Summer 2016

Towards Aggregating Time-Discounted Information in Sensor Networks

Xianping Wang

Old Dominion University, xwang012@odu.edu

TOWARDS AGGREGATING TIME-DISCOUNTED INFORMATION IN SENSOR NETWORKS

by

Xianping Wang
B.E. July 1999, Beijing Technology and Business University , China
M.E. July 2002, The First Institute of China Aerospace Science and Technology Corporation, China

A Dissertation Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

COMPUTER SCIENCE

OLD DOMINION UNIVERSITY
August, 2016

Approved by:

Stephan Olariu (Director)

Michele C. Weigle (Member)

Hussein Abdel-Wahab (Member)

Irwin Levinstein (Member)

Dimitrie C. Popescu (Member)

ABSTRACT

TOWARDS AGGREGATING TIME-DISCOUNTED INFORMATION IN SENSOR NETWORKS

Xianping Wang
Old Dominion University, 2016
Director: Dr. Stephan Olariu

Sensor networks are deployed to monitor a seemingly endless list of events in a multitude of application domains. Through data collection and aggregation enhanced with data mining and machine learning techniques, many static and dynamic patterns can be found by sensor networks. The aggregation problem is complicated by the fact that the perceived value of the data collected by the sensors is affected by many factors such as time, location and user valuation. In addition, the value of information deteriorates often dramatically over time. Through our research, we already achieved some results:

- A formal algebraic analysis of information discounting, especially affected by time. A general model and two specific models are developed for information discounting. The two specific models formalize exponetial time-discount and linear time-discount.

- An algebraic analysis of aggregation of values that decay with time exponentially. Three types of aggregators that offset discounting effects are formalized and analyzed. A natural synthesis of these three aggregators is discovered and modeled.

- We apply our theoretical models to emergency response with thresholding and confirm with extensive simulation.

- For long-term monitoring tasks, we laid out a theoretical foundation for discovering an emergency through generations of sensors, analysed the achievability of a long-term task and found an optimum way to distribute sensors in a monitored area to maximize the achievability.

- We proposed an implementation for our alert system with state-of-art wireless microcontrollers, sensors, real-time operating systems and embedded internet protocols.

By allowing aggregation of time-discounted information to proceed in an arbitrary, not necessarily pairwise manner, our results are also applicable to other similar homeland

security and military application domains where there is a strong need to model not only timely aggregation of data collected by individual sensors, but also the dynamics of this aggregation.

Our research can be applied to many real-world scenarios. A typical scenario is monitoring wildfire in the forest: A batch of first-generation sensors are deployed by UAVs to monitor a forest for possible wildfire. They monitor various weather quantities and recognize the area with the highest possibility of producing a fire — the so-called area of interest (AoI). Since the environment changes dynamically, so after a certain time, the sensors re-identify the AoI. The value of the knowledge they learned about the previous AoI decays with time quickly, our methods of aggregation of time-discounted information can be applied to get update knowledge. Close to depletion of their energy of the current generation of sensors, a new generation of sensors are deployed and inherit the knowledge from the current generation. Through this way, monitoring long-term tasks becomes feasible.

At the end of this thesis, we propose some extensions and directions from our current research:

- Generalize and extend the special classes of Type 1 and Type 2 aggregation operators;
- Analyze aggregation operator of Type 3 and Type 4, find some special applicable candidates;
- Data aggregation across consecutive generations of sensors in order to learn about events with discounting that take a long time to manifest themselves;
- Network implications of various aggregation strategies;
- Algorithms for implementation of some special classes of aggregators.
- Implement wireless sensor network that can autonomously learn and recognize patterns of emergencies, predict incidents and trigger alarms through machine learning.

ACKNOWLEDGMENTS

First of all, I would like to thank Dr. Stephan Olariu, my advisor, for his farsighted guidance, constructive advice and valuable comments on completing this dissertation. His consistent patience and help in my PhD life lead me to this accomplishment.

Next, I would like to thank my PhD committee Drs. Michele C. Weigle. She reviewed this dissertation carefully and thoroughly, gave constructive comments and suggestions, which leads to a completion of this work.

Finally, I would like to thank my PhD committee Dr. Hussein Abdel-Wahab, Dr. Irwin Levinstein and Dr. Dimitrie C. Popescu. Their broad expertise, careful reviewing, and valuable suggestions have enhanced the dissertation.

TABLE OF CONTENTS

Page

Chapter

LIST OF TABLES

LIST OF FIGURES

List of Algorithms

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

Every year, various disasters occurred all over the world. During 2015, the top six natural disasters [7] were:

- Earthquake in Nepal
- Flood in Chennai, India
- Heat wave in southern India
- Typhoon and monsoon rains in Myanmar, Bangladesh, and India
- Flood in Malawi and Mozambique
- Drought in Ethiopia.

In order to respond to such disasters efficiently and in a timely manner, the United Nations set up the GDACS (Global Disaster Alert and Coordination System) [3] framework. GDACS is a cooperation framework between the United Nations, the European Commission and disaster managers worldwide to improve alerts, information exchange and coordination in the first phase after major sudden-onset disasters.

In United States, there are many generations of national alert systems shown in Table 1.

In 2012, United States launched the Wireless Emergency Alerts [6] (WEA), formerly known as the Commercial Mobile Alert System (CMAS), and prior to that as the Personal Localized Alerting Network (PLAN). WEA is an alerting network designed to disseminate emergency alerts to mobile devices especially smart phones. It supports three types of alerts:

- *Presidential alerts:* Alerts issued by the President of the United States.

TABLE 1: The Evolution of Emergency Broadcasting Systems in United States

Name	Period	Description
CONELRAD	1951–1963	The CONtrol of Electromagnetic RADiation (CONELRAD) was established in August 1951, originally called the ”Key Station System”. Participating stations turned to 640 and 1240 kHz AM and initiated a special sequence and procedure designed to warn citizens.
EBS	1963–1997	EBS was initiated to address the nation through audible alerts. It did not allow for targeted messaging. System upgraded in 1976 to provide for better and more accurate handling of alert receptions. It was originally designed to provide the President with an expeditious method of communicating with American Public, then expanded for use during peacetime at state and local levels.
EAS	1997–2006	EAS was jointly coordinated by the FCC, FEMA and NWS. It was designed for the President to speak to American people within 10 minutes. EAS messages composed of 4 parts: digitally encoded header, attention signal, audio announcement, digitally encoded end-of-message marker.
IPAWS	2006–	IPAWS modernizes and integrates the nation’s alert and warning infrastructure, integrates new and existing public alert and warning systems and technologies, provides authorities a broader range of message options and multiple communications pathways, increases capability to alert and warn communities of all hazards impacting public safety.

- *Imminent alerts:* Alerts involving imminent threats to safety of life, issued in two different categories: extreme threats and severe threats.

- *AMBER alerts:* child abduction alert.

All the alert systems above work on a large scale, i.e. low event resolution and low spacial resolution. Some researchers even argued that their effects are not as good as expected [1]. Worse yet, they do not have functions for pre-emergency alert, only broadcast when those emergencies happened. They can not be re-tasked for emergencies other than those presets.

As a complement and enhancement to the systems above, M. Weigle and S. Olariu *et al.* [78] proposed and carried out a NSF-funded[1] project, ALERT (An Architecture for the

[1] Work supported by NSF grant CNS-1116238

Emergency Re-tasking of Wireless Sensor Networks). The novelty of ALERT lies in the theoretical foundation of re-tasking independently-deployed sensor networks, leading to a fundamental understanding of the design principles of capability reallocation and sharing to best satisfy the needs of emergency applications. Both re-tasking and integration of sensor capabilities will be transparent to the emergency applications. The resource-constrained nature of sensors, the wireless communication medium, and the failure-prone networking environment, combined with the dynamic QoS requirements of the emergency applications pose formidable challenges for the design of ALERT.

The understanding acquired from developing ALERT will promote a wider adoption of sensor network systems in support of guarding our national infrastructure and public safety in finer resolution. This project will result in significant scientific and technological advances that will provide invaluable help with disaster management and search-and-rescue operations. ALERT will have a broad societal impact as sensor networks are being integrated into the fabric of the society.

The research in this dissertation is a continuation of ALERT; it focuses on the following scenarios:

- *Highly dynamic emergencies:* Most emergencies occur suddenly and develop very fast, i.e. the value of information gathered by the wireless sensor network decays rapidly;

- *Unexpected emergencies:* Many emergencies occur and develop in an unknown pattern, a WSN can find and identify these unknown patterns through unsupervised learning, supervised learning and reinforcement learning;

- *Pre-disaster patterns:* Through pre-disaster monitoring, action can be taken to prevent disasters from occurring. This is a very important function of a monitoring system.

- *Long-term tasks monitoring:* Many emergencies occur through long-term accumulation of developments, such as earthquakes, volcano eruptions, global warming, etc.

1.2 MOTIVATING SCENARIOS

For highly dynamic emergencies, consider a sensor network deployed with the specific mission of protecting a power plant. The information about a possible attack on the power plant will be more valuable the less it is delayed. It will lose value continuously as there will not be adequate time to prepare. Thus, there are powerful incentives for reporting an attack as soon as possible. However, the cost of a false alarm is considered to be prohibitive in terms of the amount of human attention it requires. Thus, there are powerful incentives for aggregating individual actor information before reporting.

To begin, imagine that the power plant is threatened by an intruder who intends to sabotage the turbines. Again, the less delay in the information about the intrusion event, the better. The sensors that have detected the event need to decide whether to report an intrusion (and risk triggering a false alarm) or wait until several other sensors have corroborated the intrusion. With each moment of delay in notification of the intrusion, the ability to find the intruder decreases, as the intruder may be moving, and the area to search increased quadratically in the time since detection.

Next, imagine a foreign hacker who launches an attack on the network equipment controlling the power plant. The earlier the cyber-attack is detected, the higher the chance of thwarting the intruder. But as time goes on, the worse the attack gets. This type of cyber-attack may well double or triple the malicious network traffic with each time increment. Thus, the value of the information to the decision maker will deteriorate rapidly, as it becomes harder and harder to fight the attack as the network becomes overwhelmed.

The common characteristic of all the above scenarios is that getting information quickly has value. On the other hand, there are costs associated with obtaining information.

For long-term phenomenons, such as climate change in a region, global warming, continental drift, etc. they develop years by years. However, many sensors can only last several weeks, months. i.e. the density of the WSN will decrease gradually to an unusable state. Before this unusable state, new generations of sensors are deployed to replenish the WSN. In order to maintain continuity, they will learn knowledges from their previous generation. So monitoring those long-term phenomena needs generations of sensors. The problem of how to aggregate and pass information from generation to generation arises naturally.

1.3 RELATED WORK

Information is a good that has value and hence can be traded [41]. Assessing the value of information has been a topic of research in economics [12, 31, 63, 85] but has only received superficial attention in sensor networks [38, 86]. It has long been recognized that, being dependent on subjective human valuation, the value of information is hard to assess. To complicate matters, it is often the case that the value of information is subject to rapid deterioration over space and time [9, 77], which is a phenomenon of locality of space and time.

It is widely accepted that aggregating, in a suitable fashion, data collected by the various sensors that have witnessed an event is important, but measuring the value of the aggregated information remains challenging [82]. There is an obvious reason for this state of affairs: aggregation is a complex concept whose semantics are context-dependent [66]. It is important to consider formal ways of defining the aggregation of discounted information so that aggregation might be better described and understood. In turn, this may lead to better designs for emergency response systems where, in many cases, the failure to aggregate data in a timely fashion may have catastrophic consequences [75, 77].

While it is commonly accepted that the value of information often increases as a result of aggregation, almost nothing is known about the dynamics of aggregation and about the value of the resulting information, especially in the presence of discounting.

To our best knowledge, data aggregation across consecutive generations of sensor networks deployed for long term mission has not been looked at before. Because of limited resources, especially energy, the life of sensors is much shorter than the required time span of long term missions such as monitoring the climate of a region, detecting the impact pattern of meteorites on the Mars, etc. In such cases, we need many generations of sensors. Aggregating information from generation to generation to get a long term continuity of the mission remains challenging. There is a need not only for theoretical analysis but also for implementation algorithms.

1.4 RESEARCH APPROACHES

From the previous section, we see there are so many ambiguities, uncertainty and complexity in evaluation of the value of information, the process of information discounting, aggregation dynamics of data, and the interaction between them. So, quantitative and formal mathematical models are required.

First, a formal and quantitative description of evaluating the value of information which is a map from data to a value space which is mostly subjective will be constructed. We will find the factors that are reflected in such value space such as utility, probability and confidence, etc and give an explanation on their meaning.

Second, the discounting process will be described formally to reveal its essence mostly by algebra, the factors discount the value of information, the effect and the way of these factors on the underlying value of information. We will set up some families of functions to expound those factors, some families of operators to explain the influencing way such as discount by ratio or by absolute value. The final trend of value deterioration will also be solved.

Third, a formal algebraic description of the value of the aggregated information will be offered. This description will give us some feel for the result of aggregation especially whether there are transient or permanent patterns hidden in the data, the dynamics of aggregation such as steps taken in the process and grouping of values, even the whole system state transitions process.

Fourth, interaction between the discounting process and the aggregation process exerted on the value of information being processing will be modeled formally. Cases rendered from the permutation of the order of discounting effect and aggregating effect will also be modeled.

Fifth, the information discounting and aggregation from generation to generation of long term deployed large scale sensor network will be described formally. Usually, in a long term mission, there are several stages of different sub-tasks. Sensor retasking is one of our approaches to handle this challenge.

Sixth, we will develop decision strategies for the aggregation of the data collected by sets of actors, whether sensors, robots, or people and apply these strategies to fields in which hybrid networks of humans and sensors need to be formed, such as sensor networks deployed in support of emergency response and tactical applications.

Seventh, we modeled the lifespan of a single sensor, a batch of sensors, a generation of sensors and multiple generations of sensors in stochastic modeling. Through machine learning algorithms, the sensors learn from continuously sensed data and pass the learned knowledge to the following generation. With a limited number of sensors, we find the optimum way to allocate sensors to each generation and each monitored area to maximize the achievability of a long-term task.

Finally, we will develop a series of related algorithms and simulation models to implement, verify and confirm our theoretical models.

1.5 CONTRIBUTIONS

Through research, with all the theoretical models and algorithms, we give the following contributions.

Formal model to evaluate the value of information: This model will provide us a formal way to evaluate the value of information from the data collected in sensor networks. If information from different areas has the same value space, as in economy, everything can be evaluated by money, then their value can be assessed and compared to determine their merits. This can be applied to information collecting where we only collect those which is valuable to us. For information storage, people have a criterion based on the value of information to make an economical and efficient sifting on information.

Formal model of information discounting: This model will give people a whole picture of the discounting process of value of information exerted by the effect of relative discounting factors such as time, location, applications and users, etc. With the knowledge of discounting factors people may find ways to eliminate the factors in order to mitigate their effect; with the knowledge of various discounting rates on different types of information, people will have a selecting criterion to choose the information that deteriorates slowly and come up with ideas to handle that which rapidly deteriorates.

Formal model of aggregation: This model will equip us with a rig to counter the effect of discounting. From many data collected by sensor networks, some hidden static and dynamic patterns may be revealed by aggregation formally. Usually, a larger aggregated value can be expected from a set of decayed data. This will also implicate us some inspirations on natural aggregation strategies such as thresholding.

Formal model of the interaction between discounting and aggregation: With the knowledge of interaction detail between discounting and aggregation, people may find better aggregation strategies to offset different discounting. By setting thresholds on the value, the time for certain aggregation can be calculated, and with this time, relative operations can be scheduled to achieve some final targets. In emergency situation, this will become a vital factor to save lives.

Formal model of aggregation through consecutive generations of sensors: With this model, we can have a series of aggregated value across generations and achieve some long term targets finally through generational learning. Many real applications are long-term missions such as monitoring earthquake along the edges of tectonic plates, pollution in the air, water, and soil at an area, etc. Each generation gradually improves its monitoring efficiency, predicting accuracy, responding sensitivity by learning accumulated knowledge from its previous generations, which is exactly a reinforcement learning process.

Formal model of the achievability of long-term tasks: This model formalizes the continuity of a single patch, a single cluster that consists of many patches, a generation of all clusters and the whole lineage of the wireless sensor network that deployed over a monitored area with many clusters. For limited resource of sensors, we found the optimum way to allocate sensors to its generations, clusters and patches to maximize the achievability of a long-term task.

Algorithms and simulations of aggregation strategies: Aggregation strategies are implemented with algorithms and verified with simulation, which will implicate the implementation and application of these aggregation strategies in real sensor networks that deployed for various targets. They will provide reference and baseline for comparison to real platforms.

Implementation through state-of-art wireless micro-controllers: We proposed an implementation for our alert system with cutting-edge wireless micro-controllers, narrow-band wireless transceivers, transmission range extenders and sensors. By equipping the micro-controller with real-time embedded operating system, algorithms for discounting, aggregation, pattern recognition and reinforcement learning, an efficient and autonomous alert system can be constructed.

The following is a list of publications related to the work reported in this dissertation.

- Stephan Olariu, Shahram Mohrehkesh, **Xianping Wang** and Michele C. Weigle, ”On Aggregating Information in Actor Networks,” SIGMOBILE Mobile Computing Communications Review, Vol. 18, No. 1, January 2014, pp. 85-96.

- **Xianping Wang**, Aaron Walden, Michele C. Weigle and Stephan Olariu, ”Strategies for Sensor Data Aggregation in Support of Emergency Response,” In Proceedings of the Military Communications Conference (MILCOM). Baltimore, MD, October 2014.

- Shahram Mohrehkesh, Aaron Walden, **Xianping Wang**, Michele C. Weigle and Stephan Olariu, ”Towards Building Asset Registry in Emergency Response,” In

Proc. of the 3rd IEEE Annual International Workshop on Mission-Oriented Wireless Sensor Networking (MiSeNet). Philadelphia, PA, October 2014.

- Samy El-Tawab, **Xianping Wang**, Ahmed Alhafdhi and Stephan Olariu, ”On Probabilistic Data Collection in the NOTICE Architecture”, In Proc. of the 3rd IEEE Annual International Workshop on Mission-Oriented Wireless Sensor Networking (MiSeNet). Philadelphia, PA, October 2014.

- *Poster:* **Xianping Wang**, and Stephan Olariu: ”Strategies for Sensor Data Aggregation for Emergency Response” in The College of William and Mary's 13th Annual Graduate Research Symposium, Williamsburg, Virginia, March 2014.

1.6 DISSERTATION ORGANIZATION

This dissertation is organized as follows:

In Chapter 1 Section 1.1, we begin with the context of current deployed emergency alert systems, then account three representative examples as motivating scenarios in Section 1.2, we find some problems and challenges in Section 1.3, then we proposed solutions for these problems and challenges in Section 1.4 and summarized our contributions in Section 1.5.

We give a description of the recent related background and previous works in Chapter 2.

In Chapter 3, we give a set of formal algebraic descriptions of information discounting. We started the algebra from general information discounting in Section 3.1, then focused on time-discounted information in Section 3.2, and find two interesting families of time-discounting functions. We summarized this chapter in a general function view of information discounting in Section 3.3.

In Chapter 4 – Algebra of Data Aggregation, we discuss general data aggregation in Section 4.1, instantaneous aggregation in Section 4.2, anti-discount aggregation in Section 4.3 and aggregating exponentially-discounted value of aggregated information in Section 4.10. By defining a set of definitions for aggregation operators in Section 4.4, we continue this chapter by studying the first two types of aggregators and their specially-interesting candidates as well as the last two types of aggregators by expounding their properties in detail. Lastly, a specially-interesting candidate for the last type is formalized algebraically and the first two special candidates are generalized to all three basic types of aggregators.

In Chapter 5, the application of two types of aggregation operators are demonstrated and verified with simulation with Type 1 in Section 5.1 and Type 2 in Section 5.2.

In Chapter 6, data aggregation across generations of sensors is studied. Beginning with a whole view of working flow from generation to generation in wireless sensor networks for monitoring and alert, then summarizing algorithms for data preprocessing, cluster partitioning, supervised-learning and reinforcement learning, we come up with the algorithm for area-of-interesting identification. With models for the continuity of a single patch, a single cluster consisting of several patches, a single generation and a whole lineage, we find the optimum strategy to maximize the achievability of a long-term task.

An implementation with state-of-art devices and software is proposed in Chapter 7.

We summarize our current research in Chapter 8 and extend it to future work. With some future tasks in mind, we outline a blueprint of future research as well.

CHAPTER 2

STATE OF THE ART

Information is more valuable new than old. For example, real time stock quotes are more valuable than quotes which are delayed 20 min. We see evidence of this in the tiered pricing offered by exchanges for pricing information based on the delay: the less delay, the more expensive the service. In everyday situations, we recognize that today's newspaper is more important than yesterday's. This deterioration of information value is discussed in a general way in the literature of the economics of information [12, 68, 69], which is a reflection of the principle of the time value of money describing the greater benefit of receiving money now rather than later.

The principle of the time value of money describes the valuation of a likely stream of income in such a way that annual incomes are discounted and then added together in the future, hence providing an accrual sum 'present value' of the entire income stream. The standard calculation for the time value of money derives from the most basic algebraic expression for the present value of a future sum that is discounted to the present by an amount equal to the time value of money. Suppose the future value sum V_F to be received in one year is discounted at the rate of interest r to give the present value sum V_P, then

$$V_P = \frac{V_F}{1+r} \tag{2.1}$$

In other words, the same amount of money values less in the future $t = 1 year$ than at present $t = 0$ since $1 > r > 0$. Rewrite the notation in Equation (2.1), let $V(0) = V_P$ and $V(1) = V_F$, then Equation (2.1) turns into

$$V(0) = \frac{V(1)}{1+r} \tag{2.2}$$

Equation (2.2) can be rewritten into

$$V(1) - V(0) = rV(0)$$

The above equation describes that after one year, an amount of $rV(0)$ is discounted from the current value $V(0)$, then the discounted value $V_d(1)$ of $V(0)$ after one year is

$$V_d(1) = V(0) - rV(0) = (1-r)V(0). \tag{2.3}$$

Suppose the interest rate is the same for each year even though it is not true in the real-world economy, then from Equation (2.3) we have

$$\begin{aligned} V_d(1) &= (1-r)V(0) \\ V_d(2) &= (1-r)V_d(1) \\ &\cdots \\ V_d(n) &= (1-r)V_d(n-1). \end{aligned}$$

From the system of equations above, we can deduce the following equation

$$V_d(n) = (1-r)^n V(0). \tag{2.4}$$

The equation above can be proved by mathematical induction.

Equation (2.4) describes the time-discounting of value in a discrete mode. This inspires us to extend discrete time-discount into continuous time-discount and apply it in the area of wireless sensor network to describe the time-discounting procedure of the value of information collected by sensors. We also discount past information in favor of present information.

Discount functions have also been discussed by Frederick *et al.* in relationship to the psychology of individual decision making [31]. Based on discounted utility (DU), Frederick discussed spectacular disagreement among dozens of studies that all purport to be measuring time preference. The lack of agreement likely reflects the fact that the various elicitation procedures used to measure time preference consistently fail to isolate time preference from valuation of information. In our thesis, we focus on the time-discounting procedures of events themselves instead of personal time preference. Meanwhile, there is little empirical support to the DU model, even though it is widely used by economists. The discount utility is computed as the present discounted value of a future utility. For people with time preference for sooner rather than later gratification, it is less than the future utility. The utility of an event x occurring at future time t under utility function u, discounted back to the present (time 0) using discount factor δ, is

$$\alpha^t u(x_t).$$

Where $\alpha \in (0,1)$, since more distant events are less liked. The interpretation of α is not straightforward. Sometimes it is explained as the degree of a person's patience. Different people have different rates of time preference. While the DU model assumes that intertemporal preference can be characterized by a single discount rate, the large empirical literature devoted to measuring discount rates has failed to establish any stable estimate. There is extraordinary variation across studies, and sometimes even within studies. This failure is partly due to variations in the degree to which the studies take account of factors that confound the computation of discount rates (e.g., uncertainty about the delivery of future outcomes or nonlinearity in the utility function). In our thesis, we find a unary discount rate by stochastically analyzing the average life-span of events and by excluding the interference of personal psychological preferences and motives.

As an economic model, discount function is used to describe the weights exerted on rewards received at different time. Assume time is discrete and utility is time-separable, with $f(t)$ defined as the discount function, $c(t)$ as the consumption at time t and $u(\cdot)$ as the instantaneous utility function, then the total utility is given by

$$U\left(\{c_t\}_{t=0}^{\infty}\right) = \sum_{t=0}^{\infty} f(t)u(c_t).$$

Total utility in the continuous-time case is given by

$$U\left(c(t)_{t=0}^{\infty}\right) = \int_{t=0}^{\infty} f(t)u(c(t))$$

provided that this integral exists. Exponential discount function and hyperbolic discount function are two most commonly used discount functions.

Exponential discounting is used to analyze the choice over time (with or without uncertainty) in economics. Formally, exponential discounting occurs when total utility is given by

$$U\left(\{c_t\}_{t=t_1}^{t_2}\right) = \sum_{t=t_1}^{t_2} \delta^{t-t_1} u(c_t),$$

where δ is the exponential discount factor. In continuous time, exponential discounting is given by

$$U\left(c(t)_{t=t_1}^{t_2}\right) = \int_{t=t_1}^{t_2} e^{-\mu(t-t_1)}u(c(t)).$$

Exponential discounting implies that the marginal rate of substitution between consumption at any pair of time points depends only on the difference of those two points. Exponential discounting is dynamically time-consistent. For its simplicity, the exponential discounting assumption is the most commonly used in economics. However, alternatives like hyperbolic discounting have more empirical support, which is a time-inconsistent model of discounting. A large number of studies have since demonstrated that the constant discount rate assumed in exponential discounting is systematically being violated [31]. Hyperbolic discounting is a particular mathematical model devised as an improvement over exponential discounting, in the sense that it better fits the experimental data about actual behavior. However, the time inconsistency of this behavior has some quite perverse consequences. Hyperbolic discounting has been observed in both human and non-human animals. In hyperbolic discounting, valuations decay very rapidly for small delay periods, but then decay slowly for longer delay periods. This contrasts with exponential discounting, in which valuation decays by a constant factor per unit delay, regardless of the total length of the delay.

The formal mathematical model of hyperbolic discounting is described as

$$f_H(d) = \frac{1}{1+kd},$$

where $f_H(d)$ is the discount factor that multiplies the value of the reward, d is the delay in the reward, and k is a parameter governing the degree of discounting. This is compared with the formula for exponential discounting:

$$f_E(d) = e^{-kd}$$

The standard experiment used to reveal a test subject's hyperbolic discounting curve is to compare short-term preferences with long-term preferences.

Naturally, we want to find corresponding strategies to aggregate the value of time-discounting information.

Research shows that some forms of human memory decay exponentially [62]. In psychology, information aggregation [10, 11] studies how humans handle data. New information flows into a human mind and is merged into already processed information. Information integration theory, proposed by Norman H. Anderson, is used to model how a person integrates information from a number of sources to make an overall conclusion. The theory proposes three functions:

- **The valuation function** $v(S)$ is an empirically derived mapping from information S to a value v.
- **The integration function** $r = I(v_1, v_2, .., v_n)$ is an algebraic function combining the subjective values of the information. 'Cognitive algebra' refers to the class of functions that are used to model the integration or aggregation process. Typical operations are adding, averaging, weighted averaging, multiplying, etc.
- **The response production function** $R = M(r)$ is the process by which the internal mental impression is translated into an overt response.

However, for the integration function, Anderson did not take time-discount into account. Moreover, there is no consistent principle except algebraic models for information integration theory. Our ability to perceptually aggregate information is an indicator of our overall thinking abilities [53].

At the social level, economists discuss the half-life of an academic field or even of a scientific paper. Recently, the Web has provided us empirical evidence for information decay in academia. Researchers in physics post their articles online in standard venues: the citation rates decay exponentially starting from the time of posting [16].

However, there is no uniform theoretic foundation for these discount functions discussed above. They are based on experimental assumptions. In this thesis, we laid out a common theoretic foundation for discount functions and their interaction with aggregation.

Deutsch [27] has modeled aggregation in the context of politics, investigating the way coalitions form and dissolve. This kind of merger when states form coalitions is analogous, at a higher scale, to the aggregation we describe between nodes in a sensor network. Work on preferential attachment [13] has suggested that a tendency to attach to the node in a network that already has more connections will create realistic networks. Work in this area has focused on the purely formal properties of the network: once a vertex has many edges, more vertices will attach to it. In our model, once a vertex has established value, many other vertices will attempt to attach in order to share the value. But our most sought-after nodes will sometimes refuse these connections.

Lawrence and Lorsch [57] proposed a model of aggregation in which individuals ambassadors carry information back and forth between culturally distinct parts of the corporation. Porter's [84] ideas on how organizational units specialize and, as a result, create boundaries underlying the justifications offered for most aggregation technologies [91].

In work on coordination science [25, 64, 65], patterns of behaviors are classified with respect to producers and consumers of resources. Aggregation, then, can be seen as a linking between places. We will use these aggregation ideas that aggregation occurs in shared places and that it is facilitated by ambassadors.

In a series of papers, Jones *et al.* [44–48] have investigated a particular instance of information aggregation in sensor networks, namely that of reaching a consensus. They propose a new way of looking at large-scale sensor networks, motivated by lessons learned from the way biological ecosystems are organized. They believe that techniques used in small-scale sensor networks are not likely to scale to large networks; that such large-scale networks must be viewed as an ecosystem in which the sensors/effectors are organisms whose autonomous actions, based on local information, combine in a communal way to produce global results. As an example of a useful function, they also demonstrate that fully distributed consensus can be attained in a scalable fashion in massively deployed sensor networks where individual motes operate based on local information, making local decisions that are aggregated across the network to achieve globally-meaningful effects. While their model of a sensor network is similar to ours, their approach is not explicitly probabilistic.

In their previous work, Olariu *et al.* [73] have focused on latency as a key aspect of modern communication networks, and have investigated its implications in the context of sensor networks [77,81]. In [74,75] they have proposed metrics for measuring aggregation. Recently, they introduced the idea of couriers moving entities which visit sensors and aggregate the information between sensors or sensor clusters [76].

In [75] Nickerson *et al.* proposed a model of aggregation and introduced a discount function (e.g. [31]) effectively reducing the value of a network when latency delays communication.

Olariu *et al.* realize that there are some limitations in their research [82]:

They assumed that information deteriorates at a constant rate throughout the network. In a sensor network, they think this assumption is fair: the sensors have a common goal, and the information is all related to a single goal. In social networks, things may be more complex. For example, the freshness of information may be more valuable to a company on the verge of bankruptcy than to a company flush with money. There is an additional complication: a company on the verge of bankruptcy may want to disguise the urgency of its information needs, and may also want to inflate the value of the information they hold in the hope of merging prior to collapsing. Thus, in adversarial situations, there is uncertainty relating to the decay value of information to the participants. Also, participants may inflate the value of their information in the hopes of encouraging aggregation.

Such assumptions about the uncertainty of information functions and value will lead to more complex models. But such additional assumptions do not negate the primary idea explored here: both sides have different decision thresholds, with the less valuable player more likely to want to aggregate.

They assumed a simple physical model of aggregation based on physical distance. In reality, the model is complex. Some communication is instant, and some is delayed through electronic queues, and others still through transportation queues. These different assumptions in heterogeneous networks will yield more complex models.

They also have assumed a local decision model. This is realistic in the case of sensors. But in organizations, it may be possible to also reason at a global level. We can ask: how much worse is local decision making from global decision making? Another way of putting this: how far from optimal is a set of simple pairwise decisions? Our pairwise decisions are a kind of parallel greedy algorithm. We know that sometimes greedy algorithms can fail, but often they are good approximations.

They also assumed that information from the sensors is independent. This is generally true in geographic sensor networks. Sometimes there is overlap. Particularly in social networks, there are bound to be dependencies. In such cases, we cannot simply add values, but must determine the overlap and subtract that overlap from the sum. This may lead to a faster diminishment of returns from aggregating: information may become progressively less novel to the higher-value nodes.

The size of data to aggregate will effect the time to aggregate. In fully electronic networks, the throughput of a link and the message size together affect overall latency (see [73] for a discussion on this topic related to mobile computing). Thus, in a more complex model, the size of the message and the electronic throughput may be important to include.

In their decision model shown in [82], they looked at the aggregation of a particular event an intrusion across a set of sensors. They did not model the continuous monitoring of a stream of events. In such cases, the decay of information becomes important: it will not make sense to aggregate old, lower value information in comparison to newer, higher-value information. When there is an event stream, it becomes possible for the sensors to learn something about the environment, and about their fellow sensors' behaviors. Therefore, over time, sensors might make more effective decisions about who to aggregate with, based on past experiences, using reinforcement learning techniques [49].

In [82] Olariu *et al.* have provided an incentive-based probabilistic model for pairwise aggregation and have shown that the model finds applications to sensor networks, social networks and business applications such as merging between companies and departments. At the same time, they realize that there are a number of limitations in the current work.

First, it does not show what extent their incentive-based local aggregation has an optimal global effect after many pairs have had the chance to aggregate. It also does not discuss the interaction between energy conservation and regions of a sensor network gaining local information before a global aggregation takes place.

Second, their model is similar to the preferential attachment model and predict that, at some point, nodes will start refusing aggregation from lower value nodes. However, there's no comparison between the two.

Third, in their model they have assumed that information decays uniformly across the networks and the set of actors. However, it is unclear whether the decision process would change significantly if the value decay rate is non-uniform. In addition, the introduction of a stream of events may change the dynamics of the network by encouraging the processing of new information at the expense of old information.

