\ &= E_{comp,k}(i) + E_{radio,k}(i). \end{aligned} \tag{8}$$

Node k uses $E_k(i)$ energy per iteration i. $E_{radio,k}(i)$ is the communication energy consumption of node k and the computational energy consumed by node k at iteration i is $E_{comp,k}(i)$. The energy consumed by the network in one iteration i is given as:

$$E_{network}(i) = \sum_{k=1}^{K} E_k(i). \tag{9}$$

The energy consumed by the network during the observation period i_{obs} is given as:

$$E_{obs} = \sum_{i=1}^{i_{obs}} E_{network}(i). \tag{10}$$

Algorithm 4 Distributed Leader Selection

1: **for** each time instant $i > 0$:
2: each node $k = 1, 2, ..., K$ performs the update:
3: $\mathbf{w}_{k,i} = \mathbf{w}_{k,i-1} + \mu_k \mathbf{u}_{k,i}^{*}[d_k(i) - \mathbf{u}_{k,i}\mathbf{w}_{k,i-1}]$
4: **if** $\alpha_k(i-1) \geq \max_{k\in\mathbb{N}_k}(\alpha_k(i-1))$
5: $\mathbf{e}_{k,i} = \mathbf{w}_{k,i}$
6: $\alpha_k(i) = (1-\mu_k)\alpha_k(i-1) + \mu_k \sigma_{v,k}^{-2}$
7: **else**
8: $c_k = \operatorname{argmax}_{k\in\mathbb{N}_k}(\alpha_k(i-1))$
9: $\mathbf{e}_{k,i} = \mathbf{e}_{c_k,i}$
10: $\alpha_k(i) = (1-\mu_k)\alpha_{c_k}(i-1) + \mu_k \sigma_{v,k}^{-2}$
11: **end**
12: **end**

B. COMPUTATIONAL ENERGY CONSUMPTION

The computational energy consumption of the algorithms can be calculated through the complexity and the number of operations. From these values we can deduct the execution time needed for the computations and the energy consumed. It has been shown that calculating the energy consumption based on the number of operations can be used as a good benchmark [31]. We also note that the current drawn by the MCU is considered constant during different operations [29]. Therefore, the computational energy consumption from node to node differs in the execution time, which in turn is dependent on the number of operations the node has to carry out.

The complexities for the diffusion algorithm (Subsection II-B) and DLS algorithm (Subsection II-C) per node are given in Tables 1, 2 and 3. The complexities of the LMS and diffusion step have been derived in earlier works, but the complexity for calculating the weights had not been included [11]. M is the size of the LMS filter and n_k is the size of the neighbourhood. To further reduce the computational complexity, the relative variance method is used since the relative degree-variance method would introduce additional complexity due to additional multiplications in the weight calculations (4), (5). For the DLS, the complexities of the leader node and of a follower node are given separately as the nodes have different complexities, see Tables 2 and 3. We can see that the complexity of the node employing the diffusion algorithm is higher than that of the node employing the DLS algorithm, which makes the latter less complex.

The computational energy required for one iteration i at node k can be calculated as:

$$E_{comp,k}(i) = I_c V t_{c,k}(i), \tag{11}$$

where $t_{c,k}(i)$ is the execution time for node k at iteration i in seconds, V is the supply voltage and I_c is the current consumed by the MCU in a second.

The execution time for the operations at node k at iteration i is given as:

$$t_{c,k}(i) = f(\#O_{m,k}(i)C_m + \#O_{d,k}(i)C_d + \#O_{a,k}(i)C_a + \#O_{c,k}(i)C_c), \tag{12}$$

where f is the frequency of the MCU, $\#O_{m,k}(i)$, $\#O_{d,k}(i)$, $\#O_{a,k}(i)$, $\#O_{c,k}(i)$ are the number of multiplications, divisions, additions and comparisons, respectively, and C_m, C_d, C_a, C_c are the number of clock cycles required for multiplication, division, addition and comparisons, respectively.

The multiplications, divisions, additions, comparisons and vector exchanges for the diffusion algorithm and the DLS algorithm are given in Table 4. We can see that the number of operations required for the diffusion algorithm increase substantially in comparison to the DLS algorithm when the filter size M increases (Fig. 1a) or when the neighbourhood of the node increases (Fig. 1b). Therefore, for applications with larger filter lengths and more connected networks, the DLS algorithm would be preferred.

TABLE 1. Diffusion algorithm complexity.

Diffusion	Complexity
$\psi_{k,i} = \mathbf{w}_{k,i-1} + \mu_k \mathbf{u}_{k,i}^*[d_k(i) - \mathbf{u}_{k,i}\mathbf{w}_{k,i-1}]$	$\mathcal{O}(2M)$
for each $l \in \mathbb{N}_k$ $\alpha_{l,k}(i) = \frac{\sigma_{v,l}^{-2}}{\sum_{l\in\mathbb{N}_k}\sigma_{v,l}^{-2}}$ **end**	$\mathcal{O}(n_k)$
$\mathbf{w}_{k,i} = \sum_{l\in\mathbb{N}_k} \alpha_{l,k}(i)\psi_{l,i}$	$\mathcal{O}(n_k M)$
	$\mathcal{O}(2M + n_k + n_k M)$

TABLE 2. DLS (leader node) algorithm complexity.

DLS (leader node)	Complexity
$\mathbf{w}_{k,i} = \mathbf{w}_{k,i-1} + \mu_k \mathbf{u}_{k,i}^*[d_k(i) - \mathbf{u}_{k,i}\mathbf{w}_{k,i-1}]$	$\mathcal{O}(2M)$
if $\alpha_k(i-1) \geq \max_{k\in\mathbb{N}_k}(\alpha_k(i-1))$	$\mathcal{O}(n_k)$
$\mathbf{e}_{k,i} = \mathbf{w}_{k,i}$	$\mathcal{O}(1)$
$\alpha_k(i) = (1-\mu_k)\alpha_k(i-1) + \mu_k\sigma_{v,k}^{-2}$	$\mathcal{O}(2)$
	$\mathcal{O}(2M + n_k + 3)$

TABLE 3. DLS (follower node) algorithm complexity.

DLS (follower node)	Complexity
$\mathbf{w}_{k,i} = \mathbf{w}_{k,i-1} + \mu_k \mathbf{u}_{k,i}^*[d_k(i) - \mathbf{u}_{k,i}\mathbf{w}_{k,i-1}]$	$\mathcal{O}(2M)$
if $\alpha_k(i-1) \geq \max_{k\in\mathbb{N}_k}(\alpha_k(i-1))$	$\mathcal{O}(n_k)$
$c_k = \operatorname{argmax}_{k\in\mathbb{N}_k}(\alpha_k(i-1))$	$\mathcal{O}(n_k)$
$\mathbf{e}_{k,i} = \mathbf{e}_{c_k,i}$	$\mathcal{O}(1)$
$\alpha_k(i) = (1-\mu_k)\alpha_{c_k}(i-1) + \mu_k\sigma_{v,k}^{-2}$	$\mathcal{O}(2)$
	$\mathcal{O}(2M + 2n_k + 3)$

TABLE 4. Operations for diffusion and DLS.

Operation	Diffusion	DLS	
		Follower	Leader
Multiplications	$2M + Mn_k$	$2M + 2$	$2M + 2$
Divisions	n_k	$-$	$-$
Additions	$Mn_k + M + n_k$	$M + 2$	$M + 2$
Comparisons	$-$	$2n_k$	n_k
Vector exchanges	n_k	n_k	n_k

To further illustrate this point, the amount of operations required for one node in the diffusion network and the DLS network are given in an example (Fig. 1c). While the filter length M is the same for all the nodes, the number of neighbours n_k is different from node to node and nodes with larger numbers of neighbours require more operations. We can see that the number of operations for the diffusion algorithm's nodes depend more on the topology and will vary to a larger degree than for the DLS algorithm nodes.

C. RADIO COMMUNICATION ENERGY CONSUMPTION

The radio communication energy consumed $E_{radio,k}(i)$ by node k at iteration i is given as:

$$E_{radio,k}(i) = V(I_r t_{r,k}(i) + I_t t_{t,k}(i)), \tag{13}$$

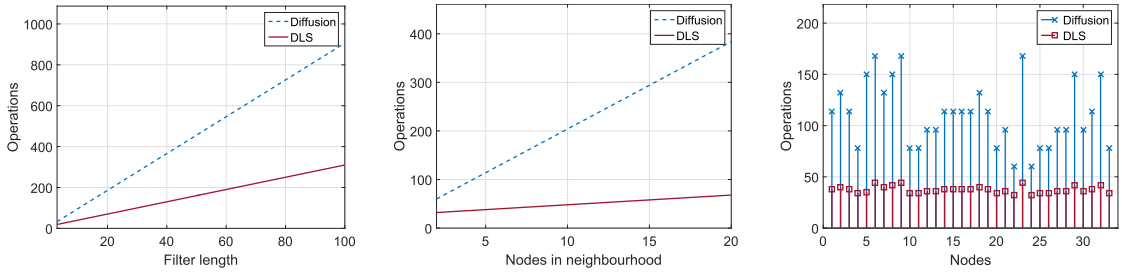


FIGURE 1. (a) Number of operations based on the length of the LMS filter. (b) Number of operations based on the number of nodes in the neighbourhood. (c) Number of operations per node of the diffusion algorithm and the DLS algorithm.

where I_r and I_t are the currents drawn by the receiver and transmitter, respectively, and $t_{r,k}(i)$ and $t_{t,k}(i)$ are the communication periods for the receiver and transmitter at node k at iteration i, respectively.

Each of the nodes communicates with each of its neighbours once per iteration. Nodes with more connections consume more energy. The vector exchanges per iteration are given as n_k in Table 4. It is important to note that the value n_k includes the self-loop i.e. the vector exchange with the node itself. Since the node does not consume energy communicating with itself, the amount of radio communications required is $n_k - 1$. We can write the energy consumed by each communication path as $V(I_r t_r + I_t t_t)$ and write the communication energy consumed at node k at iteration i as:

$$E_{radio,k}(i) = (n_k(i) - 1)V(I_r t_r + I_t t_t), \tag{14}$$

where $n_k(i) - 1$ is the number of neighbours for node k.

We can further write that the network radio consumption can be given as:

$$\begin{aligned} E_{radio}(i) &= \left(\sum_{k=1}^{K}(n_k(i) - 1)\right) V(I_r t_r + I_t t_t) \\ &= (N(i) - K)V(I_r t_r + I_t t_t), \end{aligned} \tag{15}$$

where $N(i) - K$ is the amount of connections without the self-loops in the network at iteration i.

D. NETWORK LIFETIME

The network lifetime can be computed by taking into account the computational and the communication energy consumptions. We refer to the method proposed in [32]. Each node is assumed to have the same battery capacity C. The lifetime for a sensor node k can be computed as:

$$Lt_k = \frac{C}{\overline{I_k}}, \tag{16}$$

where C is the battery capacity in mAh, Lt_k is the lifetime of the node k, $\overline{I_k}$ is average current consumption.

Since we are also interested in the amount of iterations the nodes are able to carry out before running out of battery energy and dying, we modify the formula as:

$$LI_k = \frac{C}{\overline{E}_k} \tag{17}$$

where LI_k is the lifetime of the node k in number of iterations and $\overline{E}_k$ is the average energy consumed by node k.

IV. ENERGY-EFFICIENT DISTRIBUTED LEADER SELECTION

In this section, the proposed EEDLS algorithm is introduced. The algorithm is based on the DLS algorithm introduced in Subsection II-C.