The questions discussed above lead to the work in this thesis: we lay out a uniform theoretical foundation for the time-discounting process of the value of information, the interaction process between aggregation and time-discount, generational learning and the achievability of long-term tasks.

CHAPTER 3

ALGEBRA OF INFORMATION DISCOUNTING

3.1 GENERAL INFORMATION DISCOUNTING

In reality, phenomena sometimes occur in continuous time, sometimes occur in discrete time. For simplicity, we assume the phenomena we discuss occur in continuous time. Consider an arbitrary sensor and let V be the random variable that denotes the “amplitude” of some sensed attribute. When it is sampled or sensed, it will have an initial number value from a domain Ω reflects the corresponding physical quantity which is objective. While assessing the value or worth of this number is affected by many factors such as time, location and user, etc. The worth is from a subjective domain such as $[0,1]$ or $[0,100]$, etc. As the worth can be explained as importance or utility of the number, in such a context, many objective continuous domain can be normalized to $S=[0,1]$. A value assessing function U will map the physical quantity from Ω to S:

$$U:\Omega\to S$$

In this research, we only focus on the factors of time t and space $\vec{r}$ and suppose the value assess and discount are isotropic since sensors' sensing area are usually assumed to be isotropic. The interaction between the objective domain and subjective domain is handled in the view of *operator and function*. When a data V is collected by a sensor, its number does not change, and will be assessed to have an initial value $X(\vec{0},0)=U(V)$, the moment the data is collected is treated as the origin of time, and the locale as the origin of space.

In general, the value X will decrease when time flows by or viewed from a distance away from the locale. For $t_1,t_2\geq 0, |\vec{r}_1|, |\vec{r}_2|\geq 0$, the *discounted* value, $X(\vec{r}_2,t_2)$, of $X(\vec{r}_1,t_1)$ at space time $(\vec{r}_2,t_2)$ is given by

$$X(\vec{r}_2,t_2)=X(\vec{r}_1,t_1)\bigstar M(\vec{r}_2,t_2,\vec{r}_1,t_1)$$

where the map

$$M:\mathit{\Gamma}\times\mathbb{R}^+\cup\{0\}\times\mathit{\Gamma}\times\mathbb{R}^+\cup\{0\}\to[0,1]$$

is referred to as a *discount function* that describes the effect of discount and ★ is referred to as a *discount operator* that describes the way how a discount exert on the value. $\mathit{\Gamma}$ is determined by the underline coordinate. To avoid trivialities, in the sequel of this work we shall assume that $X(\vec{0},0) \neq 0$. If the effect on value from space and time is independent, then we can decouple the discount function into two discount functions and so do the discount operators:

$$X(\vec{r}_2,t) = X(\vec{r}_1,t)\spadesuit\Phi(\vec{r}_2,t,\vec{r}_1,t) \text{ and } X(\vec{r},t_2) = X(\vec{r},t_1)\clubsuit\Psi(\vec{r},t_2,t_1)$$

In which ♠ and ♣ are spacial and temporal discount operators respectively, Φ and Ψ are spacial and temporal discount functions respectively. Figure 1 is an illustration of isotropic discounting in 2D space.

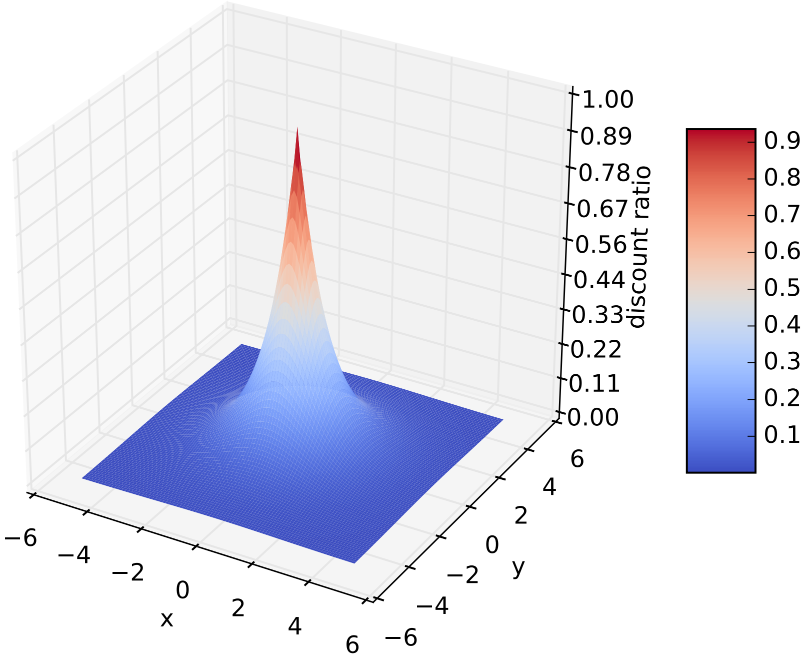

FIG. 1: Isotropic discount in 2D space

When space and time discount effect are decoupled, we can focus on each separately. In our research, we mainly focus on the effect of temporal discounting on the value of information.

3.2 TIME-DISCOUNTED INFORMATION

For the time discount effect, in its most general form, for $t \geq r$, the *discounted* value, $X(t)$, of X at time t is given by

$$X(t) = X(r) \clubsuit g(r,t) \tag{3.1}$$

where $g : \mathbb{R}^+ \cup \{0\} \times \mathbb{R}^+ \cup \{0\} \to [0,1]$ is referred to as a *time discount function*. Intuitively, if $r = t$, there is no discount, i.e.,

$$X(t) = X(t) \clubsuit g(t,t) \quad \forall\, t(0 \leq t). \tag{3.2}$$

Take $f_\clubsuit(x,y) = x \clubsuit y$ as the functional view of the discount operator $\clubsuit$, rewrite equation (3.2) as

$$X(t) = f_\clubsuit(X(t), g(t,t)).$$

In other words, $X(t)$ is a *fixed point* of function $f_\clubsuit$ at the time when the data is obtained and there is no decay at the beginning. From the view of algebra, $g(t,t)$ is the *identity element* of $\clubsuit$.

A discount operator $\clubsuit$ is called *cascade-able* if for all discount functions $g(\cdot,\cdot)$ we have

$$g(r,t) = g(r,s) \clubsuit g(s,t) \quad \forall\, r,s,t(0 \leq r \leq s \leq t).$$

Which says for cascade-able discount operators, for any discount effect represented by the discount function $g(\cdot,\cdot)$ on two consecutive time periods, can be cascaded in the way represented by the discount operator $\clubsuit$.

We stated a theorem here about cascade-able discount operators with proof.

Theorem 3.2.1. *If $\phi(y) = f_\clubsuit(x_0, y)$ is a bijective function of y with any given x_0, then $\clubsuit$ is cascade-able.*

Proof. For any r,s,t that $0 \leq r \leq s \leq t$, by equation (3.1), we have

$$X(t) = X(r) \clubsuit g(r,t),$$

$$X(t) = X(s) \clubsuit g(s,t)$$

and

$$X(s) = X(r) \clubsuit g(r,s).$$

Through simple algebra, we get

$$X(r)\clubsuit g(r,t) = X(r)\clubsuit g(r,s)\clubsuit g(s,t).$$

Rewrite the result in the functional view of $\clubsuit$,

$$f_{\clubsuit}(X(r),g(r,t)) = f_{\clubsuit}(X(r),g(r,s)\clubsuit g(s,t)),$$

if $\phi(y) = f_{\clubsuit}(x_0,y)$ is a bijective function of y with any given x_0, then

$$g(r,t) = g(r,s)\clubsuit g(s,t),$$

i.e. $\clubsuit$ is cascade-able. □

Corollary 3.2.2. *If $\clubsuit$ is cascade-able, then for any partition of time period [0,t] into $0 \le t_0 \le t_1 \le t_2 \le t_3 \le \cdots \le t_n$,*

$$g(t_0,t_n) = g(t_0,t_1)\clubsuit g(t_1,t_t2)\clubsuit\cdots\clubsuit g(t_{n-1},t_n).$$

Denote $k(t)$ as a kernel function of time t, and $k(t)$ is strictly monotonically increasing, $k(t)$ implicates some temporal structures of time t. Typical kernel functions include:

- Polynomial kernel function: $k(t) = a_nt^n + a_{n-1}t^{n-1} + \cdots + a_1t + a_0$;
- Linear kernel function: $k(t) = a_1t + a_0$ which is a special case of polynomial kernel function.

A discount function is called *additively parametrically separable* if

$$g(r,t) = \delta(k(t) - k(r))$$

An interesting special case is

$$g(r,t) = \delta(t^n - r^n) \quad n \ge 1, n \in \mathbb{N}.$$

For those theorems in Section 3.2.1 and 3.2.3, many of them can be generalized with additively parametrically separable discount functions, even though they are special cases with $g(r,t) = \delta(t-r)$. For simplicity, we do not include them in our thesis.

In a variety of practical applications, the discount function in (3.1) is, actually, a function of the difference $t-r$ only, that is, a function of the difference between the time of data collection and the current time, which is a very special case of additively parametrically separable discount functions. With this assumption, we are interested in discount functions satisfying the condition

$$X(t) = X(r) \clubsuit \delta(t-r) \tag{3.3}$$

with $\delta : \mathbb{R}^+ \cup \{0\} \longrightarrow [0,1]$.

Equation (3.3) tells us that the *penalty* of waiting for $t-r$ time is that the value of the information collected by the sensor decreases from $X(r)$ to $X(t)$. Figure 2 is an illustration of the temporal decaying of value X from time r to time t.

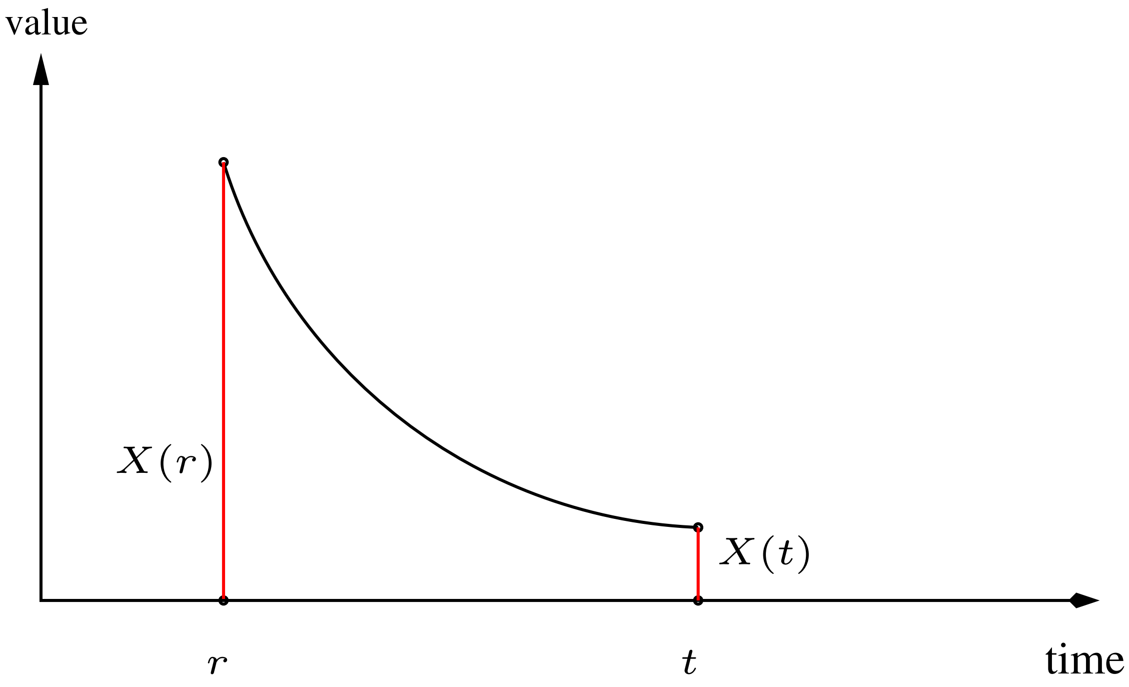

FIG. 2: Illustrating temporal discount of information

By taking the discount operator $\clubsuit$ in equation (3.3) to be specific operators and apply specific constrains on the discount function $\delta(t-r)$ such as multiplication or division and addition or subtraction, which correspond to discount by ratio and by amount respectively. As a consequence, we get two interesting special results: exponential discount function and linear discount function.

For a general multiplicative discount function, where the discount operator ♣ is multiplication, $\forall 0 \leq r \leq t$ and $\forall s, r \leq s \leq t$, we have

$$X(t) = X(s)\delta(t,s), \quad X(s) = X(r)\delta(s,r), \quad X(t) = X(r)\delta(t,r).$$

If a factor of $\delta(t,r) = f(t,r)u(t-r)$ exists and $f(t,r)$ is a constant, then $\delta(t,r) = Cu(t-r), C = f(t,r)$ and $x = t - r$ which only depends on the difference of t and r, we call it **global uniform discount behavior**; if $f(t,r)$ is not constant and only depends on t or r, then with a fixed t or r, we still have kind of uniformity which depends on t or r, we call it **local uniform discount behavior**. We explored a specified family of multiplicative discount functions with global uniform discount behavior in section 3.2.1 in which $C = f(t,r) = 1$.

For a general additive discount function, we have similar conclusions as well.

3.2.1 EXPONENTIAL TIME-DISCOUNT FUNCTION

For multiplicative discount, the discount is taken as a decaying ratio $\delta(t)$. It is intuitive to see:

- At the time the value is obtained, the decay begins, i.e. $\delta(0) = 1$;
- The decaying ratio only depends on the difference of time, i.e. it has global uniform discount behavior;
- Initial value will vanish after a long time.

Based on these assumption, $r(t)$ can be quantified. By taking the discount operator ♣ in equation (3.3) to be multiplication, based on assumption one, we have

$$X(t) = X(r) \cdot \delta(t - r) \tag{3.4}$$

Obviously, $X(r) = X(r)\delta(0)$ implies

$$\delta(0) = 1. \tag{3.5}$$

Further, we assume that after a very long time, the value of information vanishes. Formally, we assume that

$$\lim_{x \to \infty} \delta(x) = 0. \tag{3.6}$$

We begin by proving the following useful result that will be instrumental in obtaining a closed form for δ.

Lemma 3.2.3. *If $X(r) \neq 0$ then for all $r,\ s,\ t$ with $0 \leq r \leq s \leq t$*

$$\delta(t-r) = \delta(s-r)\delta(t-s). \tag{3.7}$$

Proof. Applying (3.4) to the pairs $(r,s),\ (s,t),\ (r,t)$ we obtain $X(s) = X(r)\delta(s-r)$, $X(t) = X(s)\delta(t-s)$ and $X(t) = X(r)\delta(t-r)$ which, combined, yield

$$X(r)\delta(t-r) = X(r)\delta(s-r)\delta(t-s).$$

Since $X(r) \neq 0$ the conclusion follows. □

Observe that by virtue of (3.6), δ cannot be identically 1 on $\mathbb{R}^+ \cup \{0\}$. Our next result shows that, in fact, δ takes on the value 1 if and only if $x = 0$.

Lemma 3.2.4. *$\delta(x) = 1$ if and only if $x = 0$.*

Proof. Recall that by (3.5), if $x_0 = 0$ then $\delta(x_0) = 1$. To prove the converse, let x_0 be the *largest* non-negative real for which $\delta(x_0) = 1$. It suffices to show that $x_0 = 0$. Suppose not and consider $\delta(2x_0)$. We can write

$$\begin{aligned} \delta(2x_0) &= \delta(2x_0 - 0) \\ &= \delta(2x_0 - x_0)\delta(x_0 - 0) \ \ \text{[by (3.7)]} \\ &= \delta(x_0)\delta(x_0) \\ &= 1, \ \ [\text{since } \delta(x_0) = 1] \end{aligned}$$

contradicting the maximality of x_0. Thus, $x_0 = 0$ and the proof of the lemma is complete. □

Corollary 3.2.5. *For all $x > 0,\ 0 < \delta(x) < 1$.*

Proof. Follows immediately from (3.5) and Lemma 3.2.4, combined. □

For all $r,\ s,\ t$ with $0 \leq r \leq s \leq t$ let x and y stand for $s-r$ and $t-s$, respectively. In this notation, $t - r = x + y$ and (3.7) can be written in the equivalent form

$$\delta(x+y) = \delta(x)\delta(y) \tag{3.8}$$

with both x and y non-negative. As it turns out, the functional equation (3.8) has a simple solution that we discuss next.

Theorem 3.2.6. *If the function*

$$f : [0,\infty) \longrightarrow \mathbb{R}$$

satisfies the functional equation $f(x+y) = f(x)f(y)$ and is not identically zero then there exists a constant a such that

$$f(x) = e^{ax} \tag{3.9}$$

The solution is also unique in this context. Our proof of Theorem 3.2.6 can be found in [80].

We are now in a position to show that the discount function δ is, in fact, an exponential. The details are spelled out by the following theorem.

Theorem 3.2.7. *For all r and t with $0 \leq r \leq t$,*

$$\delta(t-r) = e^{-\mu(t-r)}$$

where

$$\mu = -\ln\delta(1) > 0$$

Proof. Recall that by (3.8) the discount function δ satisfies the conditions of Theorem 3.2.6. Moreover, by Corollary 3.2.5 $0 < \delta(1) < 1$ and so $\ln\delta(1) < 0$. Thus, with

$$\mu = -\ln\delta(1) > 0$$

the expression of $\delta(t-r)$ becomes $\delta(t-r) = e^{-\mu(t-r)}$, as claimed. □

Theorem 3.2.7 shows that, under mild assumptions, the discount function is an exponential. We note that a similar result was derived by [75, 82] in the case of discrete time.

In Figure 3, three exponential time-discount functions with increasing *discount rate* μ are illustrated, for simplicity, $r = 0$.

After the determination of exponential time-discount function, the value at time t of any initial value $X(r)$ from time r with discount rate μ can be determined by equation (3.10).

$$X(t) = X(r)e^{-\mu(t-r)} \qquad (t \geq r) \tag{3.10}$$

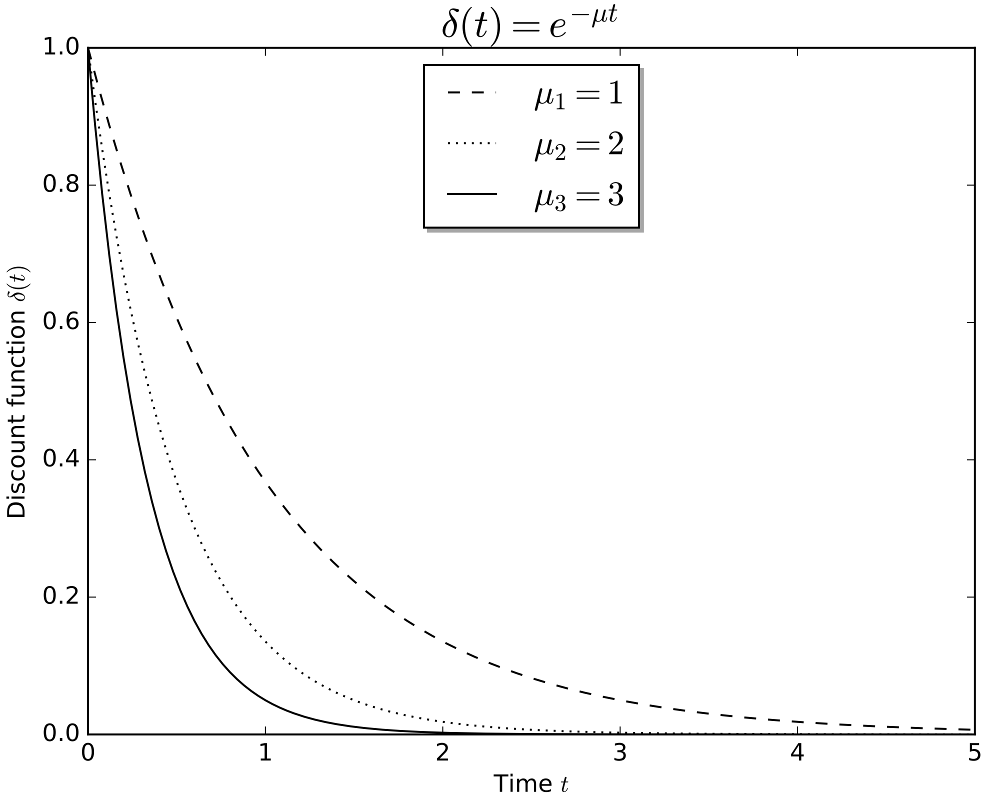

FIG. 3: Three exponential time-discount functions with increasing discount constant

Exponential time-discount function shows that the value of information decays exponentially rapid with time. So it is extremely import to find a way of aggregation to offset this decaying effect. What is the meaning of this discount rate μ?

There is almost an exact analogy between information discount and exponential decay of atom [2], which can be used to determine the meaning of μ and how to get the value of μ.

By taking derivative on both sides of equation (3.10), we find exponentially decayed value decays at rate proportional to its current value.

$$\begin{aligned}\frac{dX}{dt} &= -\mu X(r)e^{-\mu(t-r)} \\ &= -\mu X(t)\end{aligned}$$

Let $\tau = \frac{1}{\mu}$, τ is defined as *exponential time constant* or *mean life time* in exponential decay [2], then equation (3.10) can be rewritten as:

$$X(t) = X(r)e^{-\frac{t-r}{\tau}}$$

Let us take a few special time points to get a feel for this exponential decay:

- $t-r=0, \quad X(r+0\tau) = X(r)$;
- $t-r=\tau, \quad X(r+\tau) = X(r)e^{-1} \approx 0.3679X(r)$;
- $t-r=5\tau, \quad X(r+5\tau) = X(r)X(r)e^{-5} \approx 0.0067X(r) < 1\%X(r)$. This is considered as “vanished”;
- Half-life: $X(r+t_{1/2}) = \frac{1}{2}X(r) \Rightarrow t_{1/2} = \tau \ln 2 \approx 0.693\tau$, i.e. after half-life, a value is left only a half.

In order to find why τ is called *mean life time*, suppose X is an exponentially distributed random variable, then

$$\begin{aligned}\text{its cumulative distributive function is} \quad & F_X(t) = Pr\{X \le t\} = 1 - e^{-\mu t}, \\ \text{its expectation is} \quad & E(X) = \frac{1}{\mu} = \tau, && (3.11) \\ \text{and its probability density function is} \quad & f_X(t) = \frac{dF}{dt} = \mu e^{-\mu t}. && (3.12)\end{aligned}$$

From the similarity between equation (3.10) and (3.12), using integration by parts, equation (3.10) can be normalized into an exponentially distributed probability density function (3.12) with a constant c,

$$\begin{aligned} 1 &= \int_r^\infty cX(r)e^{-\mu(t-r)}dt = \frac{cX(r)}{\mu} \\ \Rightarrow c &= \frac{\mu}{X(r)}. \end{aligned}$$

Then equation (3.10) can be rewritten as

$$X(t) = \frac{X(r)}{\mu}\mu e^{-\mu(t-r)} \qquad (t \geq r) \tag{3.13}$$

From equation (3.13), it is clear that exponential time-discount $X(t)$ is a scalar $\frac{X(r)}{\mu}$ multiple of the exponential distribution $\mu e^{-\mu(t-r)}$ (i.e. the individual lifetime of each value is exponentially distributed), which has a well-known expected value (by equation (3.11)) can be computed as below:

$$\tau = \overline{t-r} = \int_r^\infty (t-r)cX(r)e^{-\mu(t-r)}dt = \frac{1}{\mu}. \tag{3.14}$$

Hence, based on historical data, we can get estimates of μ or τ.

3.2.2 A RELATED EXPONENTIAL TIME-DISCOUNT FUNCTION

The goal of this subsection is to present a variant of the discount function δ defined in (3.4). Indeed, in some applications, the discounted value of the information at time $t > r$ may be naturally expressed as

$$X(t) = X(r) - X(r)\psi(t-r) \tag{3.15}$$

where $\psi : \mathbb{R}^+ \cup \{0\} \to [0,1]$. As we are about to show, the discount functions δ and ψ are related in a simple fashion. To see this, consider three reals $r,\ s,\ t$ with $0 \leq r < s < t$ and assume that $X(r) \neq 0$. It is easy to see that, by virtue of (3.15), we can write

$$\begin{aligned} X(t) &= X(r)[1-\psi(t-r)] \\ &= X(r)[1-\psi(t-s)][1-\psi(s-r)] \end{aligned}$$

implying that

$$1-\psi(t-r)=[1-\psi(t-s)][1-\psi(s-r)].$$

By writing $\xi(t-r)=1-\psi(t-r)$, $\xi(s-r)=1-\psi(s-r)$ and $\xi(t-s)=1-\psi(t-s)$ we obtain

$$\xi(t-r)=\xi(s-r)\xi(t-s)$$

which is, essentially, identical to (3.7). As an immediate consequence of Theorem 3.2.7

$$1-\psi(t-r)=e^{-\lambda(t-r)}$$

where $\lambda=-\ln[1-\psi(1)]>0$.

Finally, simple algebra reveals that

$$X(t)=X(r)e^{-\lambda(t-r)}$$

which is of the same form as (3.4) and so the discount functions δ and ψ are, essentially, the same.

3.2.3 LINEAR TIME-DISCOUNT FUNCTION

In another typical case, discount of information can be taken as an absolute portion from the original value. We quantified this type of discounting in a way similar to exponential time-discount function based on the following intuitions:

- At the beginning $t=0$, no discount is taken from the initial value, i.e. $\delta(0)=0$;
- At a certain time T, the value vanished, i.e. $X(T)=0$, so the corresponding lifespan is T;
- *Uniform discount:* same portion of value is discounted for same length of time, i.e. $\delta(t-r)=\delta(r-s)$ as long as $t-r=r-s$, where $0\leq r\leq s\leq t$.

Following these assumption, by taking the discount operator $\clubsuit$ in equation (3.3) to be addition, we have

$$X(t)=X(r)+\delta(t-r) \tag{3.16}$$

Obviously, $X(r)=X(r)+\delta(0)$ implies

$$\delta(0)=0. \tag{3.17}$$

Further, we assume that after a certain time, the value of information becomes zero. Formally, we assume that

$$\exists T : 0 = X(r) + \delta(T - r). \text{ i.e. } \delta(T - r) = -X(r). \tag{3.18}$$

Let $r = 0$, we have a neat formula $\delta(T) = -X(0)$, i.e., at time T, the value of X decayed to zero. We begin by proving the following useful result that will be instrumental in obtaining a closed form for δ.

Lemma 3.2.8. *If $X(r) \neq 0$ then for all $r,\ s,\ t$ with $0 \leq r \leq s \leq t \leq T$*

$$\delta(t - r) = \delta(s - r) + \delta(t - s). \tag{3.19}$$

Proof. Applying (3.16) to the pairs $(r,s),\ (s,t),\ (r,t)$ we obtain $X(s) = X(r) + \delta(s - r)$, $X(t) = X(s) + \delta(t - s)$ and $X(t) = X(r) + \delta(t - r)$ which, combined, yield

$$X(r) + \delta(t - r) = X(r) + \delta(s - r) + \delta(t - s).$$

the conclusion follows. □

Observe that by virtue of (3.18), δ cannot be identically 0 on $\mathbb{R}^+ \cup \{0\}$. Our next result shows that, in fact, δ takes on the value 0 if and only if $x = 0$.

Lemma 3.2.9. *$\delta(x) = 0$ if and only if $x = 0$.*

Proof. Recall that by (3.17), if $x_0 = 0$ then $\delta(x_0) = 0$. To prove the converse, let $x_0 \leq x \leq T$ be the *largest* non-negative real for which $\delta(x_0) = 0$. It suffices to show that $x_0 = 0$. Suppose not and consider $\delta(2x_0)$. We can write

$$\begin{aligned}
\delta(2x_0) &= \delta(2x_0 - 0) \\
&= \delta(2x_0 - x_0) + \delta(x_0 - 0) \text{ [by (3.19)]} \\
&= \delta(x_0) + \delta(x_0) \\
&= 0, \text{ [since } \delta(x_0) = 0]
\end{aligned}$$

contradicting the maximality of x_0. Thus, $x_0 = 0$ and the proof of the lemma is complete. □

Corollary 3.2.10. *For all $T \geq t \geq 0,\ \delta(T) \leq \delta(t) \leq 0$.*

Proof. Follows immediately from (3.17) and Lemma 3.2.9, combined. □

For all $r,\ s,\ t$ with $0 \leq r \leq s \leq t \leq T$ let x and y stand for $s-r$ and $t-s$, respectively. In this notation, $t-r = x+y$ and (3.19) can be written in the equivalent form

$$\delta(x+y) = \delta(x) + \delta(y) \tag{3.20}$$

with both x and y non-negative. As it turns out, the functional equation (3.20) has a simple solution that we discuss next.

Theorem 3.2.11. *If the function $f : [0,\infty) \longrightarrow \mathbb{R}$ satisfies the functional equation $f(x+y) = f(x) + f(y)$ and is not identically constant then there exists a coefficient a such that*

$$f(x) = ax \tag{3.21}$$

The above functional equation is the well-known Cauchy equation. Its proof can be found in [54]. Actually, function equation (3.20) can be obtained from the functional equation (3.8) by taking the logarithm on both sides.

We are now in a position to show that the discount function δ is, in fact, linear. The details are spelled out by the following theorem.

Theorem 3.2.12. *For all r and t with $0 \leq r \leq t \leq T$,*

$$\delta(t-r) = -\mu(t-r)$$

where

$$\mu = -\delta(1) > 0$$

Proof. Recall that by (3.20) the discount function δ satisfies the conditions of Theorem 3.2.11. Thus, with

$$\mu = -\delta(T) > 0$$

the expression of $\delta(t-r)$ becomes $\delta(t-r) = -\mu(t-r)$, as claimed. □

Theorem 3.2.12 shows that, under mild assumptions, the discount function is a linear.

For simplicity and without losing generality, let $r = 0$, the formula in theorem 3.2.12 becomes $\delta(t) = -\mu t$, substitute into equation 3.16, a linear equation is gotten:

$$X(t) = X(0) - \mu t \tag{3.22}$$

Equation (3.22) shows during the lifespan T of value $X(0)$, the value discounts uniformly at a constant rate μ. Substitute $t = T$ into equation (3.22), we get

$$\mu = \frac{X(0)}{T} \tag{3.23}$$

Three linear time-discount functions with decreasing discount constants are illustrated in Figure 4, where $X(0) = 1.0$, at certain time $t = T = \frac{1}{\mu}$, $X(t)$ decreased to zero.

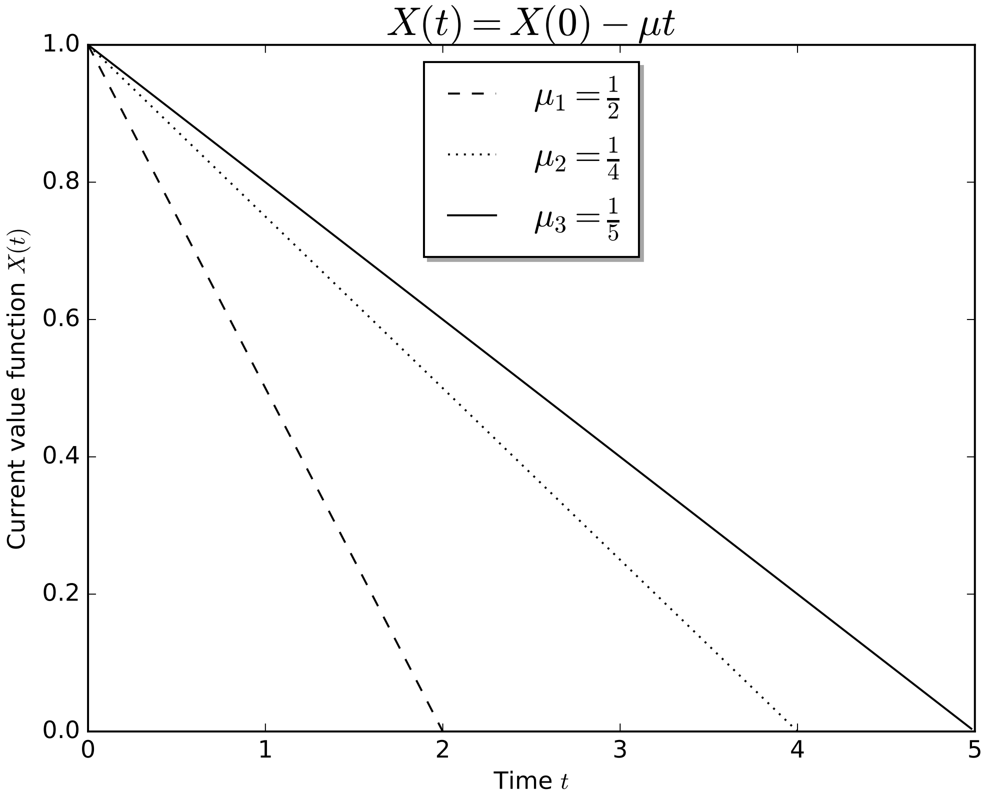

FIG. 4: Three linear time-discount functions with decreasing discount constant

3.3 FUNCTION VIEW OF INFORMATION DISCOUNT

The general information discount function

$$X(\vec{r}_2, t_2) = X(\vec{r}_1, t_1) \bigstar M(\vec{r}_2, t_2, \vec{r}_1, t_1)$$

can be viewed as

$$X(\vec{r}_2, t_2) = f(X(\vec{r}_1, t_1), \vec{r}_2, t_2, \vec{r}_1, t_1)$$

and simply denoted as

$$X_2 = f(X_1, \vec{r_2}, t_2, \vec{r_1}, t_1)$$

if the underline coordinates are polar or spherical, the notation can become even simpler, $X_2 = f(X_1, r_2, t_2, r_1, t_1)$, where function f is termed as a *general discount function* and $r_2 = |\vec{r_2}|, r_1 = |\vec{r_1}|$.This view will give us general handling on information discount as when the value and discount factors are interacted together and inseparable. The only constraints on the general discount function is that f is non-increasing with time t and non-increasing with distance $|\vec{r}|$ with assumption that the spatial discount is isotropic.