A. PROPOSED ALGORITHM

The goal of the proposed EEDLS is to reduce the energy consumed by the nodes in the DLS algorithm. Under the assumption that the energy saving mode duration has been selected appropriately so that during this duration the leader node does not change, we can reduce the amount of connections and computations. In this manner we are able to use the energy more efficiently and extend the lifetime of the network. The basis of this energy reduction is the vector $\mathbf{c}$ which contains the neighbours of all the nodes that they follow, see Algorithm 4. Based on this vector we are able to reduce the amount of connections and the traffic required in the network and modify the topology to reduce the communication energy consumption. Since the nodes only use the information from the best performing neighbour, communication to the other nodes is redundant and connections only remain between the nodes which are used to spread the information across the network.

Working under the assumption that during the energy saving mode the leader node does not change, we can determine the leader in the network and then discontinue its selection. Based on this, we are able to reduce the number of computations. In addition, since the computations from the follower nodes are not required and only computations from the leader node are required, we can reduce the computational complexity of the algorithm as well. These reductions do not impact the MSD (mean squared deviation) performance of the EEDLS algorithm, which can be seen in Fig. 2a. Therefore, the MSD performance of the EEDLS algorithm and DLS

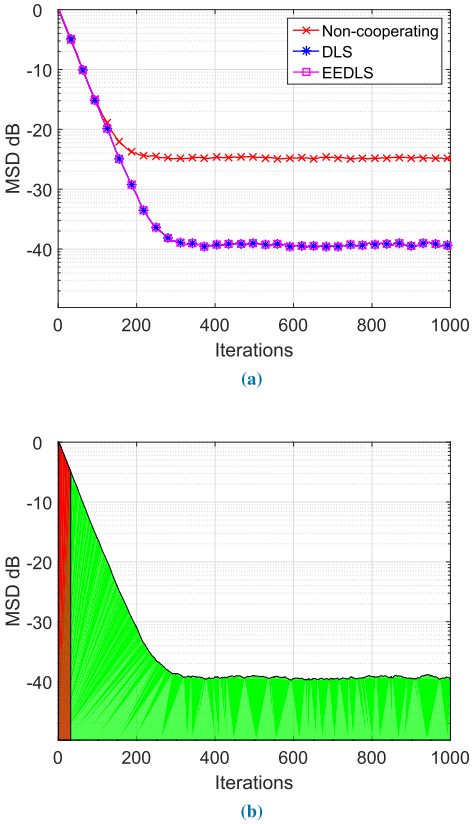


FIGURE 2. (a) MSD network performance of the non-cooperating algorithm, DLS algorithm and the EEDLS algorithm. (b) The EEDLS algorithms leader selection mode (red) and energy saving mode (green).

algorithm are identical; for reference, the performance of the DLS algorithms has been shown previously in [12] and [14].

In order to obtain the vector $\mathbf{c}$, the network has to run and select the leader, which can be accomplished after a certain number of iterations. The amount of iterations required for this depends on the topology of the network and the amount of nodes. The maximum amount of iterations for the leader selection is $i_{ls} = K - 1$ which is the longest path in the network with K nodes if all the nodes in the network are in serial connections. This is the maximum amount of iterations required for the leader node to be selected for the worst case scenario network. In other cases, the leader would be selected in lesser amount of iterations. For example, in the star topology the leader node would be selected in one iteration.

The information about the amount of nodes in the network might not be available beforehand, but during the leader selection time this can be communicated or estimated between the nodes. In this work, we assume that the nodes are able to communicate it across the network.

During the leader selection mode, the algorithm performs as the DLS algorithm (Fig. 2b (red)). After the leader has been determined, the network enters the energy saving mode where redundant connections are disabled and the computational complexity is decreased (Fig. 2b (green)). The leader node continues the adaptation step (Algorithm 5) and the follower nodes follow the leader nodes estimations. The follower nodes relay the information from the leader across the network (Algorithm 6) and the leader continues its estimation until the observation period ends.

Algorithm 5 EEDLS (Leader Node)

1: **for** each time instant $i > i_{ls}$:
2: $\quad \mathbf{w}_{k,i} = \mathbf{w}_{k,i-1} + \mu_k \mathbf{u}_{k,i}^{*}[d_k(i) - \mathbf{u}_{k,i}\mathbf{w}_{k,i-1}]$
3: $\quad \mathbf{e}_{k,i} = \mathbf{w}_{k,i}$
4: **end**

Algorithm 6 EEDLS (Follower Node)

1: **for** each time instant $i > i_{ls}$:
2: $\quad \mathbf{e}_{k,i} = \mathbf{e}_{c_k,i}$
3: **end**

B. COMPUTATIONAL ENERGY REDUCTION

The complexity of the computations in the nodes is reduced after i_{ls} iterations and is given in Table 5. Before $(i_{ls} + 1)$ iterations, the complexity of the algorithm is that of the DLS algorithm as given in Tables 2 and 3. The leader node resumes the LMS adaptation step and the follower nodes only receive and send information to other nodes. The number of operations required is given in Table 6. The size of the neighbourhood n_k does no longer affect the amount operations required for one iteration and the length of the filter M only impacts the amount of operations required for the leader node.

TABLE 5. The EEDLS algorithm's leader and follower node complexities in the energy saving mode.

EEDLS (leader)	Complexity
$\mathbf{w}_{k,i} = \mathbf{w}_{k,i-1} + \mu_k \mathbf{u}_{k,i}^{*}[d_k(i) - \mathbf{u}_{k,i}\mathbf{w}_{k,i-1}]$	$\mathcal{O}(2M)$
$\mathbf{e}_{k,i} = \mathbf{w}_{k,i}$	$\mathcal{O}(1)$
EEDLS (follower)	**Complexity**
$\mathbf{e}_{k,i} = \mathbf{e}_{c_k,i}$	$\mathcal{O}(1)$

TABLE 6. The EEDLS algorithm's operations for the leader and follower node in the energy saving mode.

Operation	EEDLS leader	EEDLS follower
Multiplications	$2M + 1$	–
Divisions	–	–
Additions	2	–
Comparators	–	–
Vector exchanges	$[2, n_k]$	$[2, n_k]$

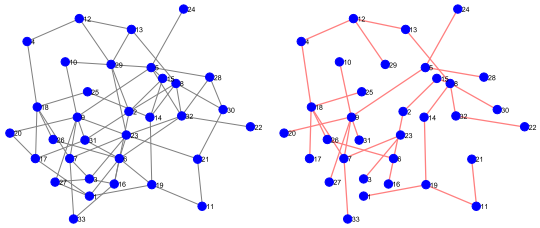
C. RADIO COMMUNICATION ENERGY REDUCTION

We switch off the communication that is redundant for the DLS algorithm (Algorithm 5, 6). The selection of the communication paths is determined by the vector **c** which holds the connections that are required to maintain the performance of the algorithm.

The resulting amount of connections might not decrease for every node; the amount of vector exchanges for one node is given in Table 6. The minimum number of vector exchanges is 2 when the node is communicating to one node and to itself. The minimum number of radio communications required is 1 as the node does not need radio communication for the vector exchange with itself. If all the connections of a node are needed for the network connectivity or to maintain the performance of the algorithm, then the resulting amount of vector exchanges is n_k and radio communications $n_k - 1$. Therefore, the radio communications for one node depend on the topology of the network, as can be seen from the example in Fig. 3. Overall, the amount of radio communications in the network is reduced to the minimum amount of communications possible to retain the connectivity which is $N_{EEDLS} = K - 1$.

$$N_{EEDLS} = \sum_{k=1}^{K} (n_k(i_{ls} + 1) - 1) = K - 1, \tag{18}$$

where $n_k(i_{ls} + 1) - 1$ is the amount of neighbours for node k in the energy saving mode.



(a) (b)

FIGURE 3. (a) Topology of the network. (b) The EEDLS algorithm topology with reduced connections.

The communication energy consumed by the network after i_{ls} iterations can be computed as:

$$E_{radio}(i_{ls} + 1) = (K - 1)V(I_r t_r + I_t t_t). \tag{19}$$

The network radio communication energy savings during the energy saving mode of the reduced topology can be expressed through (15) as:

$$\begin{aligned} E_{\Delta radio} &= ((N(i) - K) - (K - 1))V(I_r t_r + I_t t_t) \\ &= (N(i) - 2K + 1)V(I_r t_r + I_t t_t). \end{aligned} \tag{20}$$

V. SIMULATION RESULTS

In this section we present the simulation setup and the numerical results obtained from the simulations for the diffusion, DLS, and EEDLS algorithms.

A. SIMULATION SETUP

For the simulations different sensor networks were randomly generated with different number of nodes and connections, which are given in Table 9. All of the generated networks are strongly connected. We take a closer look at the network with $K = 33$ nodes and $N = 93$ connections which can be seen in Fig. 3a. The networks employ the diffusion algorithm, the DLS algorithm and the EEDLS algorithm (which were introduced in Subsections II-B, II-C and IV-A respectively). The EEDLS topology can be seen from Fig. 3b where the redundant connections have been disconnected. Each of the nodes in the networks employ a TI MSP430 family microprocessor [33]. The TI MSP430 has been selected as it is a widely used low-power MCU in WSNs as can be seen from recent examples in the literature [34]–[36]. The ON Semiconductor RSL10 Ultra-Low-Power Multi-protocol Bluetooth Radio SOC (system on a chip) [37] has been selected as the radio module.

TABLE 7. Simulation parameters.

$C = 3000mAh$	$K = 33$
$I_{Rx} = 3mA$	$T_{Rx} = 2.5ms$
$I_{Tx} = 4,6mA$	$T_{Tx} = 2.5ms$
$I_c = 3.7mA$	$N = 93$
$U = 3V$	$\mu = 0.01$
$f = 16Mhz$	$M = 8$

The parameters for the simulations are given in Table 7. The radio communication current values are used as I_{Rx} and I_{Tx} at supply voltage $U = 3V$ [37]. Communication times for sending and receiving are given as T_{Rx} and T_{Tx} [38]. For the computational current, the value I_c is used [38]. The battery capacity for each of the nodes is C. The clock frequency of the MCU is given as f [33]. The LMS filter length for the simulations is selected as $M = 8$ and the step size is selected for all nodes as $\mu = 0.01$.

B. COMPUTATIONAL ENERGY CONSUMPTION

The required clock cycles based on the MSP430 architecture for each of the operations are given in Table 8 [33], [39]. We see that the largest number of clock cycles are required for the division operation [39], which makes the diffusion algorithm computationally heavy as the weight calculations for the algorithm require n_k division for each iteration (Table 4).

TABLE 8. MSP430 clock cycles for different operations.

Operation	Clock cycles
Multiplications	13
Divisions	21
Additions	6
Comparisons	6

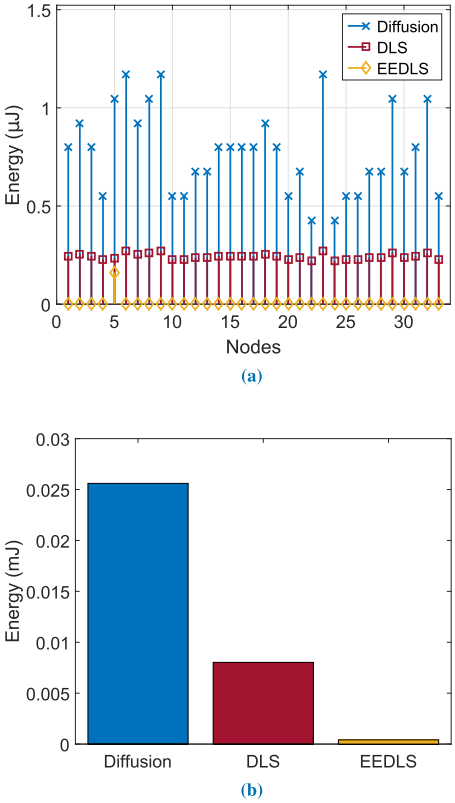


FIGURE 4. (a) Computational energy consumption at different nodes for the diffusion algorithm, DLS algorithm and EEDLS algorithm. (b) Network computational energy consumption for the diffusion algorithm, DLS algorithm and EEDLS algorithm.