The combination of discount operators and discount functions are special cases of this general discount function where the value and discount factors are separable.

CHAPTER 4

ALGEBRA OF DATA AGGREGATION

4.1 GENERAL DATA AGGREGATION

To find the patterns hidden in a set of data, we define an algebraic operation that we call *aggregation*. Consider two sensors that have collected data about an event. Let X and Y be, respectively, the data or value collected by the two sensors. The two sensors decide to aggregate their information in order to find some emerging patterns.

By definition, the aggregated result of X and Y is $X \diamondsuit Y$. The aggregation operator $X \diamondsuit Y$ can be extended, in the obvious way, to an arbitrary number of values that need to be aggregated. Sometimes, we need only a little portion of sensors to detect an event; sometimes, we need a large portion of sensors and a long period to monitoring the developing of a event; sometimes, we need the whole wireless sensor network to find global patterns such as the maximum, average and minimum temperature, humidity and air pressure of a region in order to monitor its climate.

4.2 INSTANTANEOUS AGGREGATION

Information aggregation can be thought of as being orthogonal to the effect of time on its value. In this section we discuss the basics of aggregation and look at the aggregation of information independent of time discounting. the full effect of time discounting and its effect on aggregation will be discussed in detail in Section 4.3.

Assume a number of sensors have witnessed a short-lived event (e.g. an explosion) and have collected relevant information. In some applications it may well be feasible to proceed with a relatively simple aggregation. Specifically, let $X_1, X_2, \cdots, X_n, \cdots$ be the readings of a relevant attribute (e.g., seismic tremors caused by the explosion mentioned above) collected by the sensors. Due to spatial diversity of sensor locations, we assume that the X_is are independent random variables with a common underlying distribution function F. We further assume that the X_is have finite first and second moments.

In order to proceed to an instantaneous aggregation the data collected by the sensors, we define an application-dependent threshold Δ and seek to characterize those readings X_n

that exceed Δ. What is the expected value and variance of such an X_n? To answer this question, let G be the distribution function of X_n.

For $x \geq \Delta$,

$$\begin{aligned} G(x) &= \Pr[X_n \leq x] \\ &= \Pr[X \leq x \mid X > \Delta] \\ &= \frac{\Pr[\{X \leq x\} \cap \{X > \Delta\}]}{\Pr[X > \Delta]} \\ &= \frac{\Pr[\Delta < X \leq x]}{1 - F(\Delta)} \\ &= \frac{F(x) - F(\Delta)}{1 - F(\Delta)}. \end{aligned} \tag{4.1}$$

By (4.1), we write

$$G(x) = \begin{cases} 0 & \text{for } x < \Delta \\ \\ \frac{F(x)-F(\Delta)}{1-F(\Delta)} & \text{for } x \geq \Delta. \end{cases} \tag{4.2}$$

Now that we have the distribution function of X_n we turn our attention to computing both $E[X_n]$ and $Var[X_n]$. First, by definition,

$$\begin{aligned} E[X_n] &= \int_0^\infty u \, \mathrm{d}G(u) \\ &= \int_0^\Delta u \, \mathrm{d}G(u) + \int_\Delta^\infty u \, \mathrm{d}G(u) \;\; \text{[by (4.2)]} \\ &= \int_\Delta^\infty u \, \mathrm{d}G(u) \\ &= \int_\Delta^\infty u \frac{\mathrm{d}F(u)}{1 - F(\Delta)} \;\; \text{[by (4.2)]} \\ &= \frac{1}{1 - F(\Delta)} \int_\Delta^\infty u \, \mathrm{d}F(u). \end{aligned} \tag{4.3}$$

It would be better to show that $E[X_n] > \Delta$. Note that

$$\begin{aligned} \int_\Delta^\infty u \, \mathrm{d}F(u) > \int_\Delta^\infty \Delta \, \mathrm{d}F(u) \qquad (\because u \geq \Delta) \\ = \Delta \int_\Delta^\infty \mathrm{d}F(u) = \Delta(1 - F(\Delta)) \\ \Rightarrow E[X_n] > \Delta \end{aligned}$$

as our intuition suggests.

To evaluate $Var[X_n]$ we proceed as follows:

$$\begin{aligned} Var[X_n] &= E[X_n^2] - E[X_n]^2 \\ &= \int_{\Delta}^{\infty} u^2 \,\mathrm{d}G(u) - E[X_n]^2 \\ &= \frac{1}{1-F(\Delta)} \int_{\Delta}^{\infty} u^2 \,\mathrm{d}F(u) \\ &- \left(\frac{1}{1-F(\Delta)} \int_{\Delta}^{\infty} u \,\mathrm{d}F(u) \right)^2 . \end{aligned} \tag{4.4}$$

We note that if the threshold Δ as well as the distribution function F are known to the sensors, each of them can determine the conditional expectation and variance of those readings that exceed Δ. This trick allows the sensors to judge if the values they hold are "relevant". The sensors with relevant data will transmit them without delay to a local sink which will make the final determination.

4.3 AGGREGATION: COUNT-ACTING DISCOUNTING

To counter-act the effect of time discounting, we define an algebraic operation on sensor data that we call *aggregation*. Consider two sensors that have collected data about an event at times r and s, respectively. Let $X(r)$ and $Y(s)$ be, respectively, the values of the information collected by the two sensors. At some later time τ, the two sensors decide to aggregate their information.

By definition, the aggregated value of $X(r)$ and $Y(s)$ at time t, with $0 \le r \le t$ and $0 \le s \le t$, is $X(t) \diamondsuit Y(t)$. By (3.4), it follows that

$$X(t) \diamondsuit Y(t) = [X(r) \clubsuit \delta(t-r)] \diamondsuit [Y(s) \clubsuit \delta(t-s)] . \tag{4.5}$$

Thus, what is being aggregated at time t are the discounted values $X(r) \clubsuit \delta(t-r)$ and $Y(s) \clubsuit \delta(t-s)$. The aggregation operator can be extended, in the obvious way, to an arbitrary number of values that need to be aggregated.

Useful instances of $\diamondsuit$ include $+$, max, min, XOR, OR, among many others. It is worth observing that the type of aggregation operator that should be applied is application-dependent. In fact, an aggregation operator that makes sense in one application may be irrelevant in a different context.

Most of the aggregation operators $\diamondsuit$ of practical relevance have the following fundamental properties:

Commutativity: $X(t)\Diamond Y(t) = Y(t)\Diamond X(t)$ for all $t \geq 0$. In other words, the result of the aggregation does not depend on the order in which the values are aggregated.
Associativity: $[X(t)\Diamond Y(t)]\Diamond Z(t) = X(t)\Diamond[Y(t)\Diamond Z(t)]$ for all $t \geq 0$. If several values are aggregated in groups, the value of the aggregated information does not depend on the order in which groups are formed. We shall follow established practice to write $X(t)\Diamond Y(t)\Diamond Z(t)$ instead of the cumbersome parenthesized expressions. A straightforward inductive argument shows that if $\Diamond$ is associative then for an arbitrary collection of n values $X_1(t),\ X_2(t),\ \cdots,\ X_n(t)$ we have

$$\begin{aligned}
&[X_1(t)\Diamond X_2(t)\Diamond\cdots\Diamond X_{n-1}(t)]\Diamond X_n(t)\\
&= X_1\Diamond[X_2(t)\Diamond X_3(t)\Diamond\cdots\Diamond X_n(t)]\\
&= X_1\Diamond X_2(t)\Diamond X_3(t)\Diamond\cdots\Diamond X_n(t).
\end{aligned}$$

To simplify the notation, when aggregating several values $X_1,(t)\ X_2(t),\ \cdots,\ X_n(t)$, we shall write $\Diamond_{i=1}^{n} X_i(t)$ instead of $X_1(t)\Diamond X_2(t)\Diamond\cdots\Diamond X_n(t)$.
Idempotency: If $Y(t) = 0$ then $X(t)\Diamond Y(t) = X(t)$. In other words, aggregation with data of value 0 has no effect.

4.4 A TAXONOMY OF AGGREGATION OPERATORS

Consider, again, two sensors that have collected, at time t, data about some attribute of an event they have witnessed and let $X(t)$ and $Y(t)$ be, respectively, the values of the information collected. It is important for $X(t)$ and $Y(t)$ to be aggregated in order to obtain a more reliable and, perhaps, more relevant information about the event at hand.

Suppose, further, that aggregation takes time and that the aggregated information is available at time $\tau > t$. Given that the value of the data decays over time exponentially according to (3.4), and given an aggregation operator $\Diamond$, what strategy maximizes the value of the information at time τ?

The answer to this question depends on the type of aggregation operator used. Indeed, the aggregation operator $\Diamond$ can be one of three distinct types defined as follows:

- **Type 1:** For all $t,\ \tau$ with $0 \leq t \leq \tau$, $[X(t)\Diamond Y(t)]\,\delta(\tau - t) < X(\tau)\Diamond Y(\tau)$. In other words, in the case of a Type 1 operator, it is best to defer aggregation as long as the semantics of the application permit;

- **Type 2:** For all $t,\ \tau$ with $0 \leq t \leq \tau$, $[X(t)\Diamond Y(t)]\,\delta(\tau - t) = X(\tau)\Diamond Y(\tau)$. As it is apparent, in the case of a Type 2 operator, the order between aggregation and discount does not matter, in reality, we need to aggregate as soon as possible because the value will decay quickly.

- **Type 3:** For all $t,\ \tau$ with $0 \leq t \leq \tau,\ [X(t)\diamondsuit Y(t)]\,\delta(\tau-t) > X(\tau)\diamondsuit Y(\tau)$. Thus, in the case of a Type 3 operator, the best strategy is to aggregate as early as the data is available and/or the semantics of the application permit.

- **Type 4:** For all $t,\ \tau$ with $0 \leq t \leq \tau$, the relationship between $[X(t)\diamondsuit Y(t)]\,\delta(\tau - t) and X(\tau)\diamondsuit Y(\tau)$ depends on both t and τ. Which is a *combination* of the previous three types. Thus, in the case of a Type 4 operator, the best strategy is adaptive to the type of aggregator for aggregating the data in the semantics of the application permit.

Observe that for a Type 2 operator, the discount function distributes over $\diamondsuit$. Consequently, for such an operator it does not matter whether we first aggregate and then discount the aggregated information or vice versa. In references [79, 80] we have studied extensively Type 2 operators and their properties, which will be presented in 4.8. For Type 1 operators, whose properties we will investigate in Section 4.6. Type 3 operators and the aggregation strategies they suggest, are conjugated to Type 1, whose properties will be given in section 4.9 for completeness without proof.

4.5 A FUNCTIONAL VIEW OF THE THREE TYPES OF AGGREGATORS

Similar to what we discussed in Section 3.3, every binary aggregation operator $\diamondsuit$ on two operands X,Y can be viewed as a function with two independent variables $f(X,Y)$:

$$f(X,Y) = X\diamondsuit Y.$$

For two values $X_1(r)$ and $X_2(r)$ collected at time r, considering exponential time-discount with discount rate μ, for a later time t $(t \geq r)$:

$$X_1(t) = X_1(r)e^{-\mu(t-r)}$$

and

$$X_2(t) = X_2(r)e^{-\mu(t-r)}.$$

In the comparison of

$$LHS = \{X_1(r)\diamondsuit X_2(r)\}e^{-\mu(t-r)}$$

and

$$RHS = X_1(r)e^{-\mu(t-r)}\diamondsuit X_2(r)e^{-\mu(t-r)}$$

we have:

- When $t = r$, $e^{-\mu(t-r)} = 1$, $LHS = X_1(r)\diamondsuit X_2(r) = RHS$;
- When $t \to \infty$, $e^{-\mu(t-r)} = 0$, $LHS = 0 = RHS$;
- When $r < t < \infty$, three types of aggregation operator distinguish from each other.
  - Type 1: $LHS < RHS$;
  - Type 2: $LHS = RHS$;
  - Type 3: $LHS > RHS$;

At the beginning or end, three types of aggregation operator agree. They differ from each other in the interval (r,+∞), shown in Figure 5. For Type 2, the curve of RHS is overlapped on LHS, shown as Type 2 in the figure; for Type 3, the curve of RHS is over LHS, shown as Type 3 in the figure; for Type 1, the curve of RHS is below LHS, shown as Type 1 in the figure.

4.6 AGGREGATORS OF TYPE 1

Let $\diamondsuit$ be an *arbitrary* Type 1 operator and assume that n, $(n \geq 2)$ sensors have collected data about an event at times $t_1, t_2, \cdots, t_n$. Let $X_1(t_1), X_2(t_2), \cdots, X_n(t_n)$ be, respectively, the values collected by the sensors. At some later time $\tau \geq \max_{1\leq i\leq n} t_i$, the sensors decide to aggregate their information. We take note of the following relevant result.

Lemma 4.6.1. *Assume an associative Type 1 operator* $\diamondsuit$. *For all* t, τ *with* $\max_{1\leq i\leq n}\{t_i\} \leq t \leq \tau$ *we have*

$$\left[\diamondsuit_{i=1}^{n} X_i(t)\right] \delta(\tau - t) < \diamondsuit_{i=1}^{n} X_i(\tau).$$

Proof. The proof is by induction on n. For $n = 2$, the conclusion follows straight from the definition of a Type 1 operator. For the inductive step, let $n \geq 3$ be arbitrary and assume

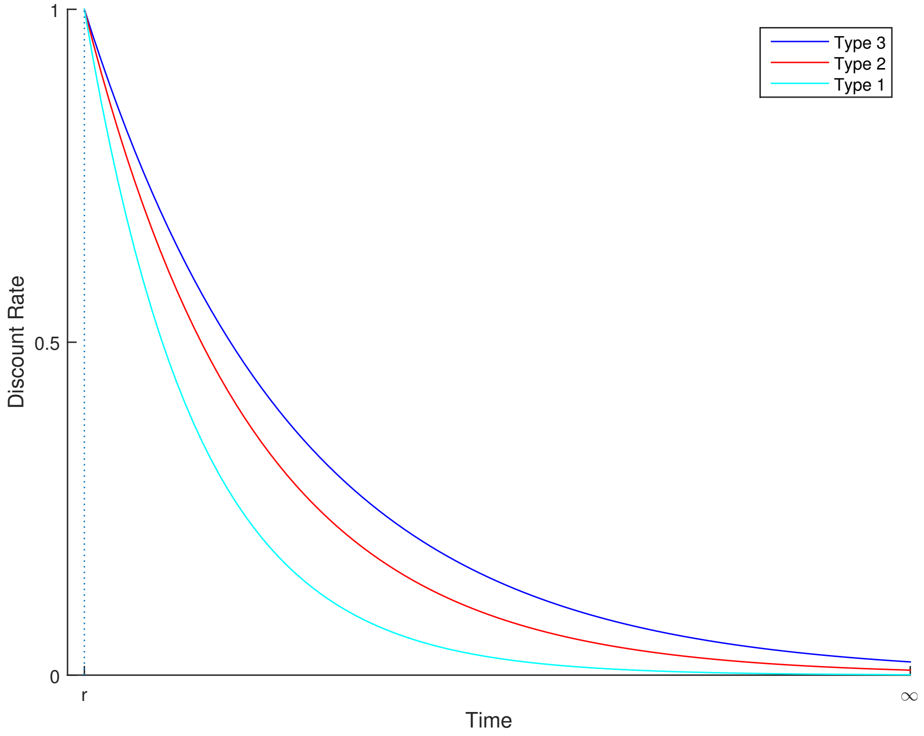

FIG. 5: *Effect on Information Discount of the Three Types of Aggregator.*

that the property holds for $n-1$ aggregated values. We write

$$\begin{aligned}
&[\diamondsuit_{i=1}^{n} X_i(t)]\,\delta(\tau-t)\\
&= \left[\left(\diamondsuit_{i=1}^{n-1} X_i(t)\right)\diamondsuit X_n(t)\right]\delta(\tau-t)\\
&< \left[\diamondsuit_{i=1}^{n-1} X_i(t)\delta(\tau-t)\right]\diamondsuit\,[X_n(t)\delta(\tau-t)]\\
&= \diamondsuit_{i=1}^{n}\,[X_i(t)\delta(\tau-t)]\\
&= \diamondsuit_{i=1}^{n} X_i(\tau),
\end{aligned}$$

completing the proof of Lemma 4.6.1. □

The left-hand side of the inequality above is the discounted value of $\diamondsuit_{i=1}^{n} X_i(t)$ at time τ, while the right-hand is the aggregated value of the discounted values of $X_i(t_i)$ at time τ. Lemma 4.6.1 asserts that the defining inequality of the Type 1 operator holds when an arbitrary number, n, of values are being aggregated.

Consider an event witnessed by n, $(n \geq 2)$, sensors and let the sensed values be $X_1, X_2, \cdots, X_n$, collected, respectively, at times $t_1, t_2, \cdots, t_n$. Assume, further, that various groups of sensors have aggregated their data before time t and that, finally, at time t the aggregation has been completed. We are interested in evaluating the time-discounted value of the information collected by the sensors at time t, where $t \geq \max\{t_1, t_2, \cdots, t_n\}$. The answer to this natural question is provided by the following fundamental result.

Theorem 4.6.2. *Assuming that the Type 1 aggregation operator $\diamondsuit$ is associative and commutative, the discounted value of the aggregated information at time t is upper-bounded by $\diamondsuit_{i=1}^{n} X_i(t)$, regardless of the order in which the values were aggregated.*

Proof. The proof is by induction on n. For $n = 2$, the conclusion follows at once from definition. Now, let $n \geq 3$, be arbitrary and assume the statement true for all m, $(m < n)$. We assume, without loss of generality, that the last aggregation takes place at time t. This aggregation must have involved a number of disjoint groups $G_1, G_2, \ldots, G_p$ each of them the result of a previous aggregation at times, respectively, u_1, u_2, $\ldots$, u_p. Observe that we can always relabel the groups in such a way that their aggregation times are ordered as $u_1 < u_2 < \cdots < u_p$.

Let us look at group G_k. By the induction hypothesis, the value of information in group G_k aggregated at time u_k is upper-bounded by $\diamondsuit_{j=1}^{n_k} X_{k_j}(u_k)$ where, of course, we assume that group G_k involves n_k sensors whose values were aggregated.

Assuming $t \geq u_k$, Lemma 4.6.1 guarantees that the discounted value at time t is upper-bounded by

$$\left[\diamondsuit_{j=1}^{n_k} X_{k_j}(u_k)\right] \delta(t - u_k) \quad < \quad \diamondsuit_{j=1}^{n_k} X_{k_j}(t),$$

which is an upper bound on the value of information collected by sensors in group G_k, had it been aggregated at time t. Since G_k was arbitrary, the conclusion follows. □

Theorem 4.6.2, in effect, says that the maximum value of the aggregated information that can be attained is independent of the order in which the values are aggregated. In practical terms, Theorem 4.8.2 gives the algorithm designer the freedom to schedule aggregation in a random manner, much in line with the stochastic nature of wireless communication and sensor data aggregation. Recall that for Type 1 operators, aggregation can be delayed as long as the semantics of the application permit. In the next subsection we look at a thresholding mechanism that defines how long it is feasible to delay aggregation.

4.7 A SPECIAL CLASS OF TYPE 1 AGGREGATOR

We begin by taking note of a non-trivial class of Type 1 aggregation operators that turns out to have interesting applications. Imagine that the data collected by sensors take on values in the range $[0,1]$ and consider the aggregation operator $\diamondsuit$ defined as

$$X(t)\diamondsuit Y(t) = X(t) + Y(t) - X(t)Y(t). \tag{4.6}$$

We leave it to the reader to verify that $\diamondsuit$ satisfies the associativity, commutativity and idempotency properties defined above. To prove that $\diamondsuit$ is a Type 1 operator, consider an arbitrary τ with $0 \le t \le \tau$ and write

$$\begin{aligned} LHS &= [X(t)\diamondsuit Y(t)]\,\delta(\tau - t) \\ &= [X(t) + Y(t) - X(t)Y(t)]\,\delta(\tau - t) \\ &= X(t)\delta(\tau - t) + Y(t)\delta(\tau - t) - X(t)Y(t)\delta(\tau - t) \\ &= X(\tau) + Y(\tau) - X(\tau)Y(t) \ \ \text{[by (3.4)]} \\ &< X(\tau) + Y(\tau) - X(\tau)Y(\tau) \ \ \text{[since } Y(\tau) < Y(t)\text{]} \\ &= X(\tau)\diamondsuit Y(\tau) = RHS \end{aligned}$$

confirming that the aggregation operator defined in (4.14) is of Type 1.

If denote $\sigma = e^{-\mu(t-r)} \in [0,1]$, and let

$$\begin{aligned} LHS &= \{X_1(r)\diamondsuit X_2(r)\}\sigma = \sigma X + \sigma Y - \sigma XY \\ RHS &= X_1(r)\sigma\diamondsuit X_2(r)\sigma = \sigma X + \sigma Y - \sigma^2 XY \end{aligned}$$

Then

$$\begin{aligned} LHS - RHS &= \sigma XY(\sigma - 1) \\ &< 0 \quad (\because\ 0 < \sigma < 1) \end{aligned} \tag{4.7}$$

Which also confirms that the aggregation operator defined in (4.14) is of Type 1.

Interestingly, rewrite this new representational equation 4.7 into

$$RHS - LHS = \sigma XY(1 - \sigma)$$

It is easy to see that there exists a maximum value of $RHS - LHS$ at a certain time t_m by the inequality of arithmetic and geometric means.

$$\begin{aligned} RHS - LHS &= \sigma XY(1-\sigma) \\ &= XY\sigma(1-\sigma) \\ &\leq XY\left(\frac{\sigma+(1-\sigma)}{2}\right)^2 \\ &= \frac{XY}{4} = \frac{X}{2}\times\frac{Y}{2} \end{aligned}$$

The equal sign satisfied when $\sigma = 1-\sigma \Rightarrow \sigma = \frac{1}{2}$, i.e. when half-life is passed, $t_m - r = t_{1/2} = \tau \ln 2 \approx 0.693\tau = \frac{0.692}{\mu}$, then $\max\limits_{t\in\{r,\infty\}}\{RHS - LHS\} = \frac{XY}{4}$. It says before t_m, the difference between $RHS - LHS$ increases, at t_m, a maximum of the difference is reached where both X and Y decreased to their half values, after t_m, the difference decreases however both X and Y begin vanishing. Even though Type 1 infers deferring aggregation as long as possible in the semantic of application, however, in order to get good enough aggregated value, it is better to aggregate before half-life.

For our special first type aggregation operator defined as

$$X\Diamond Y = X+Y-XY = f(X,Y) \quad X,Y \in [0,1] \tag{4.8}$$

Take it as a function with two independent variables, its partial derivatives are

$$\begin{aligned} \frac{\partial F}{\partial X} &= 1-Y > 0 && Y \in [0,1) \\ \frac{\partial F}{\partial X} &= 1-X > 0 && X \in [0,1) \end{aligned}$$

Which means the aggregated value of larger values is always larger than those from smaller values, i.e. it is monotonically increase with X and Y, so we need to aggregate values as soon as possible in order to get a value as larger as possible. Function (4.8) is illustrated in Figure 6.

In order to get an intuitive comprehension of the interaction between this special Type 1 aggregator and exponential time-discount on two values, we illustrated them in Figure 7, for simplicity, let $\mu = 0.25$, the below four curves are drawn. It clearly shows that this

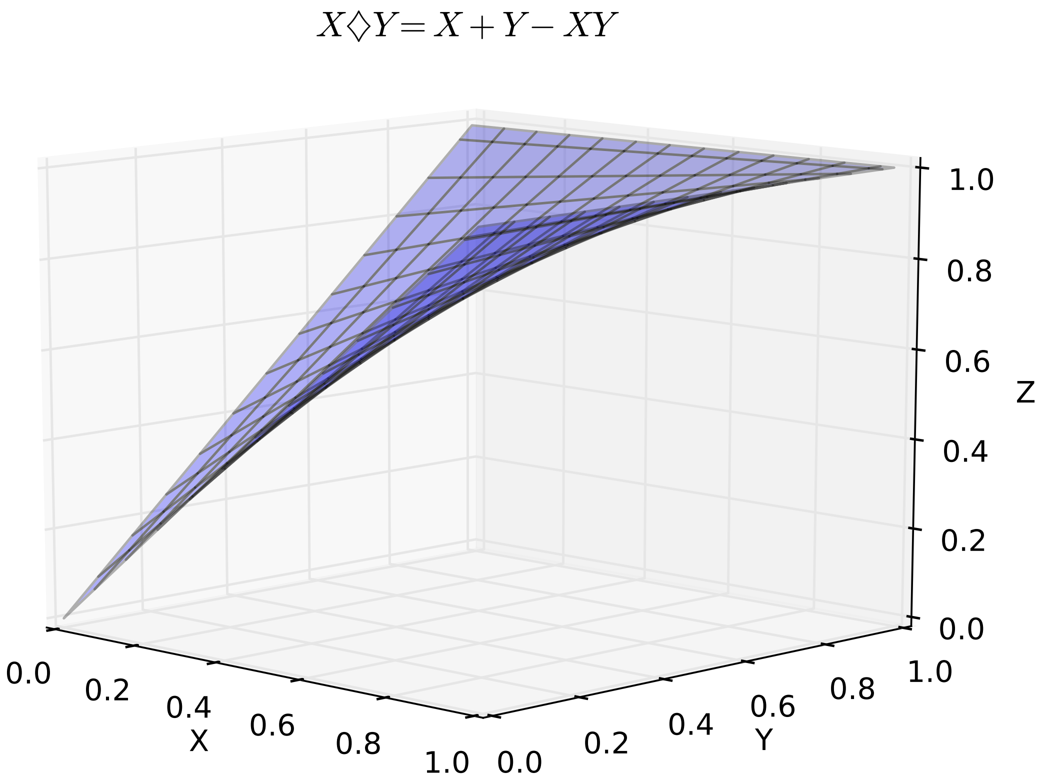

FIG. 6: Function view of the special Type 1 aggregator

aggregator is Type 1 and its offset effect on exponential time-discount.

$$
\begin{aligned}
X_1(t) &= 0.6e^{-\mu(t-r)} \\
X_2(t) &= 0.9e^{-\mu(t-r)} \\
LHS(t) &= \{X_1(r) \Diamond X_2(r)\}e^{-\mu(t-r)} \\
RHS(t) &= X_1(r)e^{-\mu(t-r)} \Diamond X_2(r)e^{-\mu(t-r)}
\end{aligned}
$$

For later reference, we take note of the following useful property of the operator in (4.14), established in [80].

Lemma 4.7.1. *Consider values $X_1, X_2, \cdots, X_n$ in the range $[0,1]$ acted upon by the aggregator $\Diamond$ defined in (4.14). Then the aggregated value $\Diamond_{i=1}^{n} X_i$ satisfies the condition*

$$\Diamond_{i=1}^{n} X_i = 1 - \Pi_{i=1}^{n}(1 - X_i). \tag{4.9}$$

Proof. The proof is by induction. To settle the basis, consider $n = 2$ and assume that

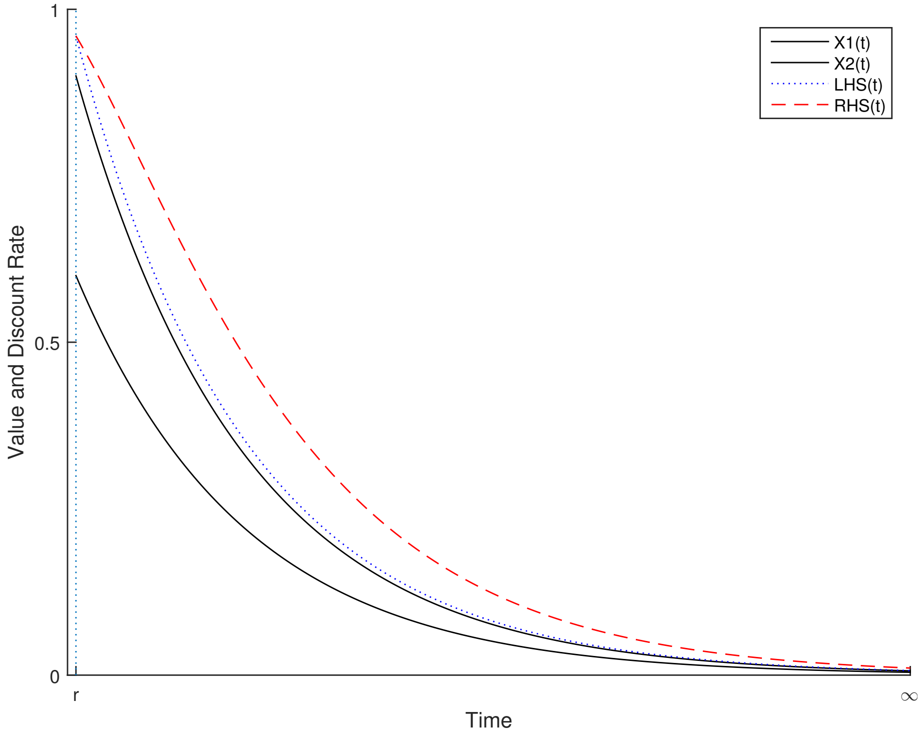

FIG. 7: Value and Discount Rate of the Special Type 1 Aggregator vs Time

both X_1 and X_2 are reals in the interval $[0,1]$. Observe that $1-(X_1 \diamondsuit X_2) = 1-(X_1+X_2-X_1X_2) = (1-X_1)(1-X_2)$.

For the inductive step, let n be arbitrary and assume that the statement of the lemma true for the chosen value of n. With this in hand, we need to show that $1-\diamondsuit_{i=1}^{n+1} X_i = \Pi_{i=1}^{n+1}(1-X_i)$. Write $Y = 1-\Pi_{i=1}^{n}(1-X_i)$. In this notation

$$\begin{aligned}
1-\diamondsuit_{i=1}^{n+1} X_i &= 1-(\diamondsuit_{i=1}^{n} X_i)\diamondsuit X_{n+1} \\
&= 1-Y\diamondsuit X_{n+1} \\
&= 1-Y-X_{n+1}+YX_{n+1} \\
&= \Pi_{i=1}^{n}(1-X_i)-X_{n+1}\Pi_{i=1}^{n}(1-X_i) \\
&= \Pi_{i=1}^{n+1}(1-X_i)
\end{aligned}$$

and the proof of the lemma is complete. □

As a generalization, this special class of Type 1 aggregation operators can be generalized as:

$$X(t)\diamondsuit Y(t) = X(t) + Y(t) - \alpha X(t)Y(t) \tag{4.10}$$

where α is a real number and $\alpha \in [0,2]$.

In particular, when $\alpha = 2$, we have

$$X(t)\diamondsuit Y(t) = X(t) + Y(t) - 2X(t)Y(t) \tag{4.11}$$

When $\alpha = 1$, we get our previous type.

A detailed study on this generalized aggregator will be scheduled in the future.

4.8 AGGREGATOR OF TYPE 2

For a Type 2 aggregation operator defined in 4.4, the order between aggregation and discount does not matter, however , we need to aggregate as soon as possible because the value will decay quickly in reality.

With the same notations, assumptions and reasoning, we have similar *lemma* and *theorem* for Type 2 aggregation operator as for Type 1.

Lemma 4.8.1. *Assume an associative Type 2 operator $\diamondsuit$. For all t, τ with $\max_{1\le i\le n}\{t_i\} \le t \le \tau$ we have*

$$\left[\diamondsuit_{i=1}^{n} X_i(t)\right] \delta(\tau - t) = \diamondsuit_{i=1}^{n} X_i(\tau).$$

Theorem 4.8.2. *Assuming that the Type 2 aggregation operator $\diamondsuit$ is associative and commutative, the discounted value of the aggregated information at time t is $\diamondsuit_{i=1}^{n} X_i(t)$, regardless of the order in which the values were aggregated.*

These *lemma* and *theorem* has similar implications as Type 1 aggregation operator and can be proved in a similar way as in section 4.6.

For exponentially time-discounted information, observe that since $t \ge r$ and $t \ge s$, what is being aggregated at time t are the discounted values $X(r)\cdot\delta(t-r)$ and $Y(s)\cdot\delta(t-s)$ as illustrated in Figure 8.

In order to be able to understand how time discounting affects aggregated values we shall find it convenient to assume that the discount operator — multiplication '·' distributes over the aggregation operator $\diamondsuit$.