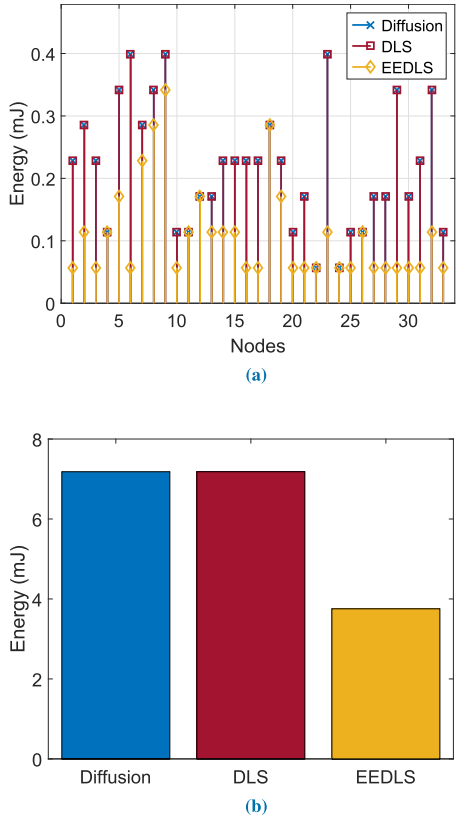


FIGURE 5. (a) Radio communication energy consumption at different nodes for the diffusion algorithm, DLS algorithm and EEDLS algorithm. (b) Network radio communication energy consumption for the diffusion algorithm, DLS algorithm and EEDLS algorithm.

We can see from Fig. 4a that the computational energy consumption for the diffusion algorithm varies from node to node as the computational amount is impacted by the amount of neighbours n_k of node k. The DLS computations vary to a lesser degree from node to node. The EEDLS algorithm in the energy saving mode attains the lowest energy consumption, requiring only the leader node to carry out operations on the MCU. From Fig. 4b we can see that the DLS algorithm consumes 68% less computational energy than the diffusion algorithm on the network level for this example. The EEDLS algorithm is further able to reduce the network computational energy by 98% as compared to the diffusion algorithm.

C. RADIO COMMUNICATION ENERGY CONSUMPTION

The radio energy consumption for the DLS algorithm and the diffusion algorithm are identical as the amounts of communication and vector exchanges are identical. From Fig. 5a we can see that the EEDLS algorithm is able to reduce the radio communication energy for each node compared to the DLS algorithm and the diffusion algorithm. The amount of radio communication energy reduced by the EEDLS algorithm depends on the topology as the amount of radio communications at node k can remain unchanged if the connections are required for the other nodes to retain connectivity to the network (Table 6). The overall network radio communication energy consumption is reduced by 47% (Fig. 5b) as the number of connections in the network has been reduced.

D. OVERALL REDUCED ENERGY CONSUMPTION

In addition to the earlier example ($K = 33$ nodes and $N = 93$ connections) we analyse the results for the two larger randomly generated sensor networks with $K = 206$ and $K = 510$, which can be seen in Table 9. We compare the different topologies employing different algorithms based on the values given in the table. The averaged values have been calculated by averaging the results over 1000 iterations. The network energy has been calculated by (9). The network lifetime in number of iterations has been calculated by considering the first node that runs out of energy and dies based on (17). The average lifetime in iterations is calculated over all the nodes in the network.

TABLE 9. Results for different topologies.

	Diffusion	DLS	EEDLS	Diffusion	DLS	EEDLS	Diffusion	DLS	EEDLS
Nodes	33	33	33	206	206	206	510	510	510
Connections	93	93	93	613	613	613	1530	1530	1530
Average Computational Energy (μJ)	**0.7759**	**0.2432**	**0.0125**	**1.0408**	**0.2618**	**0.0543**	**1.0401**	**0.2617**	**0.1334**
Average Radio Communication Energy (mJ)	0.2176	0.2176	**0.1138**	0.3392	0.3392	**0.1595**	0.3389	0.3389	**0.2281**
Average Energy per Iteration (mJ)	0.2184	0.2179	**0.1138**	0.3403	0.3395	**0.1596**	0.3399	0.3391	**0.2283**
Network Energy (mJ)	7.2076	7.1900	**3.7542**	70.0964	69.9359	**32.8694**	173.3544	172.9575	**116.4101**
Network Lifetime (Iterations)	22490	22541	**26310**	9264	9284	**17475**	9843	9864	**15483**
Average Node Lifetime (Iterations)	53318	534650	**111514**	32847	32929	**113014**	34003	34088	**109368**

We see that the diffusion algorithm's average computational energy consumption is the highest and that the DLS algorithm consumes 68% less energy in the worst case and 74% less in the best case as compared to the diffusion algorithm. The EEDLS algorithm is improved on top of this and is able to further reduce the amount of computational energy required. Compared to the diffusion algorithm the reduction is 98% in the best case and 87% in the worst case. From the average radio communication energy consumption, we see that the DLS algorithm and the diffusion algorithm consume the same amount of energy, as expected. The EEDLS algorithm is able to reduce the radio communication energy consumption by 32% in the worst case and 52% in the best case.

The average energy consumption numbers are quite similar to the radio communication energy consumption values as the impact of the computational energy consumption is marginal in the overall energy consumption. The DLS algorithm consumes less energy than the diffusion algorithm, but only by a slight margin and it can be said that there is no advantage between the two algorithms if we consider both the computational energy consumption and the radio communication energy consumption. Given the above, the EEDLS algorithm is able to reduce the average energy consumption per iteration by 32% in the worst case and by 53% in the best case. The network energy consumption values show the same improvements for the EEDLS algorithm compared to the other algorithms.

The average node lifetime and the network lifetime difference between the DLS and the diffusion algorithm are similar with marginal improvements for the DLS algorithm. The EEDLS algorithm is able to improve the network lifetime in the worst case by 14% and in the best case by 46%. As noted before, if the topology includes nodes that have multiple connections which cannot be disconnected and are needed to retain the performance or the connectivity of some the nodes in the network, the lifetime of these nodes is not improved as much as the other nodes' lifetime in the network. This is evident from the average node lifetime as it improves under the EEDLS algorithm in the worst case by 52% and in the best case by 70%.

Overall, the EEDLS algorithm notably improves the energy-efficiency of the network in every aspect and in some areas by quite large margins. Comparing different network sizes illustrates that the EEDLS algorithm is able to improve the energy-efficiency of all the networks and for larger networks with more connections the improvements are greater.

VI. CONCLUSION

In this work, we started by investigating the computational complexity of the diffusion algorithm and the DLS algorithm. We found that the DLS algorithm is less complex in terms of computations. Furthermore, the DLS algorithm requires less operations and is preferred in applications where the network is more densely connected or longer adaptive filter lengths are required. In addition, we analysed the energy consumption for both of the algorithms taking into account energy-constrained conditions. Whereas the computational efficiency is better for the DLS algorithm in comparison to the diffusion algorithm, the radio communication energy consumption for both algorithms makes the computational energy savings negligible. To further reduce the overall energy consumption we proposed EEDLS, a new energy-efficient distributed leader selection algorithm, which reduces the amount of computations and radio communication in the network while retaining the performance of the DLS algorithm. In the simulation section we demonstrated the energy consumption and illustrated the energy savings of the proposed EEDLS algorithm on the TI MSP430 microcontroller family architecture and on the On Semiconductor RSL10 Bluetooth radio module. We are able to reduce the network energy consumption compared to the diffusion algorithm and the DLS algorithm by 32% in the worst case and 53% in the best case. We are able to extend the network lifetime by 14% in the worst case and 46% in the best case compared to the diffusion and the DLS algorithm. The increase of the average lifetime of a node is 52% in the worst case and 70% in the best case.

REFERENCES

[1] K. Wang, Y. Wang, Y. Sun, S. Guo, and J. Wu, "Green industrial Internet of Things architecture: An energy-efficient perspective," *IEEE Commun. Mag.*, vol. 54, no. 12, pp. 48–54, Dec. 2016.

[2] Z. Sheng, C. Mahapatra, C. Zhu, and V. C. M. Leung, "Recent advances in industrial wireless sensor networks toward efficient management in IoT," *IEEE Access*, vol. 3, pp. 622–637, 2015.

[3] R. Mahapatra, Y. Nijsure, G. Kaddoum, N. U. Hassan, and C. Yuen, "Energy efficiency tradeoff mechanism towards wireless green communication: A survey," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 686–705, May 2016.

[4] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Comput. Netw.*, vol. 52, no. 12, pp. 2292–2330, Aug. 2008.

[5] Y. Yao, Q. Cao, and A. V. Vasilakos, "EDAL: An energy-efficient, delay-aware, and lifetime-balancing data collection protocol for heterogeneous wireless sensor networks," *IEEE/ACM Trans. Netw.*, vol. 23, no. 3, pp. 810–823, Jun. 2015.

[6] J. S. Leu, T. H. Chiang, M. C. Yu, and K. W. Su, "Energy efficient clustering scheme for prolonging the lifetime of wireless sensor network with isolated nodes," *IEEE Commun. Lett.*, vol. 19, no. 2, pp. 259–262, Feb. 2015.

[7] S. Rani, R. Talwar, J. Malhotra, S. H. Ahmed, M. Sarkar, and H. Song, "A novel scheme for an energy efficient Internet of Things based on wireless sensor networks," *Sensors*, vol. 15, no. 11, pp. 28603–28626, 2015.

[8] T. Rault, A. Bouabdallah, and Y. Challal, "Energy efficiency in wireless sensor networks: A top-down survey," *Comput. Netw.*, vol. 67, pp. 104–122, Jul. 2014.

[9] A. H. Sayed, S.-Y. Tu, J. Chen, X. Zhao, and Z. J. Towfic, "Diffusion strategies for adaptation and learning over networks: An examination of distributed strategies and network behavior," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 155–171, May 2013.

[10] A. H. Sayed, "Adaptation, learning, and optimization over networks," *Found. Trends Mach. Learn.*, vol. 7, nos. 4–5, pp. 311–801, 2014.

[11] S.-Y. Tu and A. H. Sayed, "Diffusion strategies outperform consensus strategies for distributed estimation over adaptive networks," *IEEE Trans. Signal Process.*, vol. 60, no. 12, pp. 6217–6234, Dec. 2012.

[12] S. Ulp and T. Trump, "Leader selection in cooperative network based on MDL subspace algorithm for cognitive radio," in *Proc. 50th Asilomar Conf. Signals, Syst. Comput.*, Nov. 2016, pp. 704–708.

[13] C. G. Lopes and A. H. Sayed, "Diffusion least-mean squares over adaptive networks: Formulation and performance analysis," *IEEE Trans. Signal Process.*, vol. 56, no. 7, pp. 3122–3136, Jul. 2008.

[14] S. Ulp, Y. Le Moullec, and M. M. Alam, "LMS-based leader selection for distributed estimation," in *Proc. IEEE Int. Symp. Signal Process. Inf. Technol. (ISSPIT)*, Dec. 2017, pp. 211–215.