Distributivity: For all t, τ with $0 \le t \le \tau$, we can write $[X(t)\diamondsuit Y(t)]\cdot\delta(\tau - t) =$

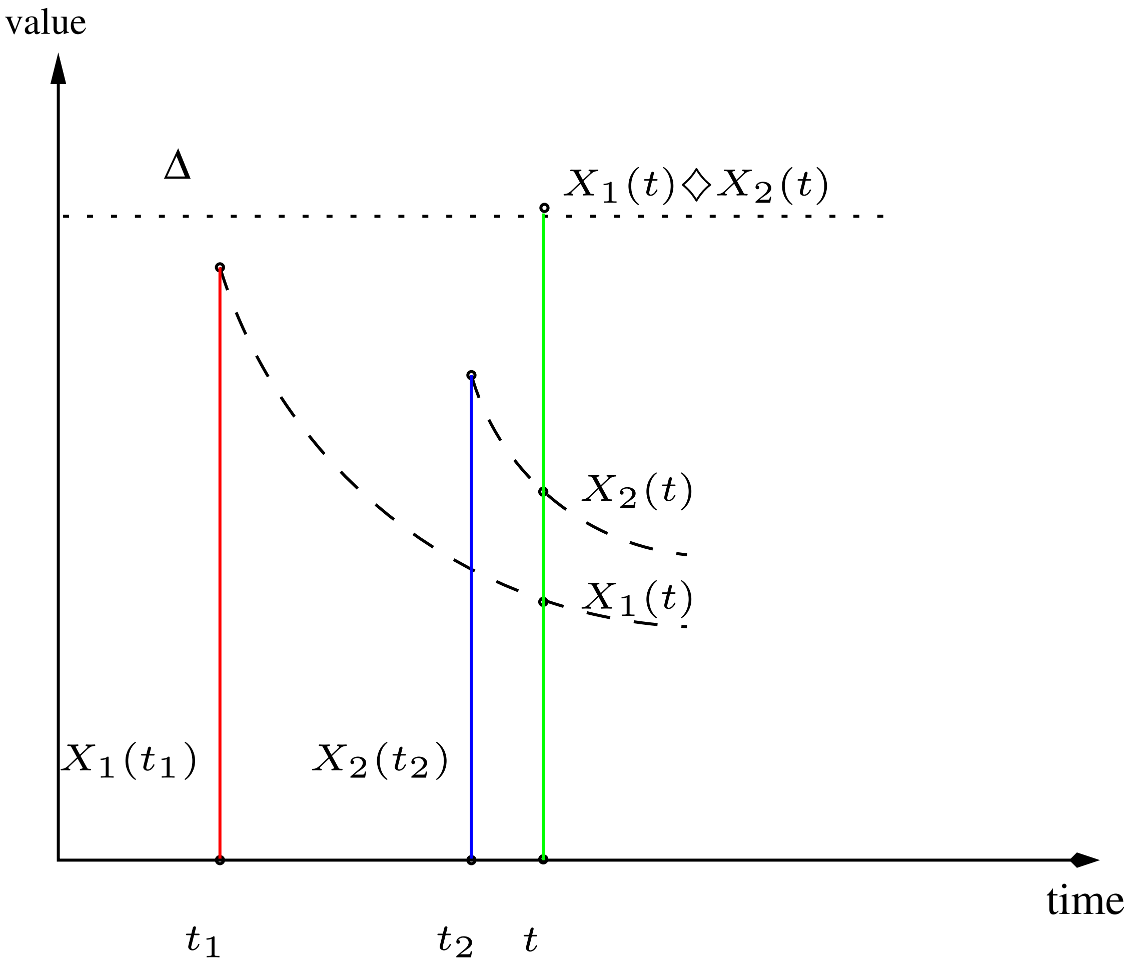

FIG. 8: *Illustrating the aggregation at time t of $X(r)$ and $Y(s)$.*

$[X(t)\cdot\delta(\tau-t)]\diamondsuit[Y(t)\cdot\delta(\tau-t)]$. The discounted value at time $\tau-t$ of the information $X(t)\diamondsuit Y(t)$ aggregated at time t matches the aggregated value at time τ of $X(t)\cdot\delta(\tau-t)$ and $Y(t)\cdot\delta(\tau-t)$. In other words, it does not matter whether we first aggregate and then discount the aggregated information or vice versa.

The distributivity property is fundamental in understanding the interplay between time discounting and aggregation. We mention in passing that, in general the distributivity property need not be verified. However, for exponentially time-discounted information, we look specifically at aggregation operators where distributivity holds.

Lemma 4.8.3. *Assuming the distributivity property, for all $0 \le r \le s \le t \le \tau$, we have*

$$[X(t)\diamondsuit Y(t)]\cdot\delta(\tau-t) = [X(r)\cdot\delta(\tau-r)]\diamondsuit[Y(s)\cdot\delta(\tau-s)]. \tag{4.12}$$

Proof. By using distributivity we write

$$\begin{aligned}
& [X(t)\Diamond Y(t)]\cdot\delta(\tau-t) \\
= \;& [X(t)\cdot\delta(t-r)\Diamond Y(t)\cdot\delta(t-s)]\cdot\delta(\tau-t) \text{ [by (3.4)]} \\
= \;& [X(t)\cdot\delta(t-r)\cdot\delta(\tau-t)]\Diamond[Y(t)\cdot\delta(t-s)\cdot\delta(\tau-t)] \\
= \;& X(r)\cdot\delta(\tau-r)\Diamond Y(t)\cdot\delta(\tau-s) \text{ [by Lemma 3.2.3]}
\end{aligned}$$

□

The left-hand side of (4.12) is the discounted value of $X(t)\Diamond Y(t)$ at time τ, while the right-hand is the aggregated value of the discounted values of $X(r)$ and $Y(s)$ at time τ.

4.9 AGGREGATOR OF TYPE 3

Since Type 3 operators and the aggregation strategies they suggest, are duel forms of Type 1, whose properties will be given here for completeness without proof. All the symbols has the same meanings as in section 4.6.

Lemma 4.9.1. *Assume an associative Type 3 operator $\Diamond$. For all t, τ with $\max_{1\le i\le n}\{t_i\}\le t\le\tau$ we have*

$$\left[\Diamond_{i=1}^{n}X_i(t)\right]\delta(\tau-t) > \Diamond_{i=1}^{n}X_i(\tau).$$

Theorem 4.9.2. *Assuming that the Type 3 aggregation operator $\Diamond$ is associative and commutative, the discounted value of the aggregated information at time t is lower-bounded by $\Diamond_{i=1}^{n}X_i(t)$, regardless of the order in which the values were aggregated.*

4.10 EXPONENTIALLY-DISCOUNTED VALUE OF AGGREGATED INFORMATION

Consider an event witnessed by n, $(n\ge 2)$, sensors and let the sensed values be $X_1, X_2, \cdots, X_n$, collected, respectively, at times $t_1, t_2, \cdots, t_n$. Assume, further, that various groups of sensors have aggregated their information before time t and that, finally, at time t the aggregation has been completed. We are interested in evaluating the time-discounted value of the information collected by the sensors at time t, where $t\ge\max\{t_1,t_2,\cdots,t_n\}$. The answer to this natural question is provided by the following fundamental result.

Theorem 4.10.1. *Assuming distributivity of discount operator $\cdot$ over the aggregation operator $\diamondsuit$, the discounted value $V(t)$ of the aggregated information at time t is*

$$V(t) = \diamondsuit_{i=1}^{n} X_i(t_i) \cdot \delta(t - t_i), \tag{4.13}$$

regardless of the order in which the values were aggregated.

Proof. The proof is by induction on n. For $n = 2$, the conclusion follows at once from Lemma 4.8.3. Now, let $n \geq 2$, be arbitrary and assume the statement true for all m, $(m < n)$. We assume, without loss of generality, that the last aggregation takes place at time t. This aggregation must have involved a number of disjoint groups $G_1, G_2, \ldots, G_p$ each of them the result of a previous aggregation at times, respectively, $u_1,\ u_2,\ \ldots,\ u_p$. Observe that we can always relabel the groups in such a way that their aggregation times are ordered as $u_1 < u_2 < \cdots < u_p$.

Let us look at group G_k. By the induction hypothesis, the value of information in group G_k aggregated at time u_k was

$$V(u_k) = \diamondsuit_{j=1}^{n_k} X_{k_j} \cdot \delta(u_k - t_{k_j})$$

where, of course, we assume that group G_k involves n_k sensors whose values were aggregated.

Assuming $t \geq u_k$, the discounted value of $V(u_k)$ at time t is

$$\begin{aligned} V_k(t) &= \left[\diamondsuit_{j=1}^{n_k} X_{k_j} \cdot \delta(u_k - t_{k_j})\right] \cdot \delta(t - u_k) \\ &= \diamondsuit_{j=1}^{n_k} X_{k_j} \cdot \delta(u_k - t_{k_j}) \cdot \delta(t - u_k) \ \text{[by distributivity]} \\ &= \diamondsuit_{j=1}^{n_k} X_{k_j} \cdot \delta(t - t_{k_j}) \ \text{[by Lemma 3.2.3].} \end{aligned}$$

which is exactly the discounted value of information collected by sensors in group G_k, had it been aggregated at time t. Since G_k was arbitrary, the conclusion follows. □

Theorem 4.10.1, in effect, says that the order in which the values are aggregated does not matter as long as each is aggregated only once. In practical terms, Theorem 4.10.1 gives the algorithm designer the freedom to schedule aggregation in a random manner, much in line with the stochastic nature of wireless communication and sensor data aggregation.

4.11 SOME SPECIAL CLASSES OF TYPE 2 AGGREGATORS

There are some special classes of Type 2 aggregation operators such as $+, \min, \max$ that turns out to have wide applications. Assume that the data collected by sensors take on values in $\mathbb{R}$, these aggregation operators $\diamondsuit$ can be defined as

$$\begin{aligned} +: &\quad X(t)\diamondsuit Y(t) = X(t)+Y(t) \\ \max: &\quad X(t)\diamondsuit Y(t) = \max\{X(t),Y(t)\} \\ \min: &\quad X(t)\diamondsuit Y(t) = \min\{X(t),Y(t)\} \end{aligned}$$

It is easy to verify that $\diamondsuit$ satisfies the associativity and commutativity properties defined above. However, for idempotency, $+$ and max satisfy in common meaning; min will satisfy by redefining $-\infty$ as value zero. To prove that these $\diamondsuit$s are Type 2 operator, as they can be proved in similar way, let's only prove $+$ here. Consider an arbitrary τ with $0 \leq t \leq \tau$ and write

$$\begin{aligned} &[X(t)\diamondsuit Y(t)]\,\delta(\tau-t) \\ &= [X(t)+Y(t)]\,\delta(\tau-t) \\ &= X(t)\delta(\tau-t)+Y(t)\delta(\tau-t) \\ &= X(\tau)+Y(\tau) \;\; \text{[by (3.4)]} \\ &= X(\tau)\diamondsuit Y(\tau) \end{aligned}$$

confirming that the aggregation operator $+$ defined in (4.14) is of Type 2.

4.12 A SPECIAL CLASS OF TYPE 4 AGGREGATORS

A special class of Type 4 aggregators arise from the following real-life scenario: Suppose there is an intruder trespassing an area monitored by a group of sensors. Take two sensors as an example, sensor a detected it as event A with probability $P(A)$, b detected it as event B with $P(B)$. The pair-wise consensus between these two sensors is illustrated as the union of those shaded areas in Figure 9. The orange part means the features detected by both of them, and the green part means their common undetected features. So it is reasonable to define the union of their common detected and undetected features as their consensus on the same event.

Definition 4.12.1. *Suppose the sample space is U, for an identical phenomenon, sensor a detected it as event A with probability $P(A)$, b detected it as event B with $P(B)$. The*

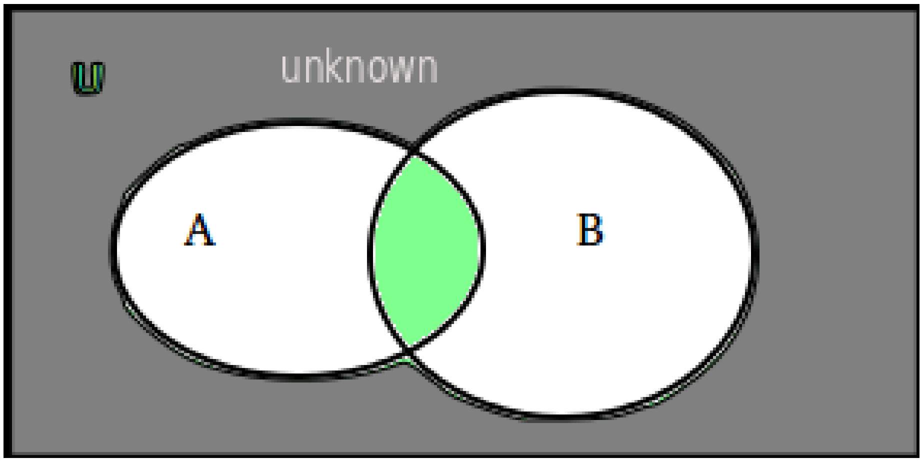

FIG. 9: *Pair-wise consensus between two sensors.*

consensus *of both sensor on that phenomenon is defined as the probability of* $U \setminus ((A \cup B) \setminus (A \cap B))$ *which is the shaded area in Figure 9. We denote it as an aggregation operator* $\diamondsuit$*:*

$$P(A) \diamondsuit P(B) = P(U \setminus ((A \cup B) \setminus (A \cap B)))$$

Let $a = P(A), b = P(B)$, events A and B are independent as sensors detected the phenomenon independently, then

$$\begin{aligned}
& P(A) \diamondsuit P(B) = a \diamondsuit b \\
& = P(U \setminus ((A \cup B) \setminus (A \cap B))) \\
& = P(U) - P((A \cup B) \setminus (A \cap B)) \\
& = 1 - P((A \cup B) \setminus (A \cap B)) \\
& = 1 - (P(A \cup B) - P(A \cap B)) \\
& = 1 - (P(A) + P(B) - P(A)P(B) - P(A)P(B)) \\
& = 1 - (a + b - ab - ab) \\
& = ab + (1-a)(1-b) \qquad (4.14)
\end{aligned}$$

From (4.14), when both $P(A)$ and $P(B)$ equal 1, they reach the consensus that the phenomenon happened; when both $P(A)$ and $P(B)$ equal 0, they reach the consensus that there the phenomenon does NOT occur; in these two cases, they reach the highest consensus 1. When $P(A)$ and $P(B)$, one is 0, the other is 1, which means they are contrary on the same event, where they have no consensus, the value is 0. For other values of $(P(A), P(B)$, the consensus is in interval $(0,1)$. Which fits our intuition.

In real application, such consensus will give us a kind of reliability. In a group of deployed sensors, one sensor detected an emergency with a probability above a preset threshold, then, it seeks supporters from its neighbors by consensus. If its sum of pair-wise consensus is also above a preset threshold, then it can trigger an alarm that there is an emergency.

4.12.1 ALGEBRAIC PROPERTIES OF THE PAIR-WISE CONSENSUS MODEL

From the view of function, $a\diamondsuit b = ab + (1-a)(1-b) = 1-a-b+2ab = f(a,b)$, it can be visualized as Figure 10:

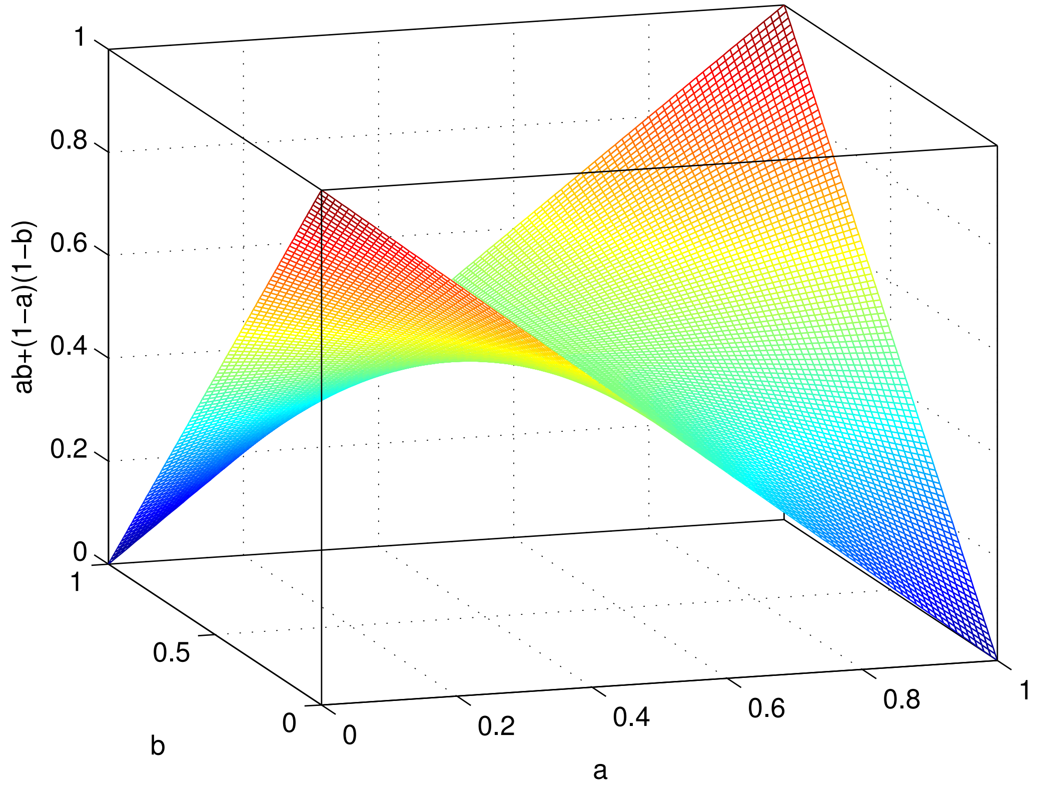

FIG. 10: *Surface of $f(a,b) = 1-a-b+2ab$*

Theorem 4.12.1. *This aggregation operator satisfies: closure on $[0,1]$ and commutativity.*

Proof of closure: We need to prove $a\diamondsuit b \in [0,1], \quad \forall a,b \in [0,1]$. It is clear that $a\diamondsuit b =$

$ab + (1-a)(1-b) \geq 0$. For $a \diamondsuit b \leq 1$:

$$\begin{aligned}
&ab \leq \sqrt{ab} \leq \frac{a+b}{2} \quad \forall a, b \in [0,1] \\
&\Rightarrow ab \leq \frac{a+b}{2} \\
&\Rightarrow 2ab \leq a+b \\
&\Rightarrow ab + 1 - a - b + ab \leq 1 \\
&\Rightarrow ab + (1-a)(1-b) \leq 1 \\
&\Rightarrow a \diamondsuit b \leq 1
\end{aligned}$$

□

Closure implies that when it runs from 0 to 1, the consensus runs from disagreement to ubiquitous agreement.

Commutativity is obvious. Commutativity implies that the result of consensus does not depend on the order of aggregation.

Denote set $[0,1]$ as G, together with binary operator $\diamondsuit$ as $(G, \diamondsuit)$. We can find a unique **Identity** element for the algebraic structure and an **Inverse** element for each element in G.

Unique identity element existence:

$$\begin{aligned}
&a \diamondsuit x = x \diamondsuit a = a \\
&\Rightarrow 1 - a - x + 2ax = a \\
&\Rightarrow (2a-1)x = 2a - 1 \\
&\Rightarrow x = 1 \quad \text{when } a \neq \frac{1}{2}
\end{aligned}$$

So the identity element e is 1 for $(G, \diamondsuit)$. Its uniqueness is easy to verify and omitted here.

$\frac{1}{2}$ is the `zero element` of $(G, \diamondsuit)$ because:

$$\begin{aligned}
&a \diamondsuit \frac{1}{2} = \frac{1}{2} \diamondsuit a \\
&= 1 - a - \frac{1}{2} + 2a\frac{1}{2} = \frac{1}{2}
\end{aligned}$$

The inverse element for an arbitrary element a except zero element in G:

$$\begin{aligned}
&a\diamondsuit x = x\diamondsuit a = 1\\
&\Rightarrow 1-a-x+2ax = 1\\
&\Rightarrow (2a-1)x = a\\
&\Rightarrow x = \frac{a}{2a-1} \quad \text{when } a \neq \frac{1}{2}
\end{aligned}$$

So the inverse element a^{-1} of a exists when $a \neq \frac{1}{2}$.

Type determination: The type of this aggregation operator $\diamondsuit$ can be determined by comparing the following two expressions:

$$(a\diamondsuit b)\delta \sim (a\delta\diamondsuit b\delta)$$

$$\begin{aligned}
LHS &= (a\diamondsuit b)\delta\\
&= (1-a-b+2ab)\delta\\
&= \delta - a\delta - b\delta + 2ab\delta\\
RHS &= (a\delta\diamondsuit b\delta)\\
&= 1 - a\delta - b\delta + 2ab\delta^2\\
LHS - RHS &= \delta - 1 + 2ab\delta(1-\delta)\\
&= (1-\delta)(2ab\delta - 1)
\end{aligned}$$

When there is no discount at the beginning, i.e. $\delta = 1$, $LHS = RHS$. When there is discount, i.e. $0 < \delta < 1$, then $1-\delta > 0$, so the type of this $\diamondsuit$ is dependent on the sign of $2ab\delta - 1$. δ behaves as a keystone here:

$\delta > \frac{1}{2}$: In this, each curve $2ab\delta - 1 = 0$ determined by a specified δ divides the unit rectangle demarcated by $a = 0, b = 0, a = 1, b = 1$ into three parts, shown in Figure 11:

- Below the curve: $LHS < RHS$, $\diamondsuit$ defined above behave as Type 1 aggregation operator for $\{(a,b)|2ab\delta - 1 < 0, \forall a,b \in J\}$.
- On the curve: $LHS = RHS$, $\diamondsuit$ defined above behave as Type 2 aggregation operator for $\{(a,b)|2ab\delta - 1 = 0, \forall a,b \in J\}$.

- Above the curve: $LHS > RHS$, $\diamondsuit$ defined above behave as Type 3 aggregation operator for $\{(a,b)|2ab\delta - 1 > 0, \forall a,b \in J\}$. From Figure 11, it is clear that $a > 1/2\delta > 1/2$ and $b > 1/2\delta > 1/2$.

$\delta = \frac{1}{2}$**:** $2ab\delta - 1 = ab - 1 = 0$. This equation has only one solution under the constraint that $a \in [0,1]$ and $b \in [0,1]$, which is $a = 1$ and $b = 1$, i.e. at point $(1,1)$ the $\diamondsuit$ defined above behave as Type 2 aggregation operator and as Type 1 for $\{(a,b)|\forall a,b \in J \setminus \{1\}\}$.

$\delta < \frac{1}{2}$**:** $2ab\delta - 1 < ab - 1 \leq 0$, the $\diamondsuit$ defined above behave as Type 1 aggregation operator for $\{(a,b)|\forall a,b \in J\}$.

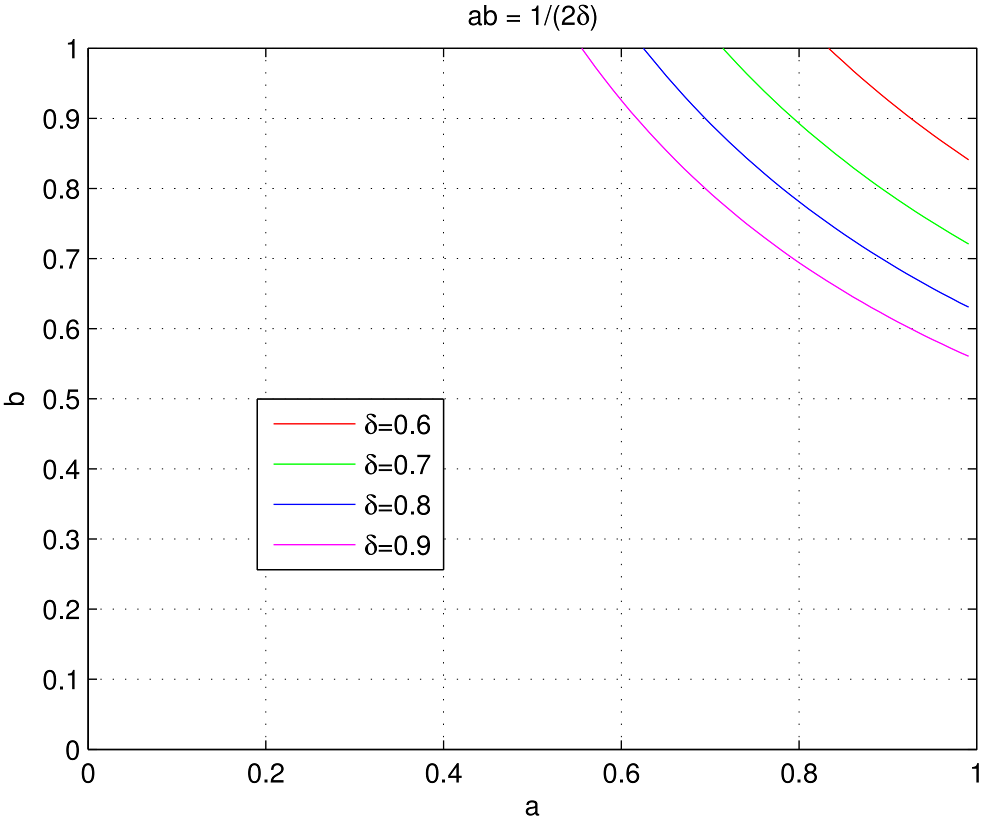

FIG. 11: *Family of curves of* $2ab\delta - 1 = 0$

$\delta = e^{-\mu(t-t_0)}, \quad t \geq t_0$, decrease with time, the curve $2ab\delta - 1 = 0$ moves away from the origin. The aggregation operator defined above is a seamless composition of all three aggregation operator types we have found in our previous work. It switches from one to another automatically when time goes by. Let $\tau = 1/\mu$ as the time constant of deterioration of value, and $t_{1/2} = \tau \ln 2, t_0 = 0$, then $\delta = e^{-\mu(t-t_0)} = e^{-\ln 2} = \frac{1}{2}$, the time axis is divided into $[0, t_{1/2}) \cup \{t_{1/2}\} \cup (t_{1/2}, +\infty)$. So $t_{1/2}$ become the key time point for different strategies of consensus.

Suppose event E occur with probability $P(E)$, then the event that E does not occur, i.e. $\overline{E}$ has probability $P(\overline{E}) = 1 - P(E)$. When they occur with the same probability, which is 1/2, exactly the *zero element* in $(G, \diamondsuit)$, we define this value as *zero confidence*, that is, both events have no confidence that their corresponding phenomenons will occur. As the probability of 1/2 has identical confidence on two opposite directions, such confidence means no confidence, then probability with value less than 1/2 can be interpreted as negative confidence about an event and those larger than 1/2 as positive confidence.

So the automatic transition of the aggregation type of consensus shows that: when the value decreased to no more than half its original value, which is also no more than 1/2, the type turns into only one possible type — Type 1, this implies it is best to defer aggregation as long as the semantics of the application permit when they have only negative confidence; when the value is fresh which does not decrease to half its original value and large enough for those values above the curve $2ab\delta - 1 = 0$ i.e. positive confidence, the best strategy is to aggregate as early as the data is available and/or the semantics of the application permit. This scenario testifies the old saying — Grab the chance as soon as possible when it is available otherwise wait for its availability.

4.13 A GENERALIZED SPECIAL CLASSES OF AGGREGATORS

The special class of Type 1 aggregation operators explained in Section 4.7

$$X \diamondsuit Y = X + Y - 1 \times XY$$

and special class of Type 2 — '+' in Section 4.11

$$X \diamondsuit Y = X + Y$$

can be generalized as

$$X \diamondsuit Y = X + Y - \alpha XY,$$

where α is a real number. When $\alpha = 1$, it turns into Type 1 aggregation operator; when $\alpha = 0$, it turns into Type 2 aggregation operator.

The type of this generalized aggregation operator $\diamondsuit$ can be determined by comparing the following two expressions:

$$(X\diamondsuit Y)\delta \sim (X\delta\diamondsuit Y\delta)$$

where $\delta = \delta(t) = e^{-\mu(t-t_0)}$.

$$\begin{aligned} LHS &= (X\diamondsuit Y)\delta \\ &= (X+Y-\alpha XY)\delta \\ &= X\delta + Y\delta - \alpha XY\delta \\ RHS &= (X\delta\diamondsuit Y\delta) \\ &= X\delta + Y\delta - \alpha XY\delta^2 \\ LHS - RHS &= \alpha XY\delta(\delta - 1) \end{aligned} \tag{4.15}$$

Disregard the trivial case $\delta(0) = 1$, then $\delta \in (0,1)$, so $\delta - 1 < 0$ and the sign of equation (4.15) is only determined by α.

- When $\alpha = 0$, then $LHS = RHS$, the aggregator $\diamondsuit$ is Type 2.
- When $\alpha > 0$, then $LHS < RHS$, the aggregator $\diamondsuit$ is Type 1.
- When $\alpha < 0$, then $LHS = RHS$, the aggregator $\diamondsuit$ is Type 3.

In a functional view with a single parameter,

$$f(X,Y) = X\diamondsuit Y = X + Y - \alpha XY \tag{4.16}$$

Three figures with $\alpha = 1, 0, -1$ are shown in Figure 12, Figure 13 and Figure 14 respectively. From these figures we can see,

- When $\alpha = 0$, equation (4.16) describes a Type 2 aggregator, the figure is a plane, which means the aggregation is monotonically increasing along the positive direction of both X axis and Y axis.
- When $\alpha = 1$, equation (4.16) describes a Type 1 aggregator, the figure is a saddle, its ridge sits along the plane $X = -Y$;
- When $\alpha = -1$, equation (4.16) describes a Type 3 aggregator, the figure is also a saddle, its ridge sits along the plane $X = Y$. This figure staggers with Figure 12 by

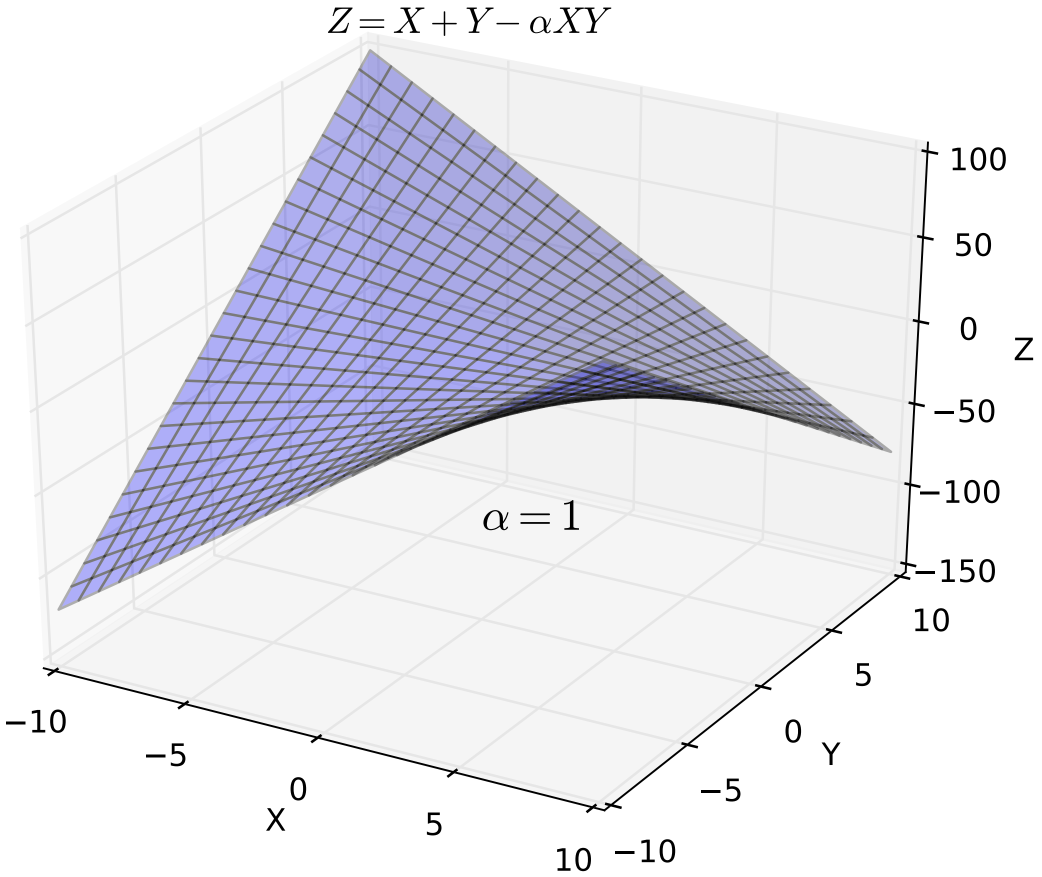

FIG. 12: Functional view of the aggregator $X \Diamond Y = X + Y - \alpha XY$ with $\alpha = 1$

90 degree. Which means Type 1 aggregator has opposite aggregation effect to Type 3 on exponentially time-discounted values.

The meaning and implication of this generalization require further exploration, especially when α is other than 0 and 1 which have been studied.

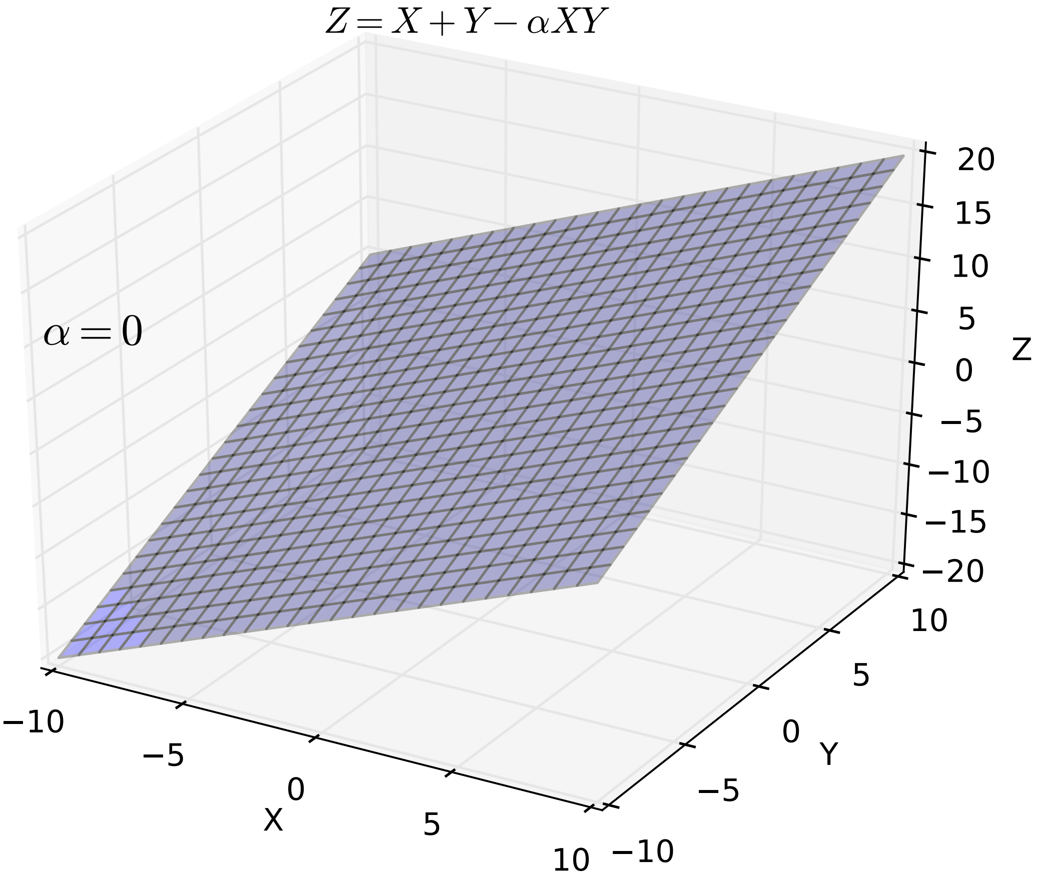

FIG. 13: Functional view of the aggregator $X \diamondsuit Y = X + Y - \alpha XY$ with $\alpha = 0$

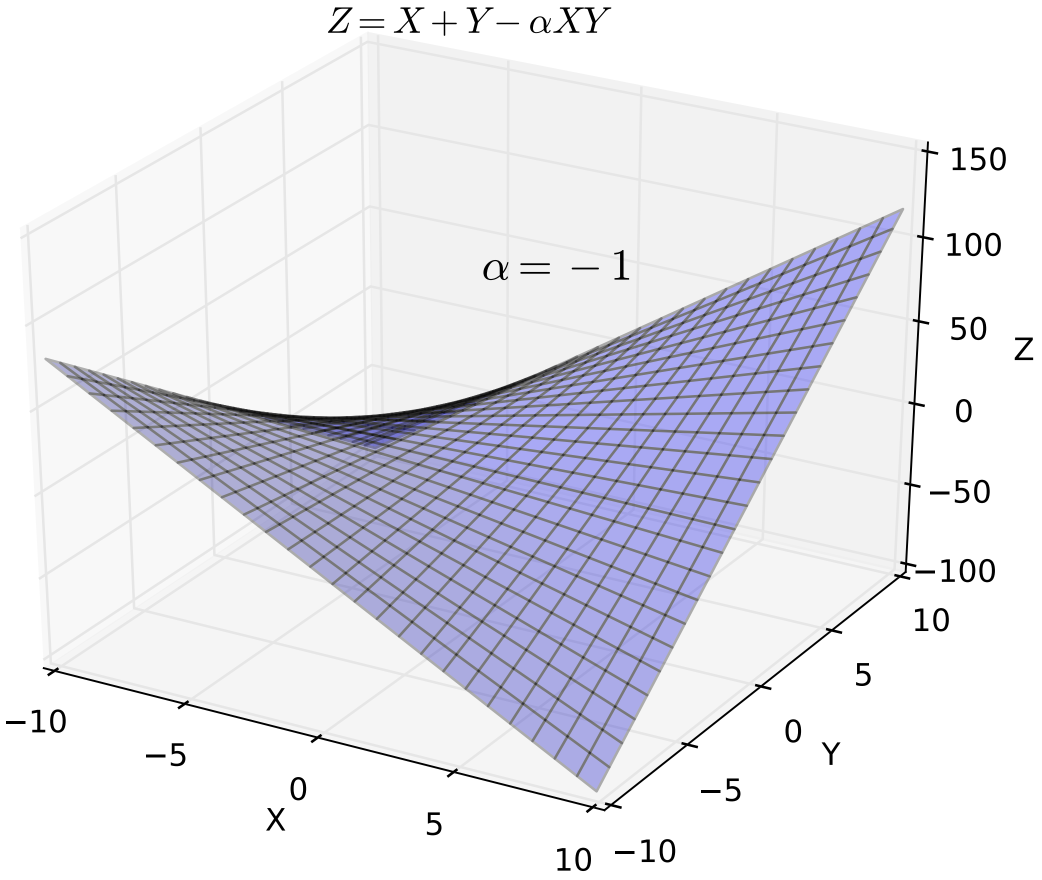

FIG. 14: Functional view of the aggregator $X \diamondsuit Y = X + Y - \alpha XY$ with $\alpha = -1$

CHAPTER 5

APPLICATIONS OF DATA AGGREGATION

5.1 APPLICATION OF TYPE 1 OPERATORS

5.1.1 THRESHOLDING

Sensor networks deployed in support of emergency response applications must provide timely and accurate reports of detected events. Aggregation of sensor data is required to accomplish this in an efficient manner.

The aggregation problem is complicated by the fact that the perceived value of the data collected by the sensors deteriorates, often dramatically, over time. Individual sensors must determine whether to report a perceived event immediately or to defer reporting until the confidence has increased after aggregating data with neighboring nodes. However, aggregation takes time and the longer the sensors wait, the lower the value of the aggregated information.

As already mentioned, we assume that reporting a false positive involves a huge overhead and is considered prohibitively expensive. Mindful of this state of affairs, and having aggregated, at time t, the information collected by the various sensors, it is important to decide whether this information warrants reporting.

5.1.2 A FIXED AGGREGATION STRATEGY

One of the natural strategies employed is thresholding. Specifically, a policy is followed of first setting up an application-dependent threshold Δ and then reporting an event only if the aggregated information exceeds Δ.

Assume that n, $(n \geq 2)$, sensors have collected data about an event at times $t_1,\ t_2,\ \cdots,\ t_n$ and let $t = \max\{t_1,\ t_2,\ \cdots,\ t_n\}$. Further, let $X_1(t_1), X_2(t_2), \cdots, X_n(t_n)$ be the values of the data collected by the sensors. Assuming that $\diamondsuit_{i=1}^{n} X_i(t) > \Delta$ the time at which the aggregation is performed is critical. We have seen that for a Type 1 operator aggregation may be delayed as long as possible. It is, however, intuitively clear that if aggregation is delayed too much, the aggregated value might not exceed the threshold and a relevant event would go unreported. Thus, the question is to determine the *time window* during which the sensors need to aggregate their values in order for the aggregated value to exceed the threshold Δ.

Let τ be the *latest* time at which aggregation should be performed. Since we are interested in Type 1 operators, it is natural to insist that $\left[\Diamond_{i=1}^{n} X_i(t)\right] \delta(\tau - t) > \Delta$. Recalling that by, Theorem 3.2.7, $\delta(\tau - t) = e^{-\mu(\tau - t)}$, the above inequality becomes $e^{-\mu(\tau-t)} > \frac{\Delta}{\Diamond_{i=1}^{n} X_i(t)}$ which, upon taking logarithms on both sides, yields

$$-\mu(\tau - t) > \ln \frac{\Delta}{\Diamond_{i=1}^{n} X_i(t)}.$$

Upon solving for τ we obtain

$$\tau < t + \frac{1}{\mu} \ln \frac{\Diamond_{i=1}^{n} X_i(t)}{\Delta}. \tag{5.1}$$

As the last data was collected at time $t = \max\{t_1,\ t_2,\ \cdots,\ t_n\}$, equation (5.1) specifies that, past t, there is a time window of size $\frac{1}{\mu} \ln \frac{\Diamond_{i=1}^{n} X_i(t)}{\Delta}$ during which aggregation must occur. This result gives the system designer a handle on the types of aggregation protocols to use. In equation (5.1), in order to get a certain length W of time window, i.e., the following inequality needs to be satisfied:

$$W \leq \frac{1}{\mu} \ln \frac{\Diamond_{i=1}^{n} X_i(t)}{\Delta} \tag{5.2}$$

By equation (3.14), W is proportional to the mean life-time $\frac{1}{\mu}$ of values $X_i(t)$. As well, the following inequality should satisfy in order to make sense for inequalities (5.1) and (5.2).

$$\Diamond_{i=1}^{n} X_i(t) > \Delta \tag{5.3}$$

Inequality (5.3) fits intuition, i.e., the aggregated value should be larger than the alarm threshold in order to trigger an alarm, and larger enough to have a time window for emergency response.

From Lemma 4.7.1, it is easy to get an estimation of the length of time window in fixed strategy. Suppose $X_1, X_2, \cdots, X_n$ are all 0.9, then $1 - X_i = 0.1 = 10^{-1}$, substitute into equation (4.9),

$$\begin{aligned}\Diamond_{i=1}^{n} X_i &= 1 - \Pi_{i=1}^{n}(1 - X_i) \\ &= 1 - \Pi_{i=1}^{n} 0.1 \\ &= 1 - 10^{-n} \\ &= 0.\underbrace{9 \cdots 9}_{n\ nines}\end{aligned}$$

Substitute back into inequality (5.2) and take equal sign, get

$$W = \frac{1}{\mu} \ln \frac{1 - 10^{-n}}{\Delta} \tag{5.4}$$

In order to get an intuitive overview of equation (5.4), we take the number of sensors for aggregation from 2 to 7, and calculate the value of μW — the ratio of the length of time window to the mean life-time of values, put into Table (2).

TABLE 2: Typical aggregated values by special aggregator Type 1 in fixed aggregation strategy

n **number of values**	2	3	4	5	6	7
$\Diamond_{i=1}^{n} X_i$	0.99	0.999	0.9999	0.99999	0.999999	0.9999999
$\mu W = \ln \frac{\Diamond_{i=1}^{n} X_i(t)}{\Delta}$	0	0.009	0.0099	0.01004	0.010049	0.01005

Where $\Delta = 0.99$.

From Table (2) we can see μW, the ratio of the length of time window to the mean life-time of value, is relatively low, only about 1%. if the mean life-time of value is $10s$, then the window length is only $0.1s$ which is almost impossible for human being's reaction. Alerting in so short time should resort to automatic systems. Even worse, μW grows very slowly with the number of sensors for aggregation which is shown in the table when number of sensors runs from 3 to 7, in other words, increasing the number of sensors in such situation does not help.

Treat W in equation (5.4) as a function of n, shown in Figure 15. Actually,

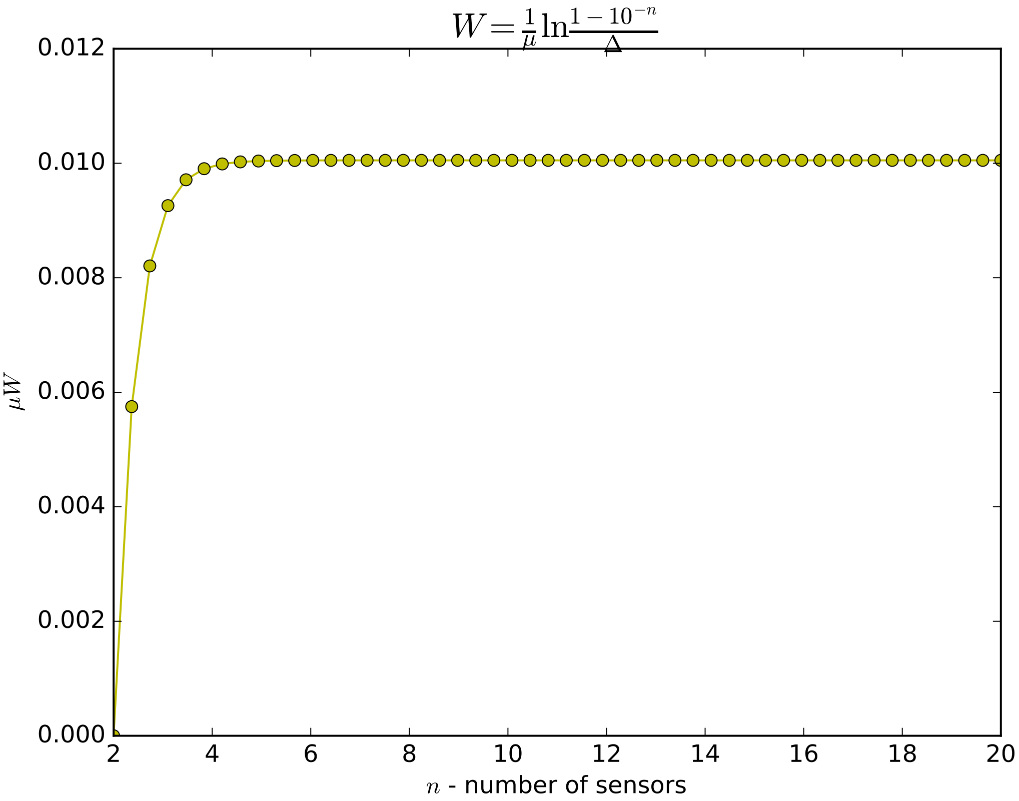

FIG. 15: *Ratio of the length of time window to the mean life-time of value vs number of sensors in fixed aggregation strategy*

$$\begin{aligned}
\lim_{n\to\infty} W(n) &= \lim_{n\to\infty} \frac{1}{\mu} \ln \frac{1-10^{-n}}{\Delta} \\
&= \frac{1}{\mu} \ln \frac{1}{\Delta} \\
&= -\frac{1}{\mu} \ln \Delta \\
&\approxeq \frac{0.01005}{\mu} |_{\Delta=0.99} \\
&\approxeq \frac{0.01}{\mu}
\end{aligned}$$

The limit of W says there is a upper bound, which is roughly 1% of the mean life-time of values when $\Delta = 0.99$ that is a relatively high threshold. The good news is that *three to* *four* sensors suffice to raise the alert provided that each sensor's value is no less than 0.9.

5.1.3 AN ADAPTIVE AGGREGATION STRATEGY

As before, sensor readings about an event were collected and the resulting values $X_1, X_2, \cdots$ are reals in $[0,1]$. Assume that one of the network actors (e.g., a sensor) is in charge of the aggregation process and that the operator $\diamondsuit$ defined in (4.3) is employed in conjunction with a threshold $\Delta > 0$.

We now state and prove a technical result that will motivate our adaptive aggregation strategy.

Theorem 5.1.1. *If* $X_{i_1}, X_{i_2}, \cdots, X_{i_m}$, $m > 1$, *satisfy* $X_{i_j} > 1 - \sqrt[m]{1-\Delta}, \quad j = 1, 2, \cdots, m$, *then* $\diamondsuit_{j=1}^{m} X_{i_j} > \Delta$.

Proof. By Lemma 4.7.1,

$$\begin{aligned}
\diamondsuit_{j=1}^{m} X_{i_j} &= 1 - \Pi_{j=1}^{m} (1 - X_{i_j}) \\
&> 1 - \Pi_{j=1}^{m} \left(1 - (1 - \sqrt[m]{1-\Delta})\right) \\
&= 1 - \Pi_{j=1}^{m} \sqrt[m]{1-\Delta} \\
&= 1 - (1-\Delta) \\
&= \Delta.
\end{aligned}$$

□

Notice what Theorem 5.1.1 says: if there are two sensors whose individual values exceed $1 - \sqrt{1-\Delta}$, then the two should aggregate their values and, having exceeded Δ, should report the event. Similarly, if there are three sensors whose individual values exceed $1 - \sqrt[3]{1-\Delta}$, then the result of their aggregated data exceeds Δ, etc.

In turn, this observation suggests the following *adaptive aggregation* strategy: in the first aggregation round, the aggregator will announce the target $1 - \sqrt{1-\Delta}$. If at least two sensors (including the aggregator) hold values in excess of the target $1 - \sqrt{1-\Delta}$, then the event will be reported. If the first round of aggregation suffices, all is well. If, however, there is an insufficient number of sensors holding suitable values, the second round begins. In this round, the aggregator announces the target $1 - \sqrt[3]{1-\Delta}$. If three or more sensors can be identified that exceed this target, then by Theorem 5.1.1 the aggregated value must exceed Δ and so the event is reported. This aggregation strategy is continued, as described, until either an event is reported, or else the results are inconclusive and no event is reported.

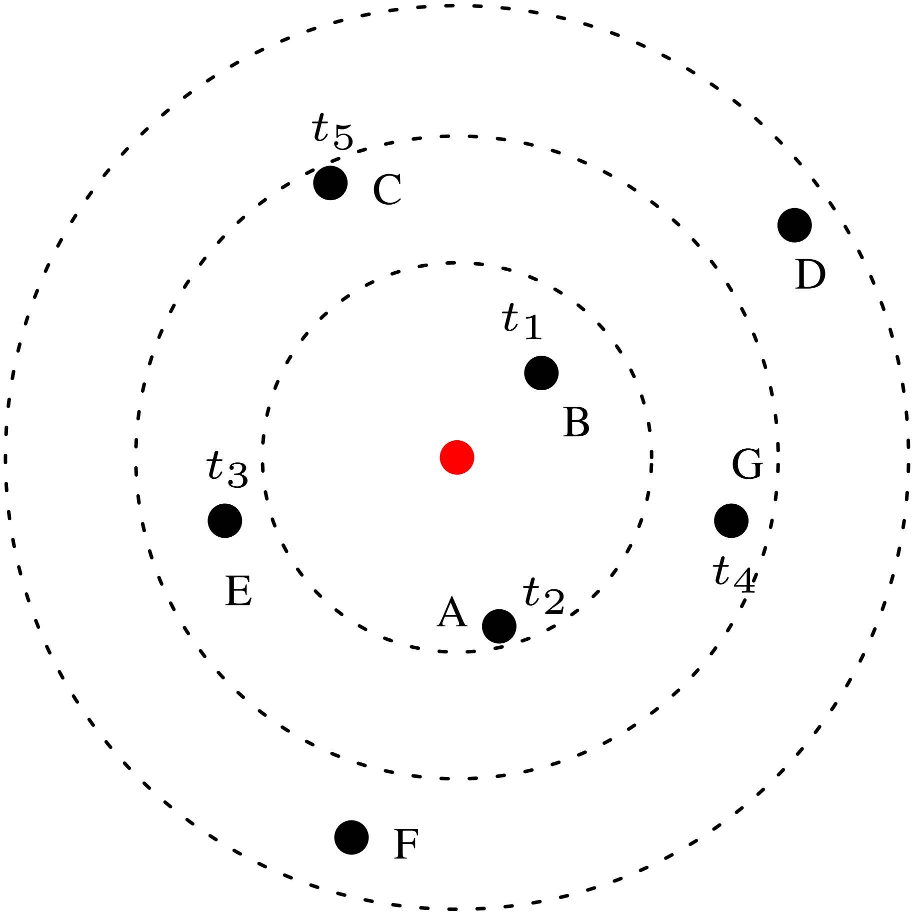

FIG. 16: *Illustrating a fire event.*

For the sake of illustration, and to fix the ideas, consider that a fire just broke out on a ship and refer to Figure 16. There are seven sensors in the area, of which five, namely A, B, C, E and G, are in close proximity of the location of the fire. As the fire spreads, these sensors will detect abnormal temperatures at times $t_1 < t_2 < t_3 < t_4$. Further, let $X_1(t_1), X_2(t_2), X_3(t_3), X_4(t_4)$ and $X_5(t_5)$ be the values thus collected, normalized to $[0,1]$. Given the layout of the sensors, it is reasonable to assume that $X_1(t_1) \geq X_2(t_2) \geq X_3(t_3) \geq X_4(t_4) \geq X_5(t_5)$.[1] Since sensor B is closest to the fire it will be the first one to sense high temperature and, thus, will become an aggregator. B will wait for other sensors to report a reading over $1-\sqrt{1-\Delta}$ and will attempt aggregation with such sensors.

Referring to Figure 17, the first two rounds of our adaptive aggregation strategy do not yield a sufficient number of sensors to effect an aggregation. In the third round, B

[1] This ordering is assumed for illustration purposes but is not really necessary.

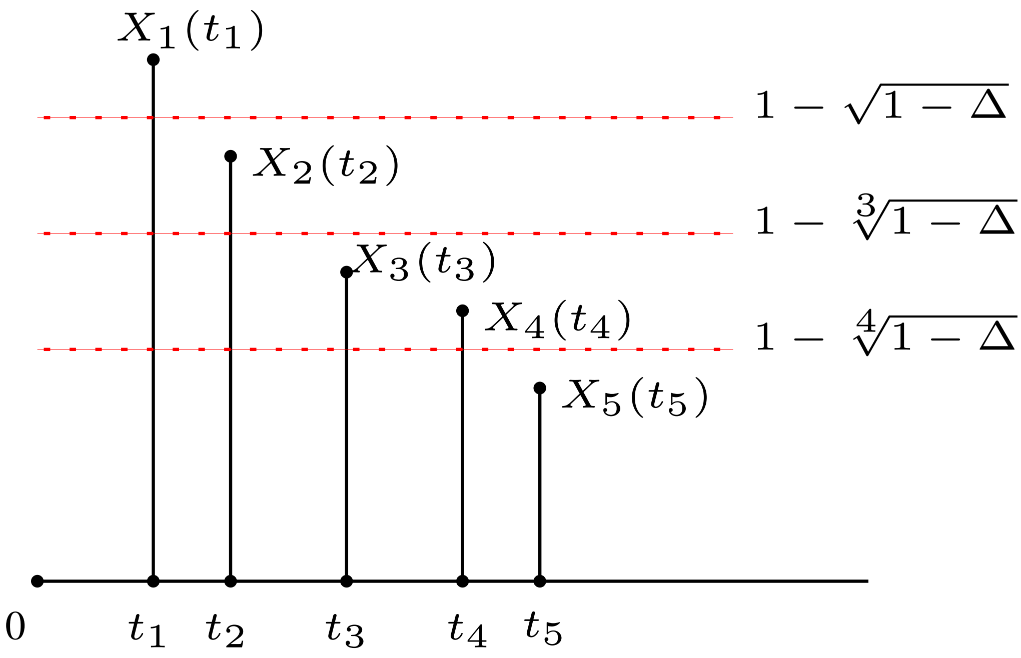

FIG. 17: *Illustrating our adaptive aggregation strategy.*

announces the target $1-\sqrt[4]{1-\Delta}$. This third rounds yields four sensors whose individual values exceed the announced target and, consequently, the fire event will be reported.

The first seven thresholds are listed in Table 3, where $\Delta = 0.99$. We can see that the seven-th thresholds is already below $\frac{1}{2}$, so in real application, thresholds' level is limited to 6 for this situation.

TABLE 3: First seven thresholds in adaptive aggregation strategy

m	1	2	3	4	5	6	7
$1-\sqrt[m]{1-\Delta}$	0.99	0.9	0.78	0.68	0.60	0.54	0.48

Notice that with each round, the value of the information decays. When the aggregator and other sensors wait for more sensors to confirm, their value will decay. In order that their values are all above the final threshold, their time interval should satisfy the following constraint.

For any i and k that $1 \le i < k \le n$, value X_i collected at time t_i is larger than value X_k collected at time t_k where $X_i > X_k$, $t_i < t_k$, $X_i \ge 1-\sqrt[i]{1-\Delta}$ and $X_k \ge 1-\sqrt[k]{1-\Delta}$, after X_i

decreased from t_i to t_k, it should still be larger than the k^{th} threshold $1-\sqrt[k]{1-\Delta}$, so the constraint is what described in inequality (5.5)

$$t_k - t_i < \frac{1}{\mu}\ln\frac{1-\sqrt[i]{1-\Delta}}{1-\sqrt[k]{1-\Delta}}, \qquad 1 \le i < k \le n \tag{5.5}$$

while $X_1(t_1) > X_1(t_2) > \cdots > X_n(t_n)$.

Proof. For $\forall i$, $\forall k$ such that $1 \le i < k \le n$, we have

$$\begin{aligned} X_i &> X_k \\ t_i &< t_k \\ X_i &> 1-\sqrt[i]{1-\Delta} \\ X_k &> 1-\sqrt[k]{1-\Delta} \end{aligned}$$

The value of X_i collected at time t_i decrease to time t_k is

$$X_i(t_i)e^{-\mu(t_k-t_i)}$$

then,

$$\begin{aligned} X_i(t_i)e^{-\mu(t_k-t_i)} &> (1-\sqrt[i]{1-\Delta})e^{-\mu(t_k-t_i)} \\ &> 1-\sqrt[k]{1-\Delta} \\ &\Rightarrow \\ (1-\sqrt[i]{1-\Delta})e^{-\mu(t_k-t_i)} &> 1-\sqrt[k]{1-\Delta} \\ e^{-\mu(t_k-t_i)} &> \frac{1-\sqrt[k]{1-\Delta}}{1-\sqrt[i]{1-\Delta}} \qquad (\because 0<\Delta<1,\ \text{so } 0<1-\sqrt[i]{1-\Delta}<1) \\ -\mu(t_k-t_i) &> \ln\frac{1-\sqrt[k]{1-\Delta}}{1-\sqrt[i]{1-\Delta}} \\ t_k-t_i &< \frac{1}{\mu}\ln\frac{1-\sqrt[i]{1-\Delta}}{1-\sqrt[k]{1-\Delta}}. \end{aligned}$$

□

When $1 \le i < k \le n$, inequality (5.5) is meaningful, i.e. the time window $W(k,i) = t_k - t_i > 0$.

Proof. Consider function $y = f(x) = a^x$, where $0 < a < 1$ and $x \in R^+$, then

$$\frac{dy}{dx} = a^x \ln a < 0 \qquad (\because \ln a < 0)$$

and

$$0 < a^x < 1$$

i.e., $f(x)$ is decreasing in R^+, so,

$$\begin{aligned}
&0 < \Delta < 1 \\
&\Rightarrow 0 < 1 - \Delta < 1 \\
&\Rightarrow \sqrt[i]{1-\Delta} < \sqrt[k]{1-\Delta} \quad (\because 1 \leq i < k \leq n) \\
&\Rightarrow 1 - \sqrt[i]{1-\Delta} > 1 - \sqrt[k]{1-\Delta} \\
&\Rightarrow \frac{1-\sqrt[i]{1-\Delta}}{1-\sqrt[k]{1-\Delta}} > 1 \\
&\Rightarrow \ln \frac{1-\sqrt[i]{1-\Delta}}{1-\sqrt[k]{1-\Delta}} > 0 \\
&\Rightarrow W(k,i) = t_k - t_i > 0.
\end{aligned}$$

Thus inequality (5.5) is meaningful. □

In practice, the number n of sensors for aggregation is a function of the deployment density and, in general, is application-specific. From equation (5.5), we also see that the length of time window between consecutive values in adaptive strategy is also proportional to the mean life-time of values. i.e., in more emergent event, the mean life-time of values is more short, the collection of data needs to be more frequent and the reaction time is more stringent. How frequent will it be? The properties of function $W(k,i)$ will reveal this frequency.

$$W(k,i) = t_k - t_i = \frac{1}{\mu} \ln \frac{1-\sqrt[i]{1-\Delta}}{1-\sqrt[k]{1-\Delta}} \qquad (1 \leq i < k \leq n). \tag{5.6}$$

First, two boundaries of $W(k,i)$ are presented:

$$\lim_{i\to\infty, k\to\infty} W(k,i) = 0. \tag{5.7}$$

Proof.

$$\lim_{i\to\infty,k\to\infty} W(k,i) = \lim_{i\to\infty,k\to\infty} \frac{1}{\mu} \ln \frac{1-\sqrt[i]{1-\Delta}}{1-\sqrt[k]{1-\Delta}}$$
$$= \lim_{i\to\infty,k\to\infty} \frac{1}{\mu} \ln \frac{1-(1-\Delta)^{\frac{1}{i}}}{1-(1-\Delta)^{\frac{1}{k}}}$$

Inside function ln, it is an $\frac{0}{0}$ indefinite

$$= \lim_{i\to\infty,k\to\infty} \frac{1}{\mu} \ln \frac{(1-\Delta)^{\frac{1}{i}} \ln(1-\Delta)}{(1-\Delta)^{\frac{1}{k}} \ln(1-\Delta)} \qquad \text{(By Hopital's Law)}$$
$$= \lim_{i\to\infty,k\to\infty} \frac{1}{\mu} \ln(1-\Delta)^{\frac{1}{i}-\frac{1}{k}}$$
$$= 0.$$

□

Limit (5.7) says that the length of consecutive time window turns to zero in order to collect future infinitesimal values from current infinitesimal values, in other words, the data-collection frequency needs to be infinitely large which is impractical. On the other hand, emergency should be responded as quickly as possible.

$$\lim_{k\to\infty} W(k,i) = \infty. \tag{5.8}$$

The proof for limit (5.8) is similar to the proof for limit (5.7). This limit says if infinitesimal values are taken into account for current decision, then the time window for them needs to be infinite which is also infeasible.

In reality, we need to respond to emergency in a timely manner, according to the discussion in section 4.12, it is practical to aggregate values before their flip-flop of confidence, i.e.,

$$1-\sqrt[k]{1-\Delta} > \frac{1}{2}.$$

From which, we get

$$k < -\log_2(1-\Delta). \tag{5.9}$$

Equation (5.9) gives engineers a design guidance to determine the practical number of sensors for aggregating values in an emergency for alert. Substitute $\Delta = 0.99$ into it and get $k < 7$, i.e., five or six sensors is a practical choice for aggregation in emergency. Figure 18 shows the practical number of sensors for different final thresholds. It shows that the

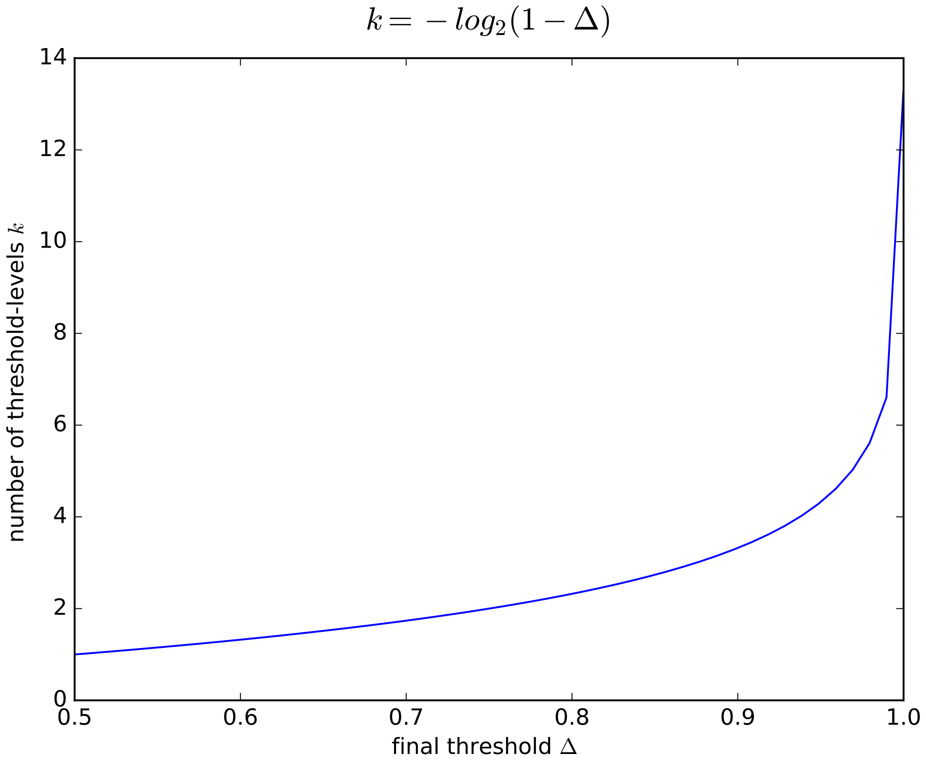

FIG. 18: Practical number of sensors for final threshold

higher threshold, the more threshold-levels, or sensors for aggregation are needed. We will choose $\Delta = 0.99$ this relatively high threshold for following simulations.

In order to get an estimate for the data-collection frequency, we represent $W(i,k)$ in a more general form as equation (5.10).

$$w(i,x) = \frac{1}{\mu} \ln \frac{1-\sqrt[i]{1-\Delta}}{1-\sqrt[i+x]{1-\Delta}} \qquad x \in R^+, 1 \le i \tag{5.10}$$

Here i is considered as a parameter and x is considered as a independent variable. Choose i from $\{1,2,3,4\}$ and $x \in [1,7]$, a family of this functions is illustrated in Figure 19.

From Figure 19, and comparing with fixed aggregation strategy, the time window started from i^{th} threshold to $(i+x)^{th}$ threshold is far more slack in adaptive aggregation strategy, which gives the monitoring system enough time for timely and reliable response.

For a numerical illustration, we give the following table for equation (5.6) with $i = 1,2,3,4$ respectively. in all of them, $\Delta = 0.99$.