[15] P. Heinrich, H. Bergler, and D. Eilers, "Energy consumption estimation of software components based on program flowcharts," in *Proc. IEEE Int. Conf. High Perform. Comput. Commun., IEEE 6th Int. Symp. Cyberspace Saf. Secur., IEEE 11th Int. Conf. Embedded Softw. Syst. (HPCC,CSS,ICESS)*, Aug. 2014, pp. 542–545.

[16] V. Konstantakos, A. Chatzigeorgiou, S. Nikolaidis, and T. Laopoulos, "Energy consumption estimation in embedded systems," *IEEE Trans. Instrum. Meas.*, vol. 57, no. 4, pp. 797–804, Apr. 2008.

[17] V. Tiwari, S. Malik, and A. Wolfe, "Power analysis of embedded software: A first step towards software power minimization," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 2, no. 4, pp. 437–445, Dec. 1994.

[18] W. Hu and W. P. Tay, "Multi-hop diffusion LMS for energy-constrained distributed estimation," *IEEE Trans. Signal Process.*, vol. 63, no. 15, pp. 4022–4036, Aug. 2015.

[19] M. O. Sayin and S. S. Kozat, "Compressive diffusion strategies over distributed networks for reduced communication load," *IEEE Trans. Signal Process.*, vol. 62, no. 20, pp. 5308–5323, Oct. 2014.

[20] J. Fernandez-Bes, R. Arroyo-Valles, J. Arenas-García, and J. Cid-Sueiro, "Censoring diffusion for harvesting WSNs," in *Proc. IEEE 6th Int. Workshop Comput. Adv. Multi-Sensor Adapt. Process. (CAMSAP)*, Dec. 2015, pp. 237–240.

[21] Y. Wang, W. P. Tay, and W. Hu, "An energy-efficient diffusion strategy over adaptive networks," in *Proc. 10th Int. Conf. Inf., Commun. Signal Process. (ICICS)*, Dec. 2015, pp. 1–5.

[22] C. G. Lopes and A. H. Sayed, "Diffusion adaptive networks with changing topologies," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar./Apr. 2008, pp. 3285–3288.

[23] C. G. Lopes and A. H. Sayed, "Steady-state performance of adaptive diffusion least-mean squares," in *Proc. IEEE/SP 14th Workshop Stat. Signal Process.*, Aug. 2007, pp. 136–140.

[24] F. S. Cattivelli and A. H. Sayed, "Diffusion LMS strategies for distributed estimation," *IEEE Trans. Signal Process.*, vol. 58, no. 3, pp. 1035–1048, Mar. 2010.

[25] X. Zhao and A. H. Sayed, "Performance limits for distributed estimation over LMS adaptive networks," *IEEE Trans. Signal Process.*, vol. 60, no. 10, pp. 5107–5124, Oct. 2012.

[26] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *Proc. 4th Int. Symp. Inf. Process. Sensor Netw.*, Apr. 2005, pp. 63–70.

[27] S.-Y. Tu and A. H. Sayed, "Optimal combination rules for adaptation and learning over networks," in *Proc. IEEE 4th Int. Workshop Comput. Adv. Multi-Sensor Adapt. Process. (CAMSAP)*, Dec. 2011, pp. 317–320.

[28] N. Takahashi, I. Yamada, and A. H. Sayed, "Diffusion least-mean squares with adaptive combiners: Formulation and performance analysis," *IEEE Trans. Signal Process.*, vol. 58, no. 9, pp. 4795–4810, Sep. 2010.

[29] A. Dunkels, F. Osterlind, N. Tsiftes, and Z. He, "Software-based online energy estimation for sensor nodes," in *Proc. ACM 4th Workshop Embedded Netw. Sensors*, 2007, pp. 28–32.

[30] W. Rukpakavong, L. Guan, and I. Phillips, "Dynamic node lifetime estimation for wireless sensor networks," *IEEE Sensors J.*, vol. 14, no. 5, pp. 1370–1379, May 2014.

[31] P. Ruberg, K. Lass, E. Liiv, and P. Ellervee, "Performance estimation of embedded applications on microcontrollers," in *Proc. IEEE Nordic Circuits Syst. Conf. (NORCAS), NORCHIP Int. Symp. Syst.-Chip (SoC)*, Oct. 2017, pp. 1–6.

[32] B. Selvig, "Measuring power consumption with CC2430 & Z-stack," Texas Instrum., Dallas, TX, USA, Appl. Note AN053, 2007, vol. 5.

[33] *MSP430x5xx MSP430x6xx Family User's Guide Rev. Q*, Texas Instrum., Dallas, TX, USA, Mar. 2018.

[34] S. Sonavane, B. Patil, and V. Kumar, "Experimentation for packet loss on MSP430 and nRF24L01 based wireless sensor network," *Int. J. Adv. Netw. Appl.*, vol. 8, no. 5, p. 25, 2017.

[35] S. Kurt, H. U. Yildiz, M. Yigit, B. Tavli, and V. C. Gungor, "Packet size optimization in wireless sensor networks for smart grid applications," *IEEE Trans. Ind. Electron.*, vol. 64, no. 3, pp. 2392–2401, Mar. 2017.

[36] J. Bito, R. Bahr, J. G. Hester, S. A. Nauroze, A. Georgiadis, and M. M. Tentzeris, "A novel solar and electromagnetic energy harvesting system with a 3-D printed package for energy efficient Internet-of-Things wireless sensors," *IEEE Trans. Microw. Theory Techn.*, vol. 65, no. 5, pp. 1831–1842, May 2017.

[37] *Ultra-Low-Power Multi-Protocol Bluetooth 5 Certified Rev. 1*, ON Semicond., Phoenix, AZ, USA, Jan. 2018.

[38] M. M. Alam, O. Berder, D. Menard, T. Anger, and O. Sentieys, "A hybrid model for accurate energy analysis of WSN nodes," *EURASIP J. Embedded Syst.*, vol. 2011, 2011, Art. no. 307079.