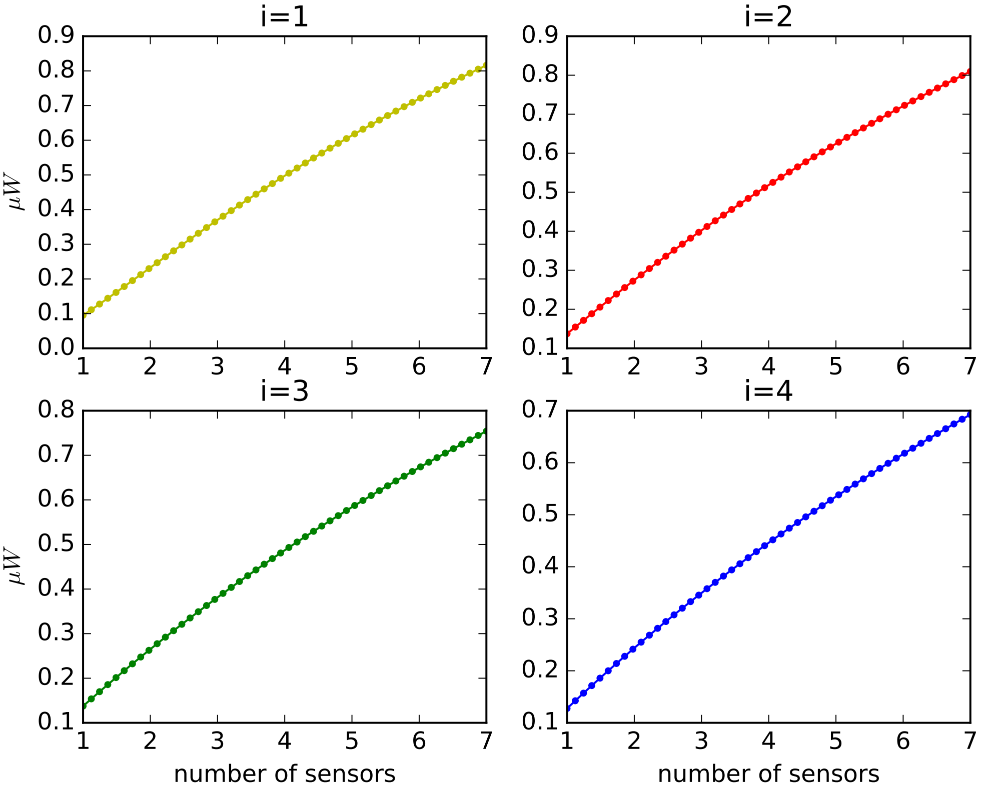

FIG. 19: *Ratio of the length of time window to the mean life-time of value vs number of sensors in adaptive aggregation strategy*

In Table 4, we can see that $W(i+1,i)$ has a peak at $W(4,3) = 0.1375$, $W(i+2,i)$ has a peak at $W(4,2) = 0.275$, $W(i+3,i)$ has a peak at $W(5,2) = 0.402$, $W(i+4,i)$ has a peak at $W(6,2) = 0.519$, and so on. This trend can be confirmed by rewriting equation (5.10) into another general form as equation (5.11).

$$w(x+k,x) = \frac{1}{\mu}\ln\frac{1-\sqrt[x]{1-\Delta}}{1-\sqrt[k+x]{1-\Delta}} \qquad x\in[1,\infty), 1\leq k \text{ and } k\in N \tag{5.11}$$

We take parameter $i = \{1,2,3,4\}$, the time windows cover from consecutive thresholds to continuous five thresholds, then draw the respective curves for equation (5.11) in Figure 20, 21, 22 and 23. From these figures, the best aggregation strategy in adaptive aggregation strategy is aggregation of a value right above next threshold or the one after the next threshold.

These properties of the function in (5.11) can be deduced as follows. As function (5.11)

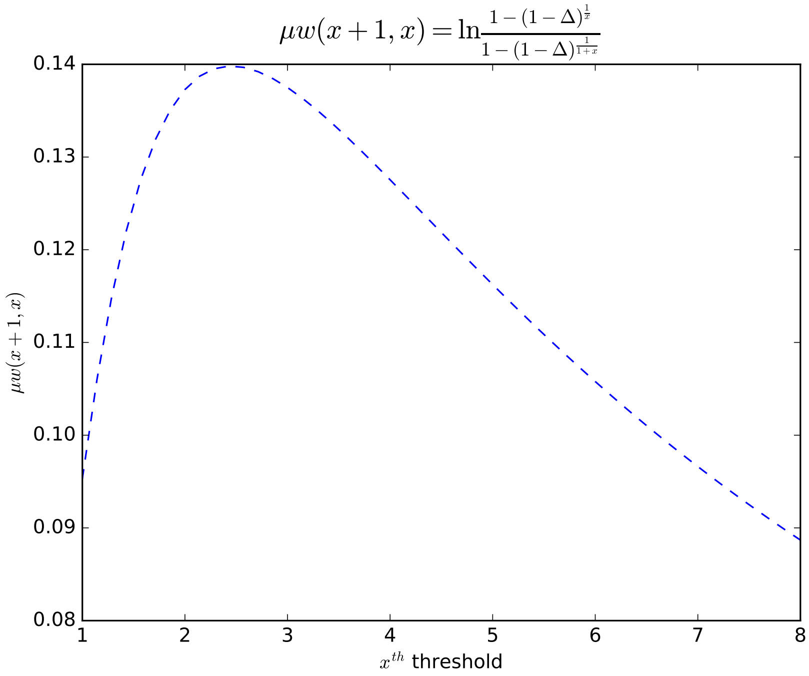

FIG. 20: *Ratio of the length of consecutive time window to the mean life-time of value in adaptive aggregation strategy when* $i = 1$

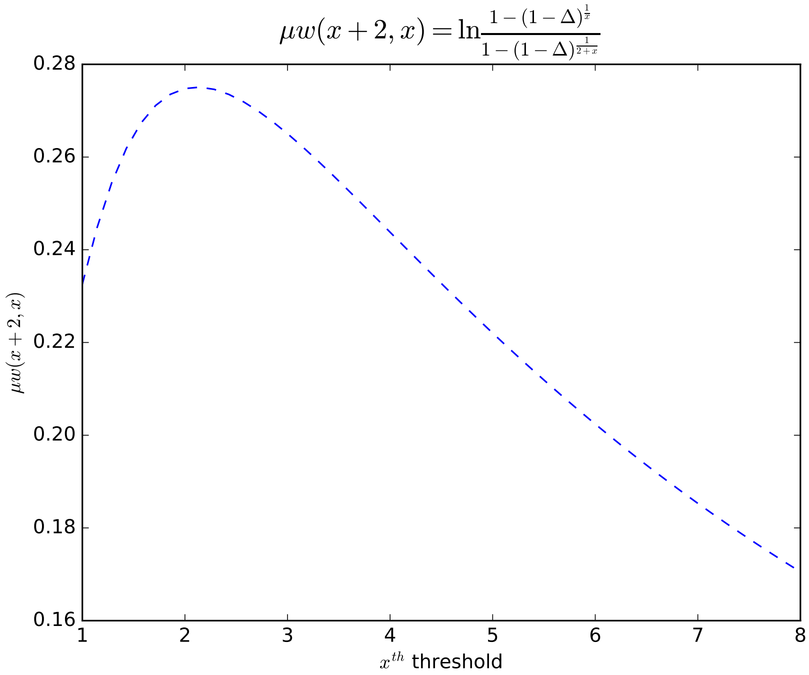

FIG. 21: *Ratio of the length of time window covering three thresholds to the mean life-time of value in adaptive aggregation strategy when* $i = 2$

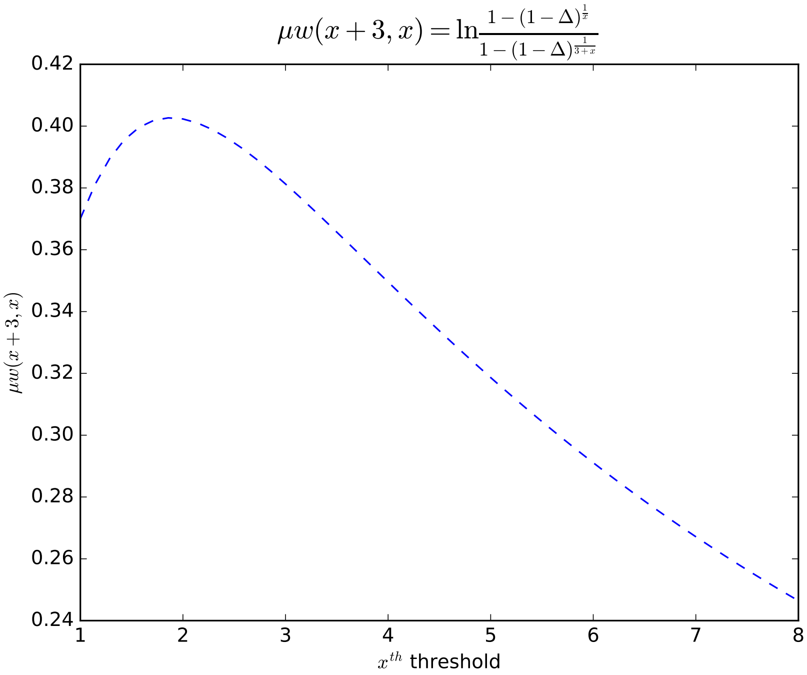

FIG. 22: *Ratio of the length of time window covering four thresholds to the mean life-time of value in adaptive aggregation strategy when* $i = 3$

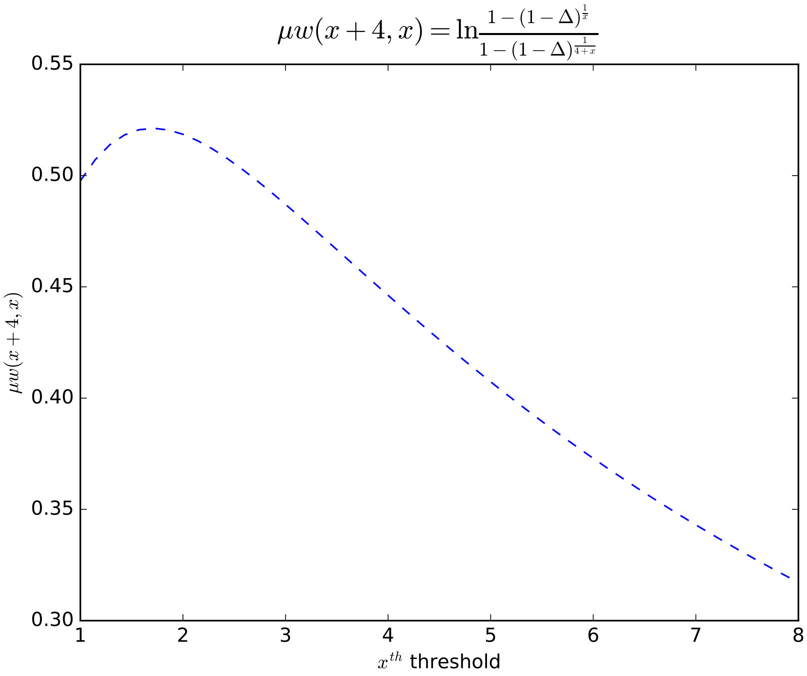

FIG. 23: *Ratio of the length of time window covering five thresholds to the mean life-time of value in adaptive aggregation strategy when* $i = 4$

TABLE 4: Ratio of the length of time window to the mean life-time of value in adaptive aggregation strategy

i=1						
k^{th} **threshold**	2	3	4	5	6	7
μW	0.0953	0.233	0.370	0.498	0.614	0.720

i=2						
k^{th} **threshold**	3	4	5	6	7	8
μW	0.1373	0.275	0.402	0.519	0.624	0.721

i=3						
k^{th} **threshold**	4	5	6	7	8	9
μW	0.1375	0.265	0.381	0.487	0.583	0.672

i=4						
k^{th} **threshold**	5	6	7	8	9	10
μW	0.1275	0.243	0.349	0.446	0.535	0.617

has the same extreme point as function $f(x)$ in (5.12).

$$f(x) = \ln \frac{1 - \sqrt[x]{1-\Delta}}{1 - \sqrt[k+x]{1-\Delta}} \qquad x \in [1, \infty), 1 \leq k \text{ and } k \in N. \tag{5.12}$$

Take derivative of the function (5.12) with respect to x, we get

$$\begin{aligned} \frac{df(x)}{dx} = \frac{1}{1-(1-\Delta)^{\frac{1}{x}}} & \left(\frac{(1-\Delta)^{\frac{1}{x}} \ln(1-\Delta)}{x^2 \left(1-(1-\Delta)^{\frac{1}{k+x}}\right)} \right. \\ & \left. - \frac{(1-\Delta)^{\frac{1}{k+x}} \left(1-(1-\Delta)^{\frac{1}{x}}\right) \ln(1-\Delta)}{(k+x)^2 \left(1-(1-\Delta)^{\frac{1}{k+x}}\right)^2} \right) \left(1-(1-\Delta)^{\frac{1}{k+x}}\right). \end{aligned} \tag{5.13}$$

Let $\frac{df(x)}{dx} = 0$ and use numerical solver, we get Table 5.

TABLE 5: Maximum ratio of the length of time window to the mean life-time of value in adaptive aggregation strategy

k^{th} **neighbor**	1	2	3	4
x_m	2.4505	2.1224	1.8818	1.7042
$\mu w(x_m + k, x_m)$	0.1398	0.2751	0.4027	0.5213
$[x_m]$	2	2	2	2
$\mu w([x_m] + k, [x_m])$	0.1373	0.2748	0.4023	0.5186

In Table 5, $[x_m]$ denotes the closest integer to x_m. Table 5 also says that the sensor has the second threshold has the maximum time window for aggregating values from following sensors. This is an advantageous point for adaptive aggregation strategy since the first sensor that detected an anomaly is usually not reliable enough to trigger an alarm, after it had a second value for confirmation, then the second one has good enough time for a third one, the further, the time window becomes shorter. This makes sense, since emergencies should be reported on time. So three to six sensors is always a good number to make a confirmation on an emergency, it is a optimum balance between reliability and timeliness.

Take second derivative of the function (5.12) with respect to x, we get

$$
\begin{aligned}
&\frac{d^2f(x)}{dx^2}\\
&= \frac{(-\Delta+1)^{\frac{1}{k+x}}\log(-\Delta+1)}{(k+x)^2\left(-(-\Delta+1)^{\frac{1}{x}}+1\right)}\left(-\frac{(-\Delta+1)^{\frac{1}{k+x}}\left(-(-\Delta+1)^{\frac{1}{x}}+1\right)\log(-\Delta+1)}{(k+x)^2\left(-(-\Delta+1)^{\frac{1}{k+x}}+1\right)^2}\right.\\
&\qquad\left.+\frac{(-\Delta+1)^{\frac{1}{x}}\log(-\Delta+1)}{x^2\left(-(-\Delta+1)^{\frac{1}{k+x}}+1\right)}\right)+\frac{1}{-(-\Delta+1)^{\frac{1}{x}}+1}\left(-(-\Delta+1)^{\frac{1}{k+x}}\right.\\
&\qquad\left.+1\right)\left(\frac{2(-\Delta+1)^{\frac{2}{k+x}}\left(-(-\Delta+1)^{\frac{1}{x}}+1\right)\log^2(-\Delta+1)}{(k+x)^4\left(-(-\Delta+1)^{\frac{1}{k+x}}+1\right)^3}\right.\\
&\qquad+\frac{2(-\Delta+1)^{\frac{1}{k+x}}\left(-(-\Delta+1)^{\frac{1}{x}}+1\right)\log(-\Delta+1)}{(k+x)^3\left(-(-\Delta+1)^{\frac{1}{k+x}}+1\right)^2}\\
&\qquad+\frac{(-\Delta+1)^{\frac{1}{k+x}}\left(-(-\Delta+1)^{\frac{1}{x}}+1\right)\log^2(-\Delta+1)}{(k+x)^4\left(-(-\Delta+1)^{\frac{1}{k+x}}+1\right)^2}\\
&\qquad-\frac{2(-\Delta+1)^{\frac{1}{x}}(-\Delta+1)^{\frac{1}{k+x}}\log^2(-\Delta+1)}{x^2(k+x)^2\left(-(-\Delta+1)^{\frac{1}{k+x}}+1\right)^2}-\frac{2(-\Delta+1)^{\frac{1}{x}}\log(-\Delta+1)}{x^3\left(-(-\Delta+1)^{\frac{1}{k+x}}+1\right)}\\
&\qquad\left.-\frac{(-\Delta+1)^{\frac{1}{x}}\log^2(-\Delta+1)}{x^4\left(-(-\Delta+1)^{\frac{1}{k+x}}+1\right)}\right)\\
&\quad-\frac{(-\Delta+1)^{\frac{1}{x}}\log(-\Delta+1)}{x^2\left(-(-\Delta+1)^{\frac{1}{x}}+1\right)^2}\left(-\frac{(-\Delta+1)^{\frac{1}{k+x}}\left(-(-\Delta+1)^{\frac{1}{x}}+1\right)\log(-\Delta+1)}{(k+x)^2\left(-(-\Delta+1)^{\frac{1}{k+x}}+1\right)^2}\right.\\
&\qquad\left.+\frac{(-\Delta+1)^{\frac{1}{x}}\log(-\Delta+1)}{x^2\left(-(-\Delta+1)^{\frac{1}{k+x}}+1\right)}\right)\left(-(-\Delta+1)^{\frac{1}{k+x}}+1\right).
\end{aligned}
\tag{5.14}
$$

Through numerical verification, $\frac{d^2f(x)}{dx^2}<0$ at x_m, so the extreme value of $f(x)$ at its extreme point is maximum.

5.1.4 SIMULATION RESULTS

Imagine that fire just broke out on a ship instrumented by a set of relevant sensors, including temperature and humidity sensors, light and smoke detectors, etc. [60]. To keep the aggregation operator simple, in what follows we assume the existence of temperature sensors only. Aggregating data across sets of sensors detecting different attributes of a fire event would proceed along similar lines, but is not covered here.

Imagine a *critical* temperature range $K = [100, 1000]$°C. The various sensors take temperature readings $T_1, T_2, \cdots$. The value X_i associated with temperature T_i is defined to be

$$X_i = \Pr[T_i \in K | F]$$

that is, the *conditional probability* of temperature T_i being recorded by a sensor, given the event F that fire is present. In order to stamp out noise, X_i is taken to be 0.9 if T_i is in the critical range K and 0 otherwise.

$$X_i = \begin{cases} 0.9 & T_i \in K \\ 0 & T_i \notin K. \end{cases} \tag{5.15}$$

We define the aggregator $\diamondsuit$ as follows

$$X_i \diamondsuit X_j = \Pr[\{T_i \in K\} \cup \{T_j \in K\} | F]. \tag{5.16}$$

In other words, aggregating two values is tantamount to computing the conditional probability of a union of events. Assuming that the sensors act independently, (5.16) implies that

$$\begin{aligned} X_i \diamondsuit X_j &= P[\{T_i \in K\} \cup \{T_j \in K\} | F] \\ &= \Pr[T_i \in K | F] + P[T_j \in K | F] \\ &- \Pr[\{T_i \in K\} \cap \{T_j \in K\} | F] \\ &= \Pr[T_i \in K | F] + P[T_j \in K | F] \\ &- \Pr[T_i \in K | F] \; P[T_j \in K | F] \\ &= X_i + X_j - X_i X_j, \end{aligned}$$

confirming that the aggregation operator $\diamondsuit$ is the one defined in (4.14). One can also see that $\diamondsuit$ has the property that the value of the aggregated information increases with the number of sensor readings in the critical temperature range.

For a group of n sensors, once their temperature reading is available, can evaluate the corresponding X_i by a simple table lookup. The aggregated value of the various X_is is described in equation (5.17).

$$\begin{aligned}\Diamond_{i=1}^{n} X_i &= Pr\{T_1 \cup T_2 \cup \cdots \cup T_n | F\} \\ &= \sum_{m=1}^{n} (-1)^{m+1} \sum_{1 \le i_1 < i_2 < \cdots < i_m \le n} Pr\{T_{i_1} \cap T_{i_2} \cap \cdots \cap T_{i_m} | F\} \\ &= \sum_{m=1}^{n} (-1)^{m+1} \sum_{1 \le i_1 < i_2 < \cdots < i_m \le n} X_{i_1} X_{i_2} \cdots X_{i_m}.\end{aligned} \tag{5.17}$$

By Lemma 4.7.1, equation (5.17) can be rewritten as equation (5.18), which gives us a simple way to calculate the aggregation of arbitrary number of values.

$$\begin{aligned}\Diamond_{i=1}^{n} X_i &= Pr\{T_1 \cup T_2 \cup \cdots \cup T_n | F\} \\ &= \sum_{m=1}^{n} (-1)^{m+1} \sum_{1 \le i_1 < i_2 < \cdots < i_m \le n} Pr\{T_{i_1} \cap T_{i_2} \cap \cdots \cap T_{i_m} | F\} \\ &= \sum_{m=1}^{n} (-1)^{m+1} \sum_{1 \le i_1 < i_2 < \cdots < i_m \le n} X_{i_1} X_{i_2} \cdots X_{i_m} \\ &= 1 - \prod_{i=1}^{n} (1 - X_i) \\ &= 1 - \prod_{i=1}^{n} (1 - Pr\{T_i | F\}).\end{aligned} \tag{5.18}$$

In the following simulations using MATLAB, the distribution of temperature in the fire is approximated by a linear model [67] with a plateau temperature of 1000°C and an ambient temperature of 20°C. The critical temperature range is [100,1000]°C, and the value discount constant μ is $0.1s^{-1}$ [80]. The fire propagation model is approximated by a dot source spreading out at the same rate in all directions at a speed of 1m/s [67]. The fire front has a temperature of 1000°C that decreases to the ambient temperature within 3m. The temperature sensors are deployed in rectangular lattices of size 3×2 in a plane with every side of 3m (Figure 24). The fire source location is randomly generated in one of the rectangles. Due to the small distances involved, wireless communication delays are ignored. We assume a sensor sampling time of 2s, and that the sensors are asynchronous, i.e., a new temperature reading will be reported every 2 seconds.

When a fire occurs and disseminates, every sensor senses a rising temperature and

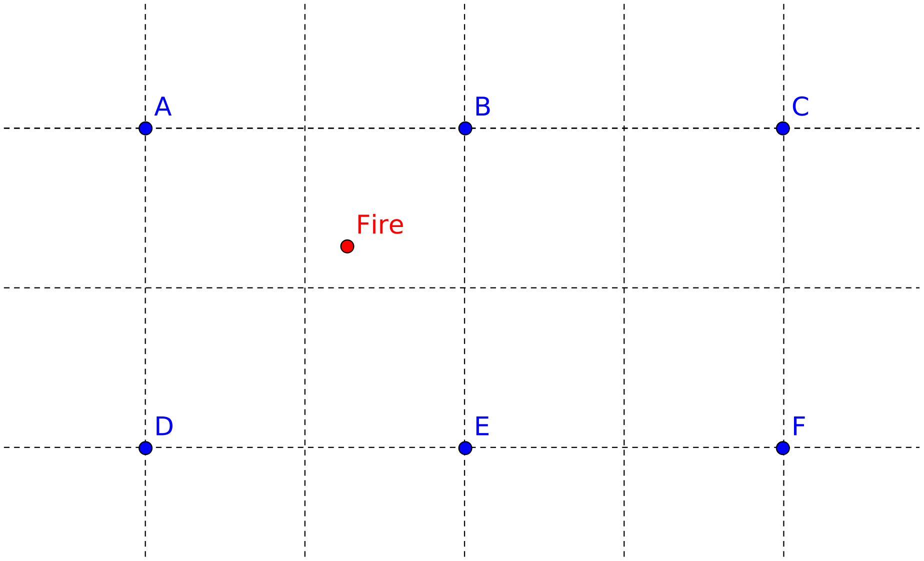

FIG. 24: *Illustrating our 3x2 lattice and fire location.*

computes the corresponding value. Figure 25 illustrates this process for all six sensors. In this scenario, the distances from the fire center to six sensors are listed in Table 6. The location for the fire center is $(3.758, 4.32)$. Where the distances of sensors to the fire center ascend in $B < C < A < E < F < D$, so sensor B takes value of 0.9 first, then sensor C, and so on.

TABLE 6: The distribution of fire center and six sensors in scenario 1

Sensor	x	y	Distance to fire center
A	0	3	3.98
B	3	3	1.52
C	6	3	2.61
D	0	0	5.72
E	3	0	4.39
F	6	0	4.87

Figure 26 illustrates fixed aggregation strategy for six sensors. Sensor B, the first sensor got value 0.9 and sent this value to the base station. This single value is below the fixed threshold 0.99, so no alarm is trigger at this time. Then the second sensor C got its value 0.9 and send it to the base station, however, the first value decayed a lot, the aggregation

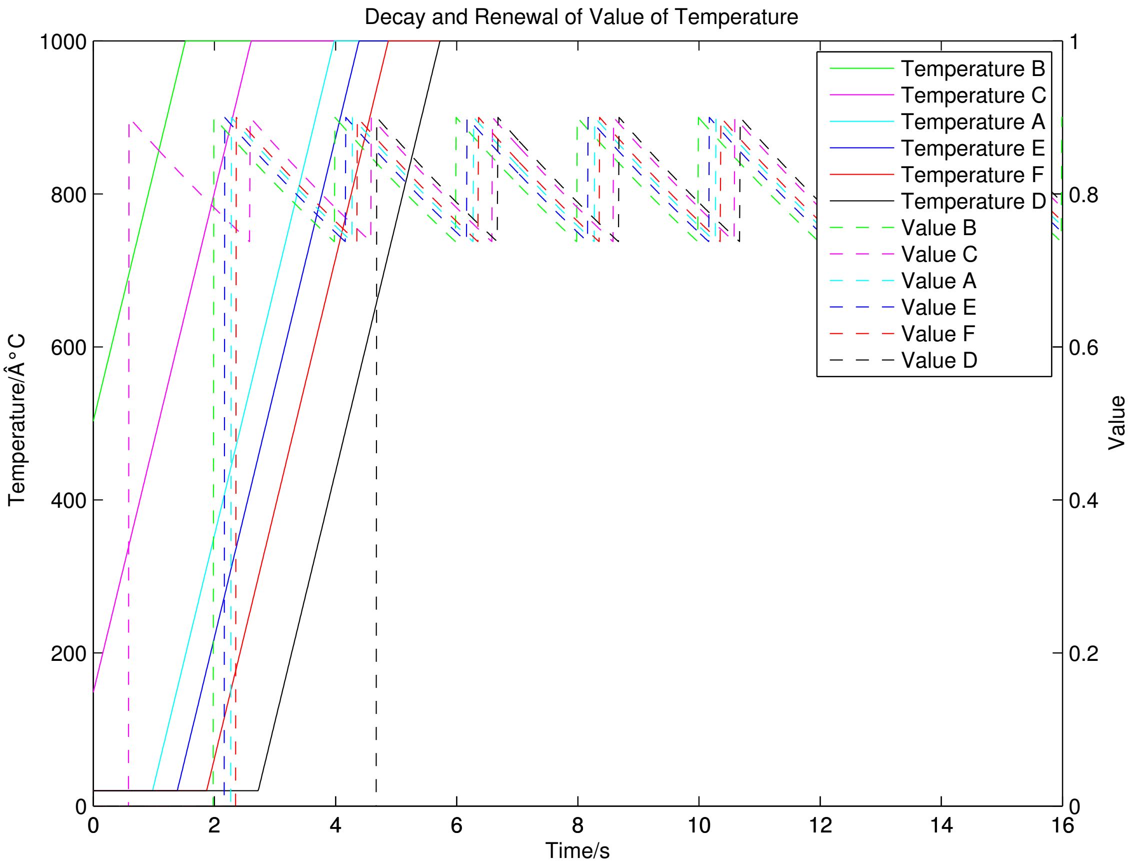

FIG. 25: *Illustrating renew and decay of values at six sensors.*

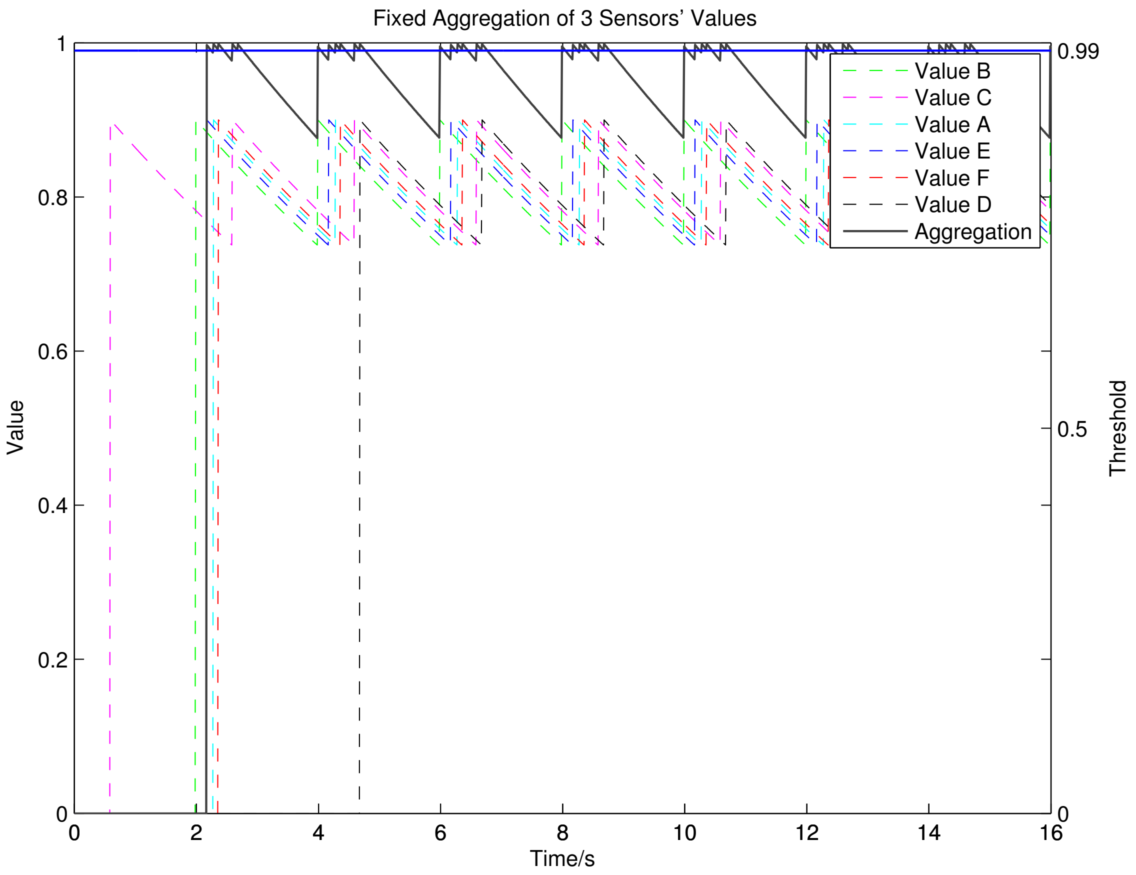

FIG. 26: *Illustrating fixed aggregation strategy for six sensors.*

of these two values are still below the threshold. During the time sensor A got its value 0.9 and send to the base station, even though the first and the second values have decayed a certain amount, finally, the aggregation of these three values are above the final threshold 0.99, so an alarm is triggered. Since the aggregated value also decays exponentially at the same decay rate as all other values, it quickly decayed to 0.9 in 0.1 seconds as calculated in Table 2. With time going by, as long as the alarm is not handled, the WSN will keep triggering alarm as the aggregated value keeps rushing above the threshold. Through this automatic periodic triggering mechanism, the alarm is reported timely and reliably.

Figure 27 illustrates adaptive aggregation strategy for six sensors. In order for a pair-wise comparison, the fire scenario here is the same as that for fixed aggregation strategy. In adaptive aggregation, there is no fixed aggregator, every sensor can work as an aggregator if applicable. When sensor B got value of 0.9 first, it broadcast this value to all other sensors. Only senors in a certain distance from B will accept this value, here all six sensors

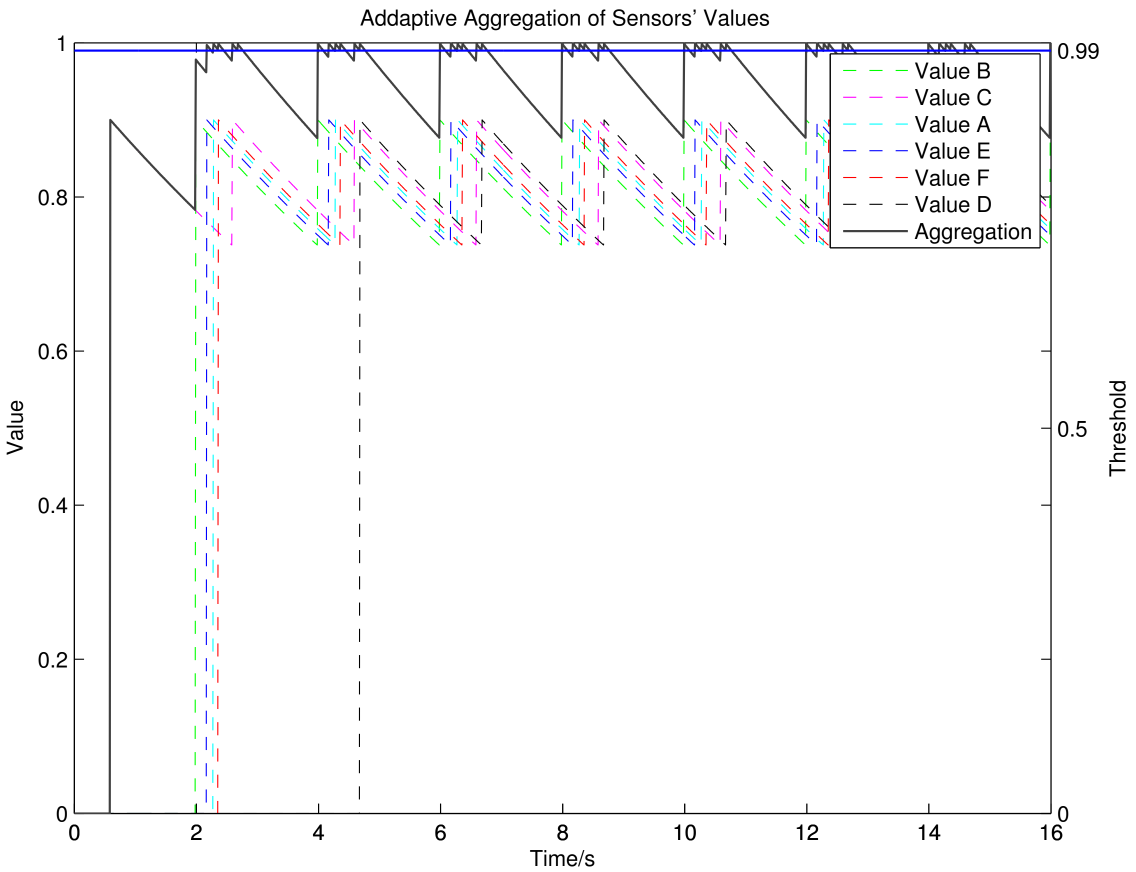

FIG. 27: *Illustrating adaptive aggregation strategy for six sensors.*

are not far away from sensor B, so the accident happened at B is very possible affects them as well. Sensor C, got value of 0.9 secondly, it broadcast this value again as sensor B and aggregated this fresh value with the decayed value it received from B a moment ago, sensor B also did the same aggregation as sensor C. As the aggregated values are not above the final threshold 0.99, so no alarm is triggered and these values continue decaying. When sensor A got value of 0.9, it broadcast this value and aggregated it with those decayed values from sensor B and C, sensor B and C also did the same aggregation. This time, these three sensors, sensor B, C and A all had aggregated value above 0.99, they consequently all triggered alarms. All other sensors do not have values of 0.9, they will keep passive and do not trigger alarm to save energy. From this adaptive aggregation strategy and alarm procedure, it is clear that adaptive aggregation strategy works more timely and reliably than fixed aggregation strategy which depends heavily on the base station.

Figure 28 illustrates this process for a single sensor to show more details. The dots are

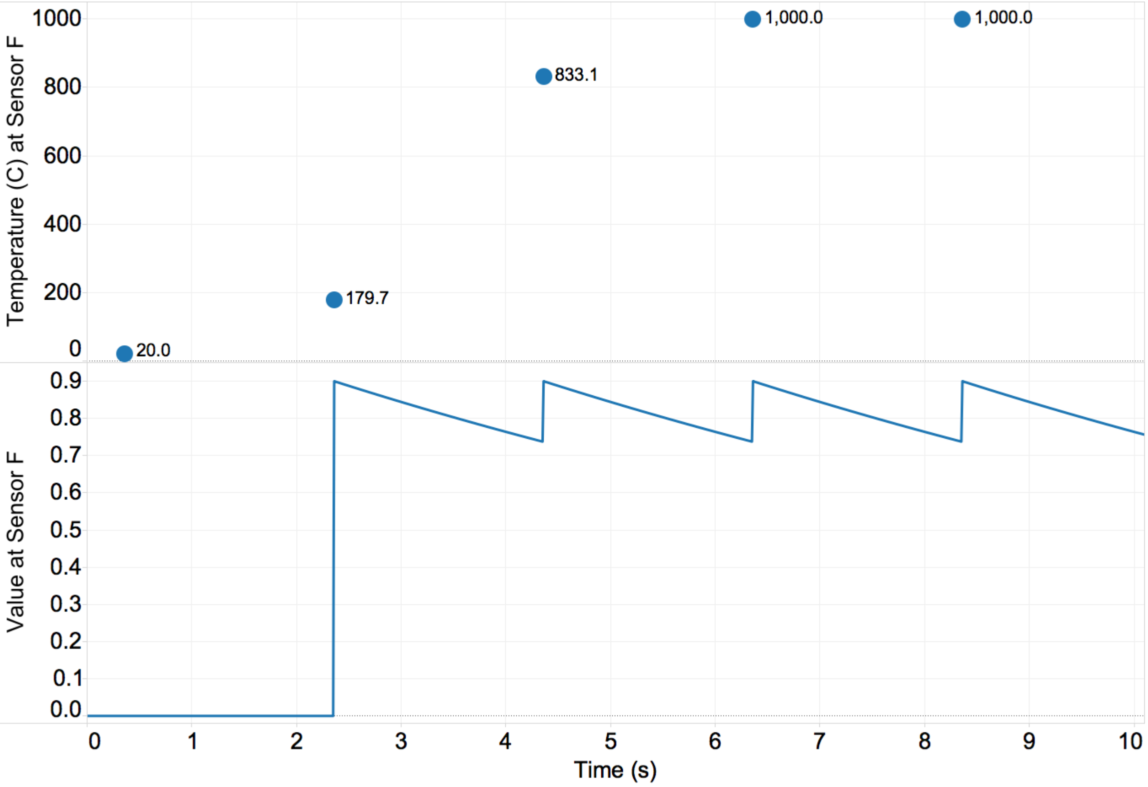

FIG. 28: *Illustrating the decay of temperature values.*

sensed temperatures, sampled every 2s. The solid line is the corresponding value, which jumps from 0 to 0.9 when the sensed temperature enters the critical range. It then decays until the next sample and the value is renewed. The process repeats to form the shape of a saw-tooth.

With fixed thresholding, the fire alarm is set if the aggregated value of three sensors is above a threshold of 0.99. In Figure 29, the top six lines that oscillate between 0.7 to 0.9 are the values of six sensors' temperature. The middle panel shows the aggregated value with fixed thresholding. The alarm is triggered repeatedly every time the aggregated value rises above the threshold Δ.

With fixed thresholding, a base station is required. We can achieve the same triggering of the alarm without a base station by using adaptive thresholding. Here, the initial threshold Δ_0 is 0.8, and the final threshold Δ is 0.99. In this case, the maximum number of sensors needed for reporting a fire event is five. The bottom panel in Figure 29 shows the adaptive aggregated value. Once a single sensor's value rises above the initial threshold, it becomes the aggregator (in this case, sensor *B*). As new readings are shared,

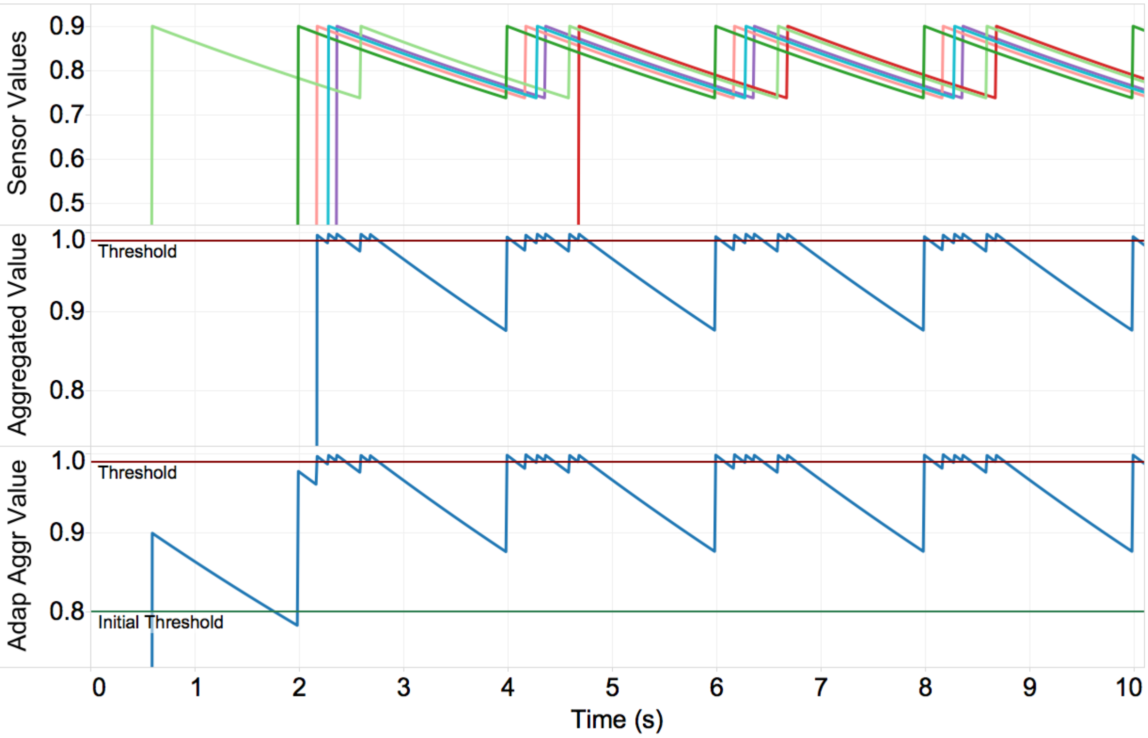

FIG. 29: *Illustrating two thresholding modes.*

the aggregated value rises above the 0.99 threshold and the alarm is triggered. As can be seen, the pattern of alarms in the bottom panel (adaptive thresholding) is similar i to that in the middle panel (fixed thresholding), without the need for a base station performing the aggregation.

5.1.5 ALGORITHMS FOR FIXED AND ADAPTIVE AGGREGATION STRATEGY

In this subsection, we developed two algorithms for fixed aggregation and one algorithm for adaptive aggregation strategy. These algorithms are adaptable to translate into any programing language for an implementation such as Nes C on IRIS motes.

IRIS [71] motes are wireless sensor networking products from MEMSIC:

- Equip with low power micro-controller — Atmel ATMega 1281 running at 8 MHz. This micro-controller draws 8 mA in active mode and only 8μA in sleep mode in total;
- Equip with IEEE 802.15.4 compliant RF transceiver RF230 from Atmel working at

2.4 GHz ISM, whose baud-rate is 250 $Kbits/s$, transmission power is 3 dBm and receive sensitivity is -101 dBm;

- Work on 2XAA batteries.
- The IRIS mote has a 51-pin expansion connector supports for I2C, SPI, UART interface, digital I/O and analog interface which can be plugged with light, temperature, barometric pressure, acceleration/seismic, acoustic, magnetic and other MEMSIC Sensor Boards such as MTS101CA, MTS300, MTS310, MTS400, etc.
- They support real-time operating systems such as TinyOS, FreeRTOS, ContikiOS, etc.

Algorithm 1 describes the key program running on each sensor for fixed aggregation strategy. From line 1 to line 4, it is system initialization:

- *Set up sampling timer with period T_s:* a system timer is set up to generate interrupt periodically every T_s, some micro-controller (MCU) offered hardware connection from the timer interrupt flag to the enable flag of its analog to digital converter (ADC). If not, an ISR (interrupt service routine) is need to drive the ADC to sample the temperature from its thermo-sensor. Such as in our previous simulation, $T_s = 2s$.
- *Set up temperature sensor and corresponding ADC:* no matter the temperature sensor and corresponding ADC are integrated in the MCU or interfaced as peripherals, they are initialized to a certain precision and range. In sensing when it is saturated, an exception can be raised. For example, a temperature outside the critical temperature range at the high end is an anomaly.
- *Initialize semaphore:* A semaphore is initialized to synchronize the ADC and the reporting task that handles conversion from temperature to value and convey to base station. Every time a temperature is sampled, the semaphore is posted, and notify the task which pending on this semaphore to go. If the underline system does not provide semaphore, other synchronization mechanism should be applied.
- *Set up reporting task:* The reporting task handles conversion from temperature to value according to equation (5.15) and transmit the value that is inside the critical temperature range to base station. Its private stack, heap, priority and main function is configured here.

Line 5 describes the main task that converts temperature to value and handles wireless transmission. Its execution is pending on the semaphore that is posted by the ADC sampling ISR, so it runs at the same pace as the sampling routine periodically.

Algorithm 1: Algorithm on sensors for fixed aggregation strategy

```
   Input :
           $Addr_h$ — Address of the host sensor
           $T_s$ — Sampling period
           $F_s$ — Timer interrupt flag
           $K$ — Critical temperature range
           $V_K$ — Value assigned when the sampled temperature is in the critical
   temperature range
           $Addr_b$ — Address of the closest base station
   Output:
           $V$ — Value converted from temperature
1  Set up sampling timer with period $T_s$
2  Set up temperature sensor and corresponding ADC
3  Initialize semaphore
4  Set up reporting task
5  while true do
6      Wait for semaphore
7      Read sampled temperature
8      Convert temperature to value
9      if $V = V_K$ then
10         Send $V$ to $Addr_b$
11     end
12 end
```

In algorithm 2, it describes how the base station works during fixed aggregation strategy. It consists of two major routines: one for receiving values transmitted from other sensors, the other for aggregation. Similarly, a semaphore is used to synchronize them. Every time the receiving routine received a value, it posted the semaphore to release the aggregation routine from waiting. Then, the aggregation routine put a time stamp on the received value. After a time t_r, if it received a new value, it first decays all previous received value by a ratio of $e^{-\mu t_r}$ (this applies the multiplicatively concatenation of exponential decay by its property 3.7), then aggregated these decayed values with the new received value. If the aggregated value is above the preset threshold, it will trigger an alarm to call for human intervening. The receiving routine is usually the wireless receiving ISR.

Algorithm 2: Algorithm on base station for fixed aggregation strategy

```
Input :
        $X_i$ — Received value
        $n$ — Number of sensors for aggregation in fixed aggregation strategy
        $\Delta$ — Preset threshold
        $\mu$ — exponential discount rate
        $F_a$ — Flag of alarm
        $Addr_b$ — Address of the closest base station
Output:
        $V$ — Current aggregated value
Set up receiving task
Set up aggregation and alarm routine
Initialize semaphore
Set up logging task
while true do
    Wait for semaphore
    Read received value
    Decay previously received values
    Calculate the aggregated value according the fixed aggregation strategy
    if $V \geq \Delta$ then
        Trigger alarm for human intervening
    end
end
```

The algorithm 3 is a distributive algorithm on each sensor for adaptive aggregation strategy, i.e. every sensor runs the same algorithm. This algorithm contains four tasks:

- *Sampling task:* This task samples temperature every T_s, and post a semaphore for the aggregating task to continue. It is triggered by a periodic timer similar to Algorithm 1.
- *Receiving task:* This task receives values from other sensors and post a semaphore to release the aggregating task from waiting. It is mainly implemented in a wireless

receiving ISR.

- *Aggregating task:* This task waits on two semaphores in *any* mode, i.e., it runs on any available semaphores, and reads the corresponding value, puts it into the priority, decays all previous values in the queue accordingly. If the value is a sampled value and it is above the final threshold, it posts the semaphore to the alarm task immediately. If the sampled value is below the final threshold but above the initial threshold Δ_0, then the value is broadcast to its neighbors. Meanwhile, it moves the current threshold to the next level and waits for a value from neighbors which are above this new threshold level. Every time the values in the queue are recalculated, those values that are lower than the application-dependent N^{th} level are removed from the queue. If the current queue length is N and many values are removed such that the new length is less than N, the threshold level retreat to a higher level. This process repeats until the emergency is cleared.

- *Alarm task:* This task waits on the semaphore from the aggregating task, every time it gets posted, it triggers an alarm for human attention.

The distributiveness shows a great advantage of adaptive aggregation strategy which has much higher reliability than fixed aggregation strategy.

Algorithm 3: Distributive algorithm on each sensor for adaptive aggregation strategy

```
Input :
        $\Delta_0$ — Initial threshold
        $\Delta$ — Final threshold
        $m$ — Current level of threshold
        $T_s$ — Sampling period
        $K$ — Critical temperature range
        $V_K$ — Value assigned when the sampled temperature is in the critical
temperature range
        $N$ — Maximum number of sensors (values) for aggregation
        $X_i$ — Sampled or received values
        $Q$ — Priority queue storing values
        $\mu$ — exponential discount rate
Output:
        $V$ — Current aggregated value
Set up sampling timer with period $T_s$
Set up temperature sensor and corresponding ADC
Initialize semaphores
Set up receiving task
Set up aggregation and alarm routine
while true do
    Wait for any semaphores
    Read received or sampled value
    $m \leftarrow m + 1$ Calculate $m^{th}$ threshold
    Broadcast sampled value that is larger than $Delta_m$
    Push the value into $Q$
    Decay previously queued values
    Remove values that are lower than the current threshold from the queue
    Calculate the aggregated value according the adaptive aggregation strategy
    if $V \geq \Delta$ then
        Trigger alarm for human intervening
    end
end
```

5.2 APPLICATION OF TYPE 2 AGGREGATORS

5.2.1 A SCENARIO

The main goal of this section is to show how the theoretical concepts developed in Chapter 4 for Type 2 aggregation operator apply to a practically-relevant scenario.

Consider a fire event witnessed by a number of sensors deployed in a given area. For simplicity, assume that each sensor has collected a temperature value. Let $X_1, X_2, \cdots$ be, respectively, the sensed temperature values collected by the various sensors at times $t_1, t_2, \cdots$. Since the sensors have witnessed the same event, it is natural to assume that the random variables $X_1, X_2, \cdots$ come from the same underlying distribution X with finite expectation $E[X] < \infty$.

We assume that the X_is are independent and, moreover, they are independent of the times $t_1, t_2, \cdots$ at which the data was collected. These assumptions can be justified by the spatial diversity of the sensors.

For a generic sensor that has collected data at time t_i, we let $X_i(t_i)$ denote the value of this information when it was collected. By (3.2.7) the discounted value of this information at a later time t is

$$X_i(t) = X_i(t_i)e^{-\mu(t-t_i)}. \tag{5.19}$$

Further, as a QoS parameter intended to avoid reporting a false positive, we need a minimum of k individual temperatures to be aggregated. Given an expected temperature reported of 100°C, this requirement is tantamount to insisting on accumulating a total of $\Delta = k \times 100$ "temperature points" as a result of aggregation. In turn, this suggests $\diamondsuit =$ "$+''$ as a suitable aggregation operator. It is easy to confirm that for the chosen $\diamondsuit$ the distributivity property holds and so the results of Section 4.3 apply.

In this context, we are interested in evaluating the *expected* time-discounted value, $V(t)$, at time t, of the information collected by the sensors where $t \geq \max\{t_1, t_2, \cdots\}$. To answer this natural question we make the simplifying assumption that $t_1, t_2, \cdots$ are the times of a Poisson process with parameter $\lambda > 0$. In other words, λ is the rate at which the sensors that witnessed an event are ready to report their sensory data.

Theorem 5.2.1. *The expected time-discounted value, E[V(t)], of the information collected* *by sensors at times* $t_1, t_2, \cdots$ *is*

$$E[V(t)] = \frac{\lambda}{\mu} E[X]\left[1 - e^{-\mu t}\right] \tag{5.20}$$

where $\lambda > 0$ *is the rate at which the sensors collect their data and* $E[X]$ *is the common expectation of* $X_1, X_2, \ldots$*.*

Proof. Recall that we assumed that the sensors collected their data at the times of a Poisson process with parameter λ. By the Law of Total Expectation,

$$E[V(t)] = \sum_{n \geq 1} E[V(t)|\{N = n\}]P[\{N = n\}] \tag{5.21}$$

where N is the random variable that counts the number of sensors that have data ready for aggregation by time t. By (5.19), Theorem 3.2.6 the conditional expectation, $E[V(t)|N = n]$, can be written as

$$\begin{aligned} E[V(t)|\{N = n\}] &= E[\sum_{i=1}^{N} X_i(t_i)e^{-\mu(t-t_i)}|\{N = n\}] \\ &= \sum_{i=1}^{n} E[X_i(t_i)e^{-\mu(t-t_i)}] \end{aligned}$$

It is well known that, given that n Poisson events were recorded in $(0, t]$, their conditional distribution is uniform. Thus,

$$\begin{aligned}
& E[V(t)|\{N=n\}] \\
= & \sum_{i=1}^{n} E[X_i(t_i)e^{-\mu(t-t_i)}] \\
= & \sum_{i=1}^{n} E[X_i(t_i)e^{-\mu(t-U_i)}] \\
& \text{[where the } U_i\text{s are uniform in } (0,t]] \\
= & \sum_{i=1}^{n} E[X_i(t_i)]E[e^{-\mu(t-U_i)}] \\
& \text{[because the } X_i\text{s and } U_i\text{s are independent]} \\
= & \sum_{i=1}^{n} E[X]E[e^{-\mu(t-U_i)}] \\
& \text{[recall, } X \text{ is the common distribution of the } X_i\text{s]} \\
= & E[X]e^{-\mu t}\sum_{i=1}^{n} E[e^{\mu(U_i)}] \\
= & e^{-\mu t}E[X]\sum_{i=1}^{n}\int_0^t e^{\mu u}\frac{\mathrm{d}u}{t} \\
= & e^{-\mu t}E[X]\sum_{i=1}^{n}\frac{e^{\mu t}-1}{\mu t} = \frac{E[X]}{\mu t}\sum_{i=1}^{n}[1-e^{-\mu t}] \\
= & \frac{nE[X]}{\mu t}[1-e^{-\mu t}]. \qquad (5.22)
\end{aligned}$$

On plugging (5.22) back into (5.21), we obtain

$$\begin{aligned}
E[V(t)] &= \sum_{n\geq 1}\frac{nE[X]}{\mu t}[1-e^{-\mu t}]P[\{N=n\}] \\
&= \sum_{n\geq 1}\frac{nE[X]}{\mu t}[1-e^{-\mu t}]\frac{(\lambda t)^n}{n!}e^{-\lambda t} \\
&= \frac{e^{-\lambda t}E[X][1-e^{-\mu t}]}{\mu t}\sum_{n\geq 1}\frac{(\lambda t)^{(n-1)}}{(n-1)!} \\
&= \frac{e^{-\lambda t}E[X][1-e^{-\mu t}]\lambda t}{\mu t}e^{\lambda t} \\
&= \frac{\lambda}{\mu}E[X][1-e^{-\mu t}].
\end{aligned}$$

□

There are a number of interesting things to note here:

- The actual distribution of the X_is does not appear explicitly in Theorem 5.2.1. This is telling us that two quite different distributions with the same expectation are equivalent as far as Theorem 5.2.1 is concerned;
- $E[V(t)] = \frac{\lambda}{\mu}E[X][1-e^{-\mu t}]$ is an increasing function of time and

$$\lim_{t\to\infty} E[V(t)] = \frac{\lambda}{\mu}E[X].$$

  Thus, for every application-dependent threshold Δ, there exists an earliest time when Δ is exceeded.

Note that, as mentioned above, Theorem 5.2.1 allows us to evaluate the earliest time t at which the expected discounted value of the information collected by the sensors exceed an application-dependent threshold Δ. Thus, at time t, $E[V(t)] \geq \Delta$, or equivalently,

$$\frac{\lambda}{\mu}E[X][1-e^{-\mu t}] \geq \Delta.$$

Solving for t, we obtain

$$t \geq \frac{1}{\mu}\ln\frac{\lambda E[X]}{\lambda E[X]-\Delta\mu} \tag{5.23}$$

In fact, (5.23) states that a value of t exists only if $\lambda E[X] > \Delta\mu$ or, equivalently,

$$\Delta < \frac{\lambda}{\mu}E[X]. \tag{5.24}$$

Inequality 5.24 can be rewritten as

$$\Delta < \frac{\frac{1}{\mu}}{\frac{1}{\lambda}}E[X]. \tag{5.25}$$

Where $\frac{1}{\mu}$ is the mean life-time of values, and $\frac{1}{\lambda}$ is the mean length of reporting interval. If $\frac{1}{\mu} \leq \frac{1}{\lambda}$, i.e., collected values averagely vanished because of the untimely reporting, from inequality (5.25), we get $E(X) > \Delta$, however in practice it is the case that $E[X] < \Delta$, for otherwise there are no incentives for aggregation. Which requires $\frac{1}{\mu} > \frac{1}{\lambda}$, the reporting interval should be shorter than the mean life-time of values for a timely aggregation and reporting. However, if the reporting interval is too short, in wireless sensor network, the probability of collision will rise and lots of valuable resource especially wireless transmission resource are wasted, so an application-dependent trade-off is needed.

5.2.2 SOME DISCUSSIONS ON TYPE 2 AGGREGATORS

In this subsection, we evaluate our theoretical model for Type 2 aggregators through some discussions.

Effect of Aggregation Method

For a scenario in which the sensors collect information and report it to a central node. We want to evaluate the effect of wireless communication and aggregation methods on the *average time to aggregate*. Collisions in the wireless channel typically garble messages beyond recognition. Thus messages need to be retransmitted which will increase the time to receive all pieces of information to be aggregated. Clearly, the denser the traffic the more collisions and the more retransmissions causing delays in aggregation. A non-negligible side-effect of all this is that due to collision-caused time delays, the value of individual pieces of information decays and what is being aggregated has lesser value.

Furthermore, the aggregation method could affect the time to achieve the defined aggregation. This latency is due, to a large extent, to the logic behind the transfer of information for aggregation. For example, the aggregation method may use one of the following strategies:

(I) : wait until at least k nodes report;

(II) : wait until the aggregated value exceeds a threshold;

(III) : wait until there is enough spatial diversity in the reported information.

In all these example, the sensor nodes can decide about reporting in a way that the only the effective ones reports. For example, in the former method, it would be enough if only one node out of several neighbor nodes transmits the message. This can be performed through many methods such as cluster head or a probabilistic transmission policy. Independent of policy, transmitting less number of messages would reduce the traffic and faster reception of required information by the aggregation method.

Time for the Expectation of Aggregated Value to Exceed a Threshold

Recall that because of information value decay, the longer the time to aggregate, the lower the value of the aggregated information. The delay due to waiting for arriving pieces of information as well as collision in the reception of those pieces of information will result in reducing the value of information. From equation (3.10), the decayed value X_d of information with an initial value X_0 can be deduced:

$$\begin{aligned} X_d &= X_0 - X(t) && (5.26) \\ &= X_0 - X_0 e^{-\mu t} \\ &= X_0(1 - e^{-\mu t}) \\ &= X_0(1 - e^{-\frac{t}{\tau}}), && (5.27) \end{aligned}$$

where $\tau = \frac{1}{\mu}$, which is the mean life-time of values. Hence, the ratio R_d of decayed value at time t to the initial value is

$$R_d = \frac{X(t)}{X_0} = 1 - e^{-\frac{t}{\tau}}. \quad (5.28)$$

Some typical decay ratios are listed in Table 7.

TABLE 7: Typical ratios of decayed value to its initial value

$\frac{t}{\tau}$	1	2	3	4	5	6	7	8
$R_d = 1 - e^{-\frac{t}{\tau}}$	63.21%	86.47%	95.02%	98.17%	99.33%	99.75%	99.91%	99.97%

Figure 30 illustrates the ratio of decayed value to its initial value. It shows that:

- After 1 mean life-time, 63.21% value of initial information decayed;
- After 2 mean life-times, 86.47% value of initial information decayed;
- After 5 mean life-times, 99.33% value of initial information decayed, which can be considered the value is vanished.

This delay may be due to waiting for pieces of information to be generated by different sources and arriving at the aggregator, as well as delay due to the wireless channel conditions.

In the following simulation, we investigate how the rate of information generation in relation to decay should be designed to achieve the value of information as soon as

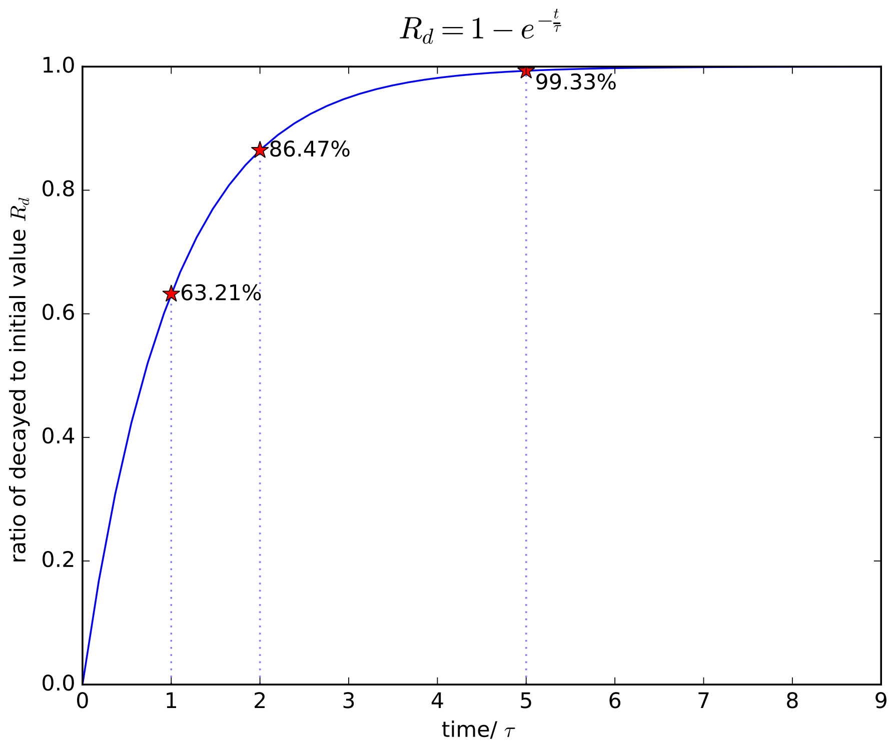

FIG. 30: Ratios of decayed value to its initial value

possible, so that a decision maker can make a proper determination. Here, we evaluate the applicability of the defined threshold time in (5.23). Recall that (5.23) shows the minimum time at which the expected value of information will exceed the threshold. This can be useful, for example, to plan for capturing an intruder. If we assume that an intruder staying time in the monitored area is an exponential random variable S with expected value $E[S] < \infty$ and the reaction time of the security personnel to arrive at location after receiving the alarm is a random variable R with expected value $E[R]$, then the expected detection time D can be written as $D \leq E[S] - E[R]$. In other words, the design of λ, i.e., sensor reporting periods, should not let t in (5.23) exceed D. In the following, we simulate this scenario to show how λ and μ can affect the effectiveness of the monitoring application. Assume that $E[S] = 300s$ and $E[R] = 100s$. This means that the detection time should be less than 200 seconds. In this simulation, we assume that V is selected uniformly from $\mathbb{R}[V_{min}, V_{max}]$, which are set to 0 and 100, respectively. This can be a valid setting since it could mean a direct functional mapping from the percentage of confidence that a sensor has in detecting the intruder. Then, we could set the threshold Δ to 196, which means that we need to have a confidence equal to two 98% confidence in detection of intrusion before reporting. Letting $VoI(t)$ stand for the random variable that keeps track of the value of information at time t, Table 8 shows the moments in time when the expected value, $E[VoI(t)]$, of $VoI(t)$ will be above Δ for theory and simulation. We show the average time over 10,000 trials for various values of λ and μ. As it turns out, the simulation results match the theoretical predictions well. In our defined scenario, we required that $E[VoI(t)]$ exceed Δ within 200 seconds. Table 8 shows that only configurations 5 and 6 meet that requirement, due to the arrival rate of 0.1, which has sensors reporting every 10 seconds.

TABLE 8: Time when $E[VoI(t)]$ exceeds a threshold

Configuration No.	λ	μ	*Theoretical Result*	*Simulated Result*
1	0.01	2.50×10^{-3}	1475s	1700s
2	0.01	1.25×10^{-3}	534s	557s
3	0.02	2.50×10^{-3}	267s	280s
4	0.02	1.25×10^{-3}	223s	247s
5	0.10	2.50×10^{-3}	41s	47s
6	0.10	1.25×10^{-3}	40s	46s

Number of alerts

In many situations, continuous alerts are generated since the emergency happened, developed and response is not carried out to clear the emergency on time. In this section, we developed quantified theory for the number of alerts from original reports and aggregated alerts. It is assumed that at the view of a base station, reports come as a Poisson process with reporting rate λ. The temperature T sampled and reported is modeled as a random variable with a bounded probability density distribution $f_T(t)$, so:

$$F_T(t) = P\{T \leq t\} = \begin{cases} 0 & T < T_l \\ \int_{u=T_l}^{t} \mathrm{d}F_T(u) & T_l \leq T < T_h \\ 1 & T \geq T_h \end{cases} \tag{5.29}$$

where $F_T(t)$ is the probability cumulative distribution. Figure 31 is an illustration for such a probability density distribution.