[39] *Efficient Multiplication Division Using MSP430*, Texas Instrum., Dallas, TX, USA, 2006.

SANDER ULP received the B.S. and M.Sc. degrees *(cum laude)* in telecommunication from the Tallinn University of Technology (TTÜ), Tallinn, in 2011 and 2013, respectively, where he is currently pursuing the Ph.D. degree. From 2013 to 2016, he was a Junior Researcher with the Department of Radio and Communication Engineering, TTÜ. Since 2017, he has been with the Thomas Johann Seebeck Department of Electronics, TTÜ. His research interests are in distributed estimation, learning and adaptation over networks, wireless sensor networks, digital signal processing, and cognitive radio.

YANNICK LE MOULLEC received the M.Sc. degree in electrical engineering (EE) from the Université de Rennes I, France, in 1999, and the Ph.D. degree in EE from the Université de Bretagne Sud, France, in 2003. From 2003 to 2013, he was a Post-Doctoral Professor, an Assistant Professor, and an Associate Professor with Aalborg University, Denmark. He then joined the Tallinn University of Technology, Estonia, where he was a Senior Researcher, from 2013 to 2016 ,and is currently a Professor. He has supervised nine Ph.D. theses and more than 50 M.Sc. theses; he is also supervising five M.Sc. theses and five Ph.D. theses. His research interests span HW/SW co-design, embedded systems, reconfigurable systems, and IoT. He is a Co-Principal Investigator for the H2020 COEL ERA-Chair Project.

MUHAMMAD MAHTAB ALAM received the M.Sc. degree in electrical engineering from Aalborg University, Denmark, in 2007, and the Ph.D. degree from the INRIA Research Center, University of Rennes 1, in 2013. He did his Post-Doctoral Research in the Qatar Foundation funded project Critical and Rescue Operations Using Wearable Wireless Sensor Networks from the Qatar Mobility Innovations Center, from 2014 to 2016. In 2016, he was elected as a European Research Area Chair holder in Cognitive Electronics Project and an Associate Professor with the Thomas Johann Seebeck Department of Electronics, Tallinn University of Technology. In 2018, he received a tenure professorship to a chair Telia professorship under the cooperation framework between Telia and the Tallinn University of Technology. He has authored and co-authored over 55 research publications. His research interests include self-organized and self-adaptive wireless sensor and body area networks specific to energy efficient communication protocols and accurate energy modeling, the Internet-of-things, public safety and critical networks, embedded systems, digital signal processing, and software defined radio. He is a Principal Investigator of NATO-SPS-G5482 Grant and the Estonian Research Council PUT-Team Grant.

CURRICULUM VITAE

Personal data

Name:	Sander Ulp
Date of birth:	May 25th 1989
Place of birth:	Tallinn, Estonia
Citizenship:	Estonian

Contact data

Address:	Rohula 4, 10912, Tallinn, Estonia
Phone:	+372 56986324
E-mail:	sander.ulp@taltech.ee

Education

2014 – 2019:	Tallinn University of Technology Electronics and Telecommunication	PhD studies
2011 – 2013:	Tallinn University of Technology Telecommunication Engineering	Master of Science (*cum laude*)
2008 – 2011:	Tallinn University of Technology Telecommunication Engineering	Bachelor of Science
2005 – 2008:	Tallinn Secondary Science School	Secondary Education

Language competence

Estonian #1:	Native speaker
English #2:	Fluent
Finnish #3:	Intermediate
Swedish #4:	Basic
Russian #5:	Basic
German #6:	Basic

Professional employment

2018 – :	Eliko Competence Centre	Research Scientist
2017 – :	Tallinn University of Technology Thomas Johann Seebeck Department of Electronics	Early Stage Researcher
2013 – 2016:	Tallinn University of Technology Department of Radio and Communications Engineering	Early Stage Researcher
2010 – 2014:	Estonian Public Broadcasting	Playout Engineer

Supervised theses

2016:	MATLAB Application for Digital Audio Processing Laboratory	Artjom Savitski Master's Degree
2016:	Laboratory Assignment for the Course of Modern Wireless Communications. LTE Air Interface Downlink	Jürgen Algma Master's Degree

ELULOOKIRJELDUS

Isikuandmed

Nimi:	Sander Ulp
Sünniaeg:	25.05.1989
Sünnikoht:	Tallinn, Eesti
Kodakondsus:	Eestlane

Kontaktandmed

Aadress:	Rohula 4, 10912 Tallinn, Eesti
Telefon:	+372 56986324
E-post:	sander.ulp@taltech.ee

Hariduskäik

2014 – 2019:	Tallinna Tehnikaülikool Elektroonika ja telekommunikatsioon	Doktorantuur
2011 – 2013:	Tallinna Tehnikaülikool Telekommunikatsioon	Magistrantuur (*cum laude*)
2008 – 2011:	Tallinna Tehnikaülikool Telekommunikatsioon	Bakalaureus
2005 – 2008:	Tallinn Reaalkool	Keskharidus

Keelteoskus

Eesti #1:	Emakeel
Inglise #2:	Suurepärane
Soome #3:	Keskmine
Rootsi #4:	Algtase
Vene #5:	Algtase
Saksa #6:	Algtase

Teenistuskäik

2018 – :	OÜ Eliko Tehnoloogia Arenduskeskus	Teadur
2017 – :	Tallinn tehnikaülikool Thomas Johann Seebecki elektroonikainstituut	Nooremteadur
2013 – 2016:	Tallinn tehnikaülikool Raadio-ja sidetehnika instituut	Nooremteadur
2010 – 2014:	Eesti Rahvusringhääling	Väljastusinsener

Juhendatud lõputööd

2016:	MATLABi rakendus digitaalse helitöötluse laboratoorses töös	Artjom Savitski Magistrikraad
2016:	Laboratoorne töö kursusele ”Kaasaegne traadita side”. LTE raadioliidese allalüli.	Jürgen Algma Magistrikraad