In monitoring, two thresholds are set: T_r is the threshold for a temperature to be aggregated, T_a is the threshold for alert for both original temperature and aggregated temperature. Notice that aggregation is taken at the time a temperature is reported, if the original

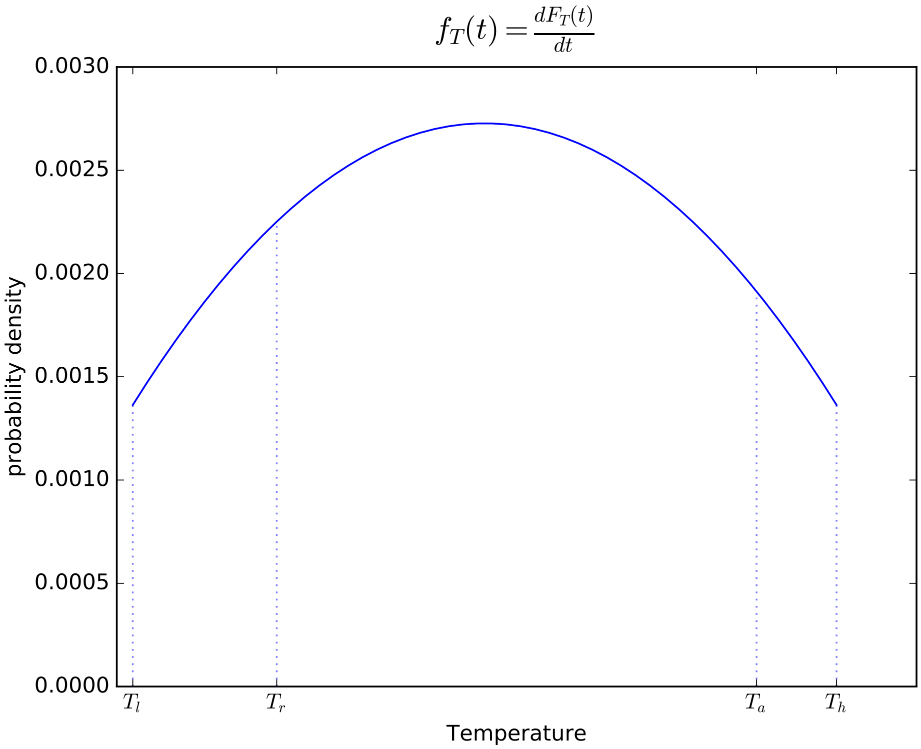

FIG. 31: A typical distribution of temperature around a fire

temperature and the aggregated temperature gotten at the same time are both above the alert temperature, then only one alert is generated. Then by equation (5.29), the probability for a sampled temperature above the reporting threshold T_r, $(T_l < T_r < T_h)$ is:

$$P\{T > T_r\} = 1 - F_T(T_r) \tag{5.30}$$

and the probability for a sampled temperature above the alert threshold T_a, $(T_r < T_a < T_h)$ is:

$$P\{T > T_a\} = 1 - F_T(T_a) < 1 - F_T(T_r) \tag{5.31}$$

Then within a certain time τ, since the reporting process is a Poisson process, the expected number of total reports is $\lambda\tau$. So the expected number N_{as} of alerts from sampled temperature is:

$$N_{as} = P\{T > T_a\} \times \lambda\tau = (1 - F_T(T_a))\lambda\tau \tag{5.32}$$

The expected number of sampled temperature that is above reporting threshold is:

$$N_{rs} = P\{T > T_r\} \times \lambda\tau = (1 - F_T(T_r))\lambda\tau \tag{5.33}$$

These temperatures will be used for aggregation at least once before they decayed below the reporting threshold.

In order to find the expected number of alerts from aggregated values, we assume the exponential decay rate of value is μ,

5.2.3 SIMULATION RESULTS

As for Type 1 aggregator, we use similar situation to verify our theoretical conclusions. A wireless sensor network is deployed in the wild for monitoring fire through measuring temperature. It is supposed to aggregate the value of temperature at a central node, for simplicity, here the value of temperature is taken as the number value of temperature, for example, the value of 100°C is dimensionless 100 , and '+' is taken for aggregation, as proved in section 4.11 who is a Type 2 aggregator. A temperature is reported to the central base station only when it is in the critical temperature range [100, 500]°C. The temperature is modeled as a uniform random variable in the range [10, 450]°C. An alarm is triggered when the aggregated value is above 400. Here, $\Delta = 400$ and $E(X) = \frac{10+450}{2} = 230 < \Delta$, so aggregation is needed for reliable emergency report.

The wireless sensor network covers 100 km^2 area of forest with a density of 1 sensor per 8000 m^2, the area is evenly divided into 100 patches, each patch is installed with a central base station for information aggregation. All sensors in this wireless sensor network work asynchronously, when an emergency happened, usually from a single point, only those sensors close to this emergency locale can sense abnormal temperature and report it to their base station, due to the asynchronous working mode, transmission delay and collision, the base station will see the incoming reports as Poisson process [79], the reporting rate is supposed to be $0.2s^{-1}$, or once every 5 seconds. It is further assumed that the decay rate $\mu = 5 \times 10^{-3} s^{-1}$, i.e. the mean life-time of value is $200s$, or roughly 3 minutes. The reporting time is far shorter than the mean life-time of value.

In Figure 32, 50 points of temperature are collected, there are ten points below the report threshold, 2 points are above alert threshold, all others are between the two thresholds. Only the decays of those temperature in the critical temperature range are shown. If the

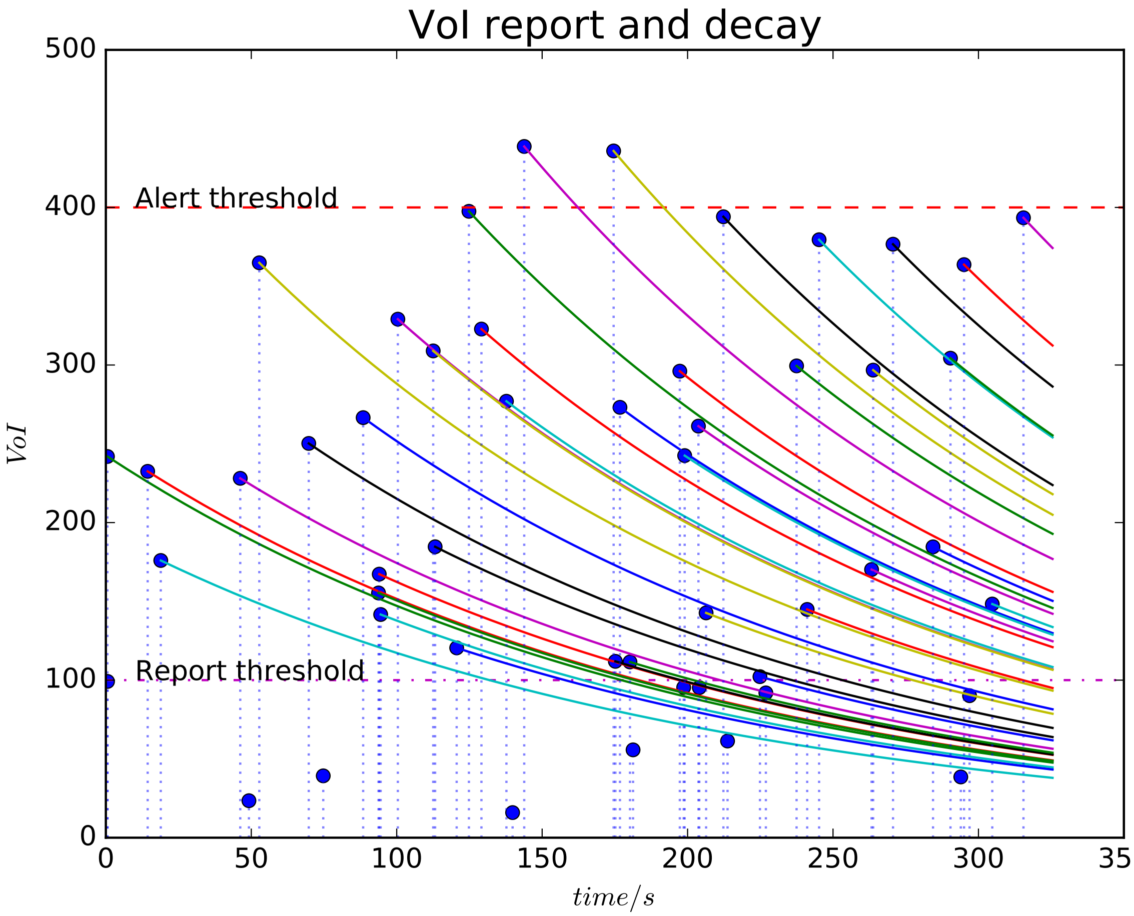

FIG. 32: Reporting and decaying of the value of temperature

only 2 points above alert threshold are missed, this emergency will be missed, for a complement, those values include decayed values above reporting threshold are aggregated to trigger more alters.

In Figure 33, the number of reporting temperatures that are above alert threshold without aggregation is 4 in 350 seconds, about 86 seconds, more than 1 minutes for each alert averagely. Interestingly, with aggregation, there are 13 alerts from aggregation, total 17 alerts, about 20 seconds for each alerts. This improved the reliability for alert greatly. However, some aggregated values go to above 1000, in some case, we may not need so frequent alert.

In Figure 34, the reporting rate is decreased to $0.01s^{-1}$, once for every 100 seconds. In 100 minutes, there are 3 alerts without aggregation, 23 alerts from aggregation, total 26 alerts, roughly 4 minutes for every alerts. This way benefits both the wireless sensor network and alert reaction time, less frequent alerts avoid lots of wireless transmission

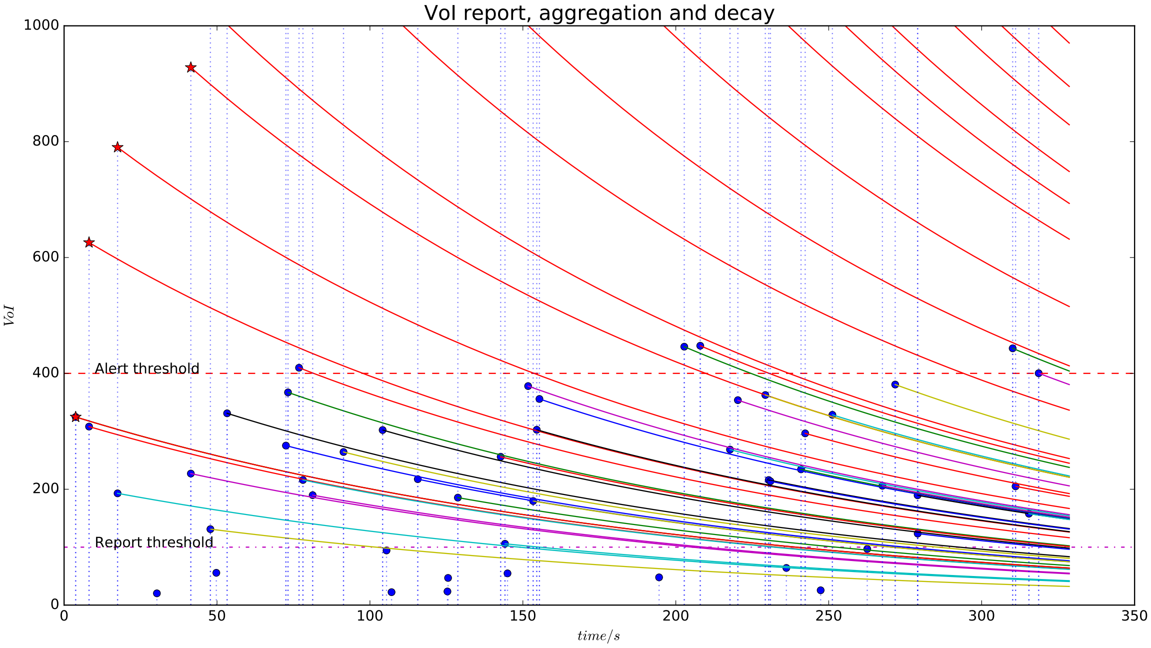

FIG. 33: Reporting , aggregation and decaying of the value of temperature with reporting rate $0.2s^{-1}$

to save energy, also make the alert period reasonably longer for related departments to intervene.

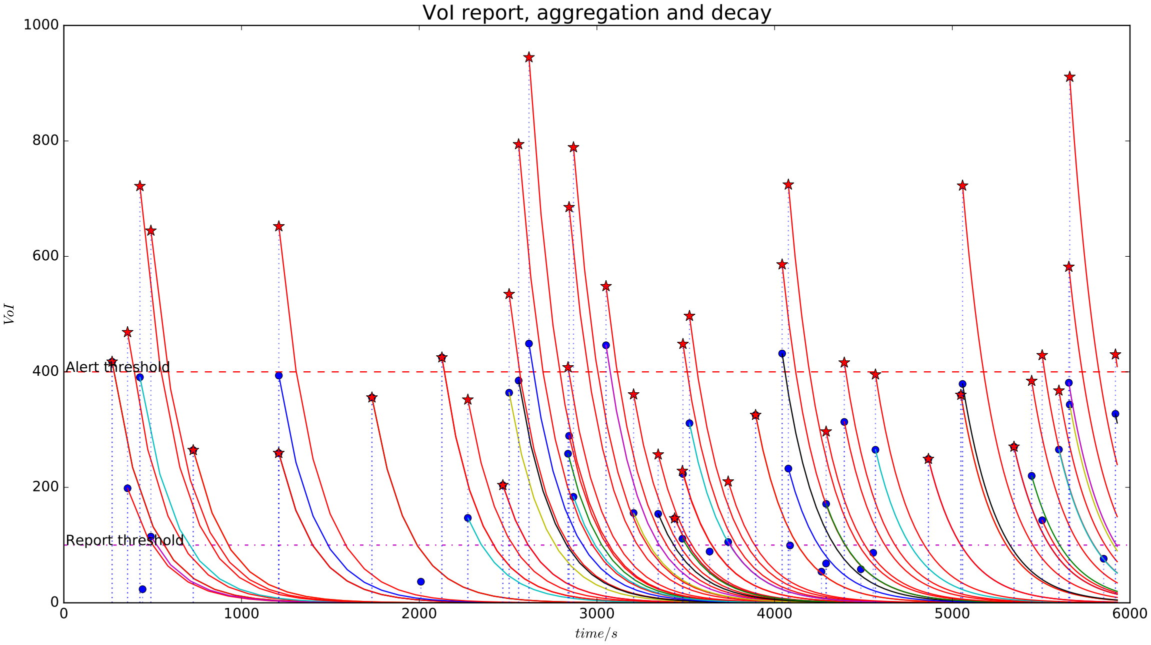

FIG. 34: Reporting , aggregation and decaying of the value of temperature with reporting rate $0.01s^{-1}$

CHAPTER 6

DATA AGGREGATION ACROSS GENERATIONS

6.1 INTRODUCTION AND MOTIVATION

Many phenomena last a long time, such as climate change in a region, global warming, continental drift, etc. However, many sensors can only last several weeks or perhaps months. Monitoring those long-term phenomena needs the cooperation of generations of sensors. The problem on how to aggregate and pass informations from generation to generation arises naturally.

In most of current deployed wireless sensor networks monitoring long-term pollutions in air, water, soil, extreme climate and wildfire, etc. Usually, they set alert on a single physical quantity, such as temperature for wildfire, precipitation for drought and flood. In our research, we will focus on the combination of relative quantities, which form a high dimensional space, apply machine learning algorithms to find patterns of emergency or potential accident, and make alert through prediction.

This idea comes from simple intuition: usually, the cause of a wildfire is a combination of many physical quantities, such as temperature, humidity, barometric pressure, wind direction and strength, lightning, etc. Their distribution in the wild forms many different patters, these patterns will be reflected in the corresponding high dimensional space. How to find these patterns and how to determine which pattern has higher possibility to cause a fire is our target. In such complex and highly correlated situation, analytical models are hard to find and set up. We will resort to machine learning to handle this problem and propose an implementation of a wireless sensor network based on cutting edge technologies. Machine learning and data mining have been used for wireless sensor network, meteorology, oceanography and climatology for a long time [8,28,55,61].

In a deployed wireless sensor network, every live sensor works continuously and it senses the environment periodically. The period is from several seconds to several minutes. So datasets are generated at high speed. Suppose the number of sensors is 10000 and the period of sensing is $5s$, and each sample contains 5 physical quantities, each quantity represented by a 4-byte floating-point real number, then the speed of data generation is

$5*4*10000/5 = 40000 bytes/s$, about $39KB/s$. One day, it will generate 3.2 Gigabytes, which is a large data set. Since data mining is a process of discovering patterns in large datasets through computation [35], this process extracts information from data sets and transform it into an comprehensible knowledge [32] for future usage, and work as a real-time on-line system. Being deployed in the wild field terrain, wireless sensor networks not only carry out spatial data ming to find spatial patterns [51] but also temporal data mining to recognize temporal patterns [58].

As a continuation of our previous project — ALERT [78], which can be re-tasked according field evolved requirements, we will enhance this system with machine learning [72] to re-task automatically by learning the evolution of the monitored objects. Equipped with algorithms from machine learning, the system can learn from the datasets on-line and make predictions for possible emergencies [29].These algorithms operate by building optimum mathematical models from sample inputs to make data-driven prediction or alert, instead of following strictly static programs, by learning from current and previous generations and transferring parameters of built models to future generations [36] to improve itself continuously.

6.2 A WHOLE SCENARIO

We assume our single hop WSN implemented on CC1310, a state-of-the-art wireless micro-controller from Texas Instruments, has been deployed and localized with location (longitude, altitude, elevation) by unmanned aerial vehicles (UAV) [95]. The WSN, through many generations of sensors, monitors F physical quantities $(x_1, x_2, \ldots, x_F)$ for a long term T_T. One typical example is to monitoring climate parameters such as temperature, barometric pressure, humidity, precipitation and wind speed, etc for potential disasters such as wildfire, tornado, drought and flood, etc. This WSN not only monitor short term weather dynamics, but also long term climate trends such as global warming, sea level elevation, etc. Climate satellites, weather stations and weather radars usually provide large scale weather information [39, 89, 96]. Meanwhile, WSN complements that with micro-climate in high resolution through highly dense distribution of sensors. Our algorithm is an implementation of the general machine learning clustering algorithm — *k-means* clustering, implemented with Distributed Random Grouping (DRG) algorithm [21], whose efficiency and convergence have been proved in [30]. Chen. J *et al.* [21] has given an expectation of the upper-bound of number of transitions for DRG. P.Sasikumar *et al.* [87] implemented both centralized and distributed k-means clustering algorithms in network simulator. However, both their implementations has fatal deficiencies:

- Every node knows the information about the positions and energies of all nodes in the WSNn, the communication and storage for these information incurred will render the WSN impractical.
- They did not discuss the convergence and dynamics of both algorithms in the context of WSN.

In our implementation, we not only solve the problems in [87], but also give firm theoretical foundation for the convergence and dynamics of k-means. As well as, We propose a algorithm to solve the inherent constraint of k-means whose cluster number needs to be specified and discuss its span for many generations of sensors. Finally, we implemented our version of k-means in both network simulator and a small scale real platforms of WSN.

The whole scenario includes the following stages:

- *Deployment*: When the WSN with N_0 sensors is deployed by a UAV to cover the monitored area uniformly, they are localized and synchronized by the UAV which is equipped with GPS. As the deployed WSN is covered within every sensor's transmission range, so their future synchronization is trivial, a single broadcast is good enough. Each sensor is identified by its location (x, y, z).
- *Periodic Sensing*: Every sensor senses the monitored F physical quantities X = (x_1, x_2, ..., x_m) at period of T_s when it is needed. The set of all X is called data space. After deployment, they started this periodic sensing to get to know its environment. This periodic sensing will continue until the sensor is shutdown. However, T_s will change according to demands from task and situation.
- *AoI identification*: In a huge monitored area, emergency occurs with different probability at different location. The area with highest probability is called hot spot, or area of interesting (AoI) in the deployed geometric space. After bootstrapping a certain time T_b, the WSN begins AoI identification through k-means algorithm with mini-batch mode if the scale of the WSN is fairly large. The number of clusters begins with a reasonable preset number K_0 depends on the size of the monitoring area A and the density D_0 of the distribution of sensors, with increase in the number of clusters, the area of AoI will decrease, until the AoI decreased to a manageable

area A_m which measured by the largest distance d_m of all sensor-to-sensor distances within this area. There may exist more than one AoIs as one cluster in data space may reside in several geometric space area.

- *Boundary and centroid designation*: After clusters are determined and mapped onto geometric space, the geometric patches of each cluster can determine their boundaries and centroids easily. The sensor with highest energy closest to the geometric centroid will be chosen as the centroid or AoI head.
- *Work Adaption*: After a manageable AoI is found, only sensors in the AoI continue their sensing at period of T_s, all other sensors will decrease their sensing period to γT_s, $\gamma \in [0,1)$ according to their distance to the AoI. The farer away, the slower they work. Some of them are even been put into low-power shutdown to save energy and work as further generations.
- *Alarm*: When the possibility of an emergency in an AoI turns above preset thresholds, alarms of corresponding level will be sent by the centroid to monitoring center for notification even call for rescue. As a multiple task system, each sensor can send alarms if their monitored quantities are above threshold when they are in any other stages. The possibility of an emergency is a function of the monitored physical quantities X, $p = f(X) = f(x_1, x_2, \ldots, x_m)$.
- *AoI re-identification*: Environment is always changing as a dynamic system, so is the AoI. The WSN will re-identify AoI after an application dependent period T_r. Which is another round of the stage of AoI identification.
- *Redeployment*: The continued deployment will exploit information (number of clusters, number of AoIs, boundaries of AoIs) from its previous generation in order to attain more specified and efficient monitoring. i.e. AoIs will be covered with sensors in higher density, other clusters are covered with lower density. Higher achievability can be reached in this way.

6.3 DATA PREPROCESSING

After time T_b, each sensor got $W = T_b/T_s$ data samples. Before applying the *k-Means clustering algorithm* on these data, data preprocessing is applied first, there are no universal preprocessing methods since machine learning is an adaptive technique. Different situations need different preprocessing methods. Common data preprocessing methods [35] include:

- Imputation by inferring and filling missing values;
- Standardization by removing mean and scaling variance;
- Normalization by scaling individual samples to unit norm;
- Binarization by thresholding numerical features to boolean values;
- Encoding by assigning codes to discrete or categorical features;
- Enrichment by generating nonlinear such as polynomial features, such as the ratio of temperature to humidity can be considered as an index for possibility of fire.

Imputation [35]

Wireless sensor networks are usually deployed in harsh environments, various faults can occur, such as power depletion, transmission collision, sensor malfunction, etc. These factors cause outliers — abnormal data, NaNs (Not a Number), placeholders even missing values to the collected raw datasets. On the other hand, when machine learning estimators work on the datasets, they require that all values be in predefined formats such as data frames, arrays and matrix are numerical, and that all have and hold reasonable meaning. A simple strategy to use incomplete and deficient datasets is to discard entire rows and/or columns containing NaNs, placeholders or missing values. However, the risk of this deleting strategy is losing data which may be valuable even critical. A better strategy is imputing [56] the missing values, in other words, to infer from the known part of the data. Since physical phenomenons develop continuously in micro-world in both spatial and temporal dimensions, so it is always a suitable strategy to use interpolation and extrapolation in space and time.

Common strategies for imputing missing values are either using the mean, the median or the most frequent value of the row or column in which the missing values are located. If consider the trends of data, interpolated or extrapolated data from their neighbors of the missing values can be calculated and used for spatial continuity. For a missing value X_m, how to choose its neighbors to get a good imputation to form better spatial continuity of dataset? It is clear that the more neighbor sensors are chosen for imputation, the more precise the inferred value will be, however, more energy, wireless transmission, memory and computing resource will be consumed, so an application-dependent optimum strategy is needed. In other words, how to choose a radius r to draw a circle centered at the sensor who has a missing value, use the values from those sensors inside this circle for imputation? Suppose the location vector of the sensor s_m with the missing value $X(m)$ is $\vec{r_{s_m}}$, S is the set of all sensors, the set N_b of s_m's neighbors is $Nb = \{s \mid \|\vec{r_{s_m}} - \vec{r_s}\| \leq r\}$, then

$$X_m = \frac{\sum_{s \in Nb} X(s)}{|Nb|}$$

For temporal continuity of dataset on a single sensor, interpolation or extrapolation are carried out on corresponding sensors' historic datasets where the missing values reside. Suppose in a sensor's historic dataset, there are N data $X_1, X_2, \ldots, X_N$, N is a relatively large number, missing values occur as rare event modeled by a Poisson process with rate λ. A typical value of this λ may be 1 missing value out of 1000 data. Then the expected number of missing value is λN. Then for a missing value X_m, how to choose the length W of the time window that centers this missing value? Since we need to make a trade-off between the goodness of the inferred value for this missing value and the cost of calculation for the imputation. Suppose $t(X)$ is the time when data X is collected, then the set of neighbor data with the time window of the missing value X_m is $Nb = \{X_i \mid \|t(X_i) - t(X_m)\| \leq \frac{W}{2}, i \in \{1, 2, \ldots, N\}\}$. We get similar temporal imputation equation for X_m:

$$X_m = \frac{\sum_{X \in Nb} X}{|Nb|}$$

Near the end of each generation of wireless sensors, most of them are malfunctioned, i.e. there are many more missing values than observed values, here, missing values are encoded by 0 and are thus implicitly stored in sparse matrix to save valuable memory resource.

Standardization

Standardization [33] of datasets is a preprocessing methods, it centers the dataset to its mean through removing the mean value of each feature, then scales it through dividing non-constant features by their standard deviation. Standardization simply ignore the shape of the distribution of the dataset, so they might behave badly if the individual features deviate far away from the standard normal distribution — Gaussian with zero mean and unit variance. We can use the following formulas to standardize N samples $X_1, X_2, \cdots, X_N$ into

$X_1^s, X_2^s, \cdots, X_N^s$, each sample X_i has m features, i.e. $X_i = (x_{i1}, x_{i2}, \ldots, x_{im})$, the standardization operation on sample X_i's k^{th} feature x_{ik} into x_{ik}^s is given by the equation below:

$$x_{ik}^s = \frac{x_{ik} - \mu_k}{\sigma_k} \qquad i \in \{1, 2, \cdots, N\},\ k \in \{1, 2, \cdots, m\}$$

where

$$\mu_k = \frac{\sum_{i=1}^{N} x_{ik}}{N}$$

and

$$\sigma_k = \sqrt{\frac{\sum_{i=1}^{N} (x_{ik} - \mu_k)^2}{N-1}}$$

Standardization is used in many machine learning algorithms. For example, many elements in the objective function of a learning algorithm, such as the RBF kernel of SVMs (Support Vector Machines) or the l1 and l2 regularizers of linear models, assume that all features are standard-normally distributed. If a feature has a variance that is orders of magnitude larger that others, it might dominate the objective function and make the estimator unable to learn from other features correctly as expected.

In order to be robust to very small standard deviations of features and preserving zero entries in sparse datasets, an alternative standardization is used. It scales features to lie between a given minimum and maximum value, often between zero and one, or so that the maximum absolute value of each feature is scaled to unit size. Suppose we have N samples $X_1, X_2, \cdots, X_N$, each sample X_i has m features, i.e. $X_i = (x_{i1}, x_{i2}, \ldots, x_{im})$, the scaling operation on sample X_i's k^{th} feature x_{ik} into $x_{ik}^{'}$ is given by the equation below:

$$x_{ik}^{'} = \frac{x_{ik} - min\{x_{1k}, x_{2k}, \cdots, x_{Nk}\}}{max\{x_{1k}, x_{2k}, \cdots, x_{Nk}\} - min\{x_{1k}, x_{2k}, \cdots, x_{Nk}\}}$$

Normalization

Normalization [88] is useful for kernel to quantify the similarity of pairs of samples such as dot-product in a quadratic form by scaling individual samples to unit norm. It is the base of the vector space model used in text classification and clustering contexts frequently. There are several typical types of normalization in machine learning (Suppose a sample X has m features, i.e. $X = (x_1, x_2, \ldots, x_m)$):

- l_1 *norm:* or Taxicab norm or Manhattan norm,

  $$\|\mathbf{x}\|=\sum_{k=1}^{m}|x_i|$$

  which is the sum of the absolute values of the features.

- *Euclidean norm:* is also called L^2 distance or L^2 norm, the length of sample X in the m-dimensional Euclidean space $\mathbf{R}^n$,

  $$\|\mathbf{x}\|=\sqrt{\sum_{k=1}^{m}x_i^2}$$

  which can be written as dot product or inner product in a concise form,

  $$\|\mathbf{x}\|=\sqrt{\mathbf{x}\cdot\mathbf{x}}$$

- *p-norm:* or generalized norm,

  $$\|\mathbf{x}\|_p=\left(\sum_{k=1}^{m}|x_i|^p\right)^{\frac{1}{p}}$$

  where $p \geq 1$. When $p = 1$, it turns into Manhattan norm; when $p = 2$, it is Euclidean norm; when $p \to \infty$, it turns into infinity norm or maximum norm.

  $$\|\mathbf{x}\|_\infty= max\{|x_1|,|x_2|,\cdots,|x_n|\}$$

6.4 AOI IDENTIFICATION

In an area monitored by wireless sensor network, the WSN senses the environment periodically, for any given time, it got an stereotype image of this area. Suppose there are N sensors, each one can sample m physical variables, or features. For instance, temperature, humidity, atmospheric pressure, wind speed, precipitation in this environment, i.e. 5 features. Then, for a given time, there are 5 images super-positioned on this area, the resolution of these images are determined by the density of sensors deployed in this area. When time goes by, frames of images are generated, which will show the dynamical evolution of this area. Patterns not only hide in images of single feature, but also in the composite images. In such a general composite image, each pixel is a sample, the depth of this pixel is the number features, each feature is a color channel, some intricate patterns hide inside. For unknown patterns, unsupervised learning [37] such as clustering can be applied to find them; for known patterns, we use supervised learning [23, 49, 97] such as classification. In the view of image and video, techniques from computer vision [20] can be applied.

Clustering [37]

Clustering is a group of algorithms for unsupervised learning used form find and describe hidden structure from unlabeled data by clustering samples together according their similarities. Number of clusters and the measurement of similarity between two samples are two most important parameters for a clustering algorithm. Different clustering algorithms have different scalability with the size of input samples, suit different situations. Common clustering algorithms include:

- *K-Means* is a general-purpose clustering algorithm, uses distance between points to measure similarity, can be scaled to very large dataset, suits cases such as few clusters, even cluster size and flat geometry.

- *Affinity propagation* uses graph distance to measure similarity. It is not scalable with samples, suits cases such as many clusters, non-flat geometry and uneven cluster size. Its parameters are damping and sample preference.

- *Mean-shift* is similar to affinity propagation except using distance between points for similarity and taking bandwidth as parameter.

- *Spectral clustering* use the same similarity measurement as affinity propagation with number of clusters as parameter. It suits cases such as few and even clusters, non-flat geometry.

- *Self-organizing maps* uses neural networks to learn the distribution and topology of the data. It can produce a low-dimensional representation of the composite image from the WSN to get synthesized information. With small number of nodes, self-organizing map works similarly to K-means.

- *Ward hierarchical clustering* suits large number of samples, many clusters, possibly connectivity constraints. It uses distance between points for similarity and takes number of clusters as parameter. The output is a multilevel hierarchy of clusters in the a cluster tree.

- *Agglomerative clustering* uses any pairwise distance for similarity, suits many clusters, non-Euclidean distances and possibly connectivity constraints, takes number of clusters and linkage type as parameters.

- *DBSCAN* takes neighborhood size as parameter and distances between nearest points for similarity. It suits very large number of samples and medium number of clusters.

- *Gaussian mixtures* has many parameters, it is not scalable, uses Mahalanobis distance to centers for similarity, suits cases such as flat memory and density estimation. If each feature has equal covariance then Gaussian mixtures degenerated to K-Means.

- *Hidden Markov models [90]* uses historical data to recover the sequences of states. Which could be used to find the incubation and evolution of emergencies.

In this paper, we implemented a distributed and customized version of K-Mean to find hidden pattern from the collected data space. These patterns reflect the probability that an incident will occur imminently or in the future. Through ordering the probabilities of these patterns, hot spots — those areas have patterns with highest incidental probability, can be identified and then monitored intensively. Lots of energy will be saved when such knowledge is transfered from generation to generation.

Classification and regression [23,49,97]

At the time when the first generation of sensors are deployed, they has zero knowledge about the environment, with time goes by, more and more data are collected, through clustering, patterns are discovered gradually and continuously, knowledge are accumulated steadily. With historical data especially knowledge learned from previous generations, later generations can train themselves through supervised learning such as classification and regression, and become more sensitive, reliable and efficient to predicting incidents and trigger alarms.

As physical phenomenons develop and evolve gradually and continuously, the probability p of incidents can be considered as an dependent variable that depends on those monitored features $X = (x_1, x_2, \ldots, x_m)$ which can be taken as independent variables. Then a regression model relates p to a function — the regression model f of X:

$$E[p|X] = f(X,\boldsymbol{\beta})$$

where **fi** is unknown parameters, it may be a scalar for a single parameter or a vector for multiple parameters. The form of the function f must be specified to carry out regression analysis. For a simple and approximate model such as general linear model, suppose the WSN has collected N samples for regression analysis,

$$p_i = \beta_0 + \sum_{k=1}^{m} \beta_i x_i k + \varepsilon^i \qquad i \in \{1,2,\ldots,N\}$$

where $x_i k$ is the i^{th} samples on the k^{th} feature, β_0 is the regression intercept. The residual can be written as

$$\varepsilon^i = p_i - \hat{\beta}_0 - \sum_{k=1}^{m} \hat{\beta}_i x_{ik}$$

The least squares parameter estimates are computed from m normal equations:

$$\sum_{i=1}^{N}\sum_{k=1}^{m} x_{ij} x_{ik} \hat{\beta}_k = \sum_{i=1}^{N} x_{ij} p_i, \qquad j = 1,\ldots,m$$

Rewrite the normal equations into matrix form

$$\left(\mathbf{X}^T\mathbf{X}\right)\hat{\boldsymbol{\beta}} = \mathbf{X}^T\mathbf{P}$$

where the ij^{th} element of $\mathbf{X}$ is x_{ij}, the i_{th} element of the column vector P is p_i, and the j^{th} element of $\hat{\boldsymbol{\beta}}$ is β_j. Thus the size of matrix $\mathbf{X}$ is $N \times m$, P is $N \times 1$, and $\hat{\boldsymbol{\beta}}$ is $m \times 1$. The solution for parameter $\hat{\boldsymbol{\beta}}$ is

$$\hat{\boldsymbol{\beta}} = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{P}$$

for the objective function

$$\min_{\boldsymbol{\beta}} ||\mathbf{X}\boldsymbol{\beta} - \mathbf{P}||_2^2$$

However, in the above ordinary least squares, coefficient estimates rely on the independence of the model features. When features are correlated and the columns of the design matrix $\mathbf{X}$ have an approximate linear dependence, the design matrix becomes close to singular and render the least-squares estimate becomes highly sensitive to random errors in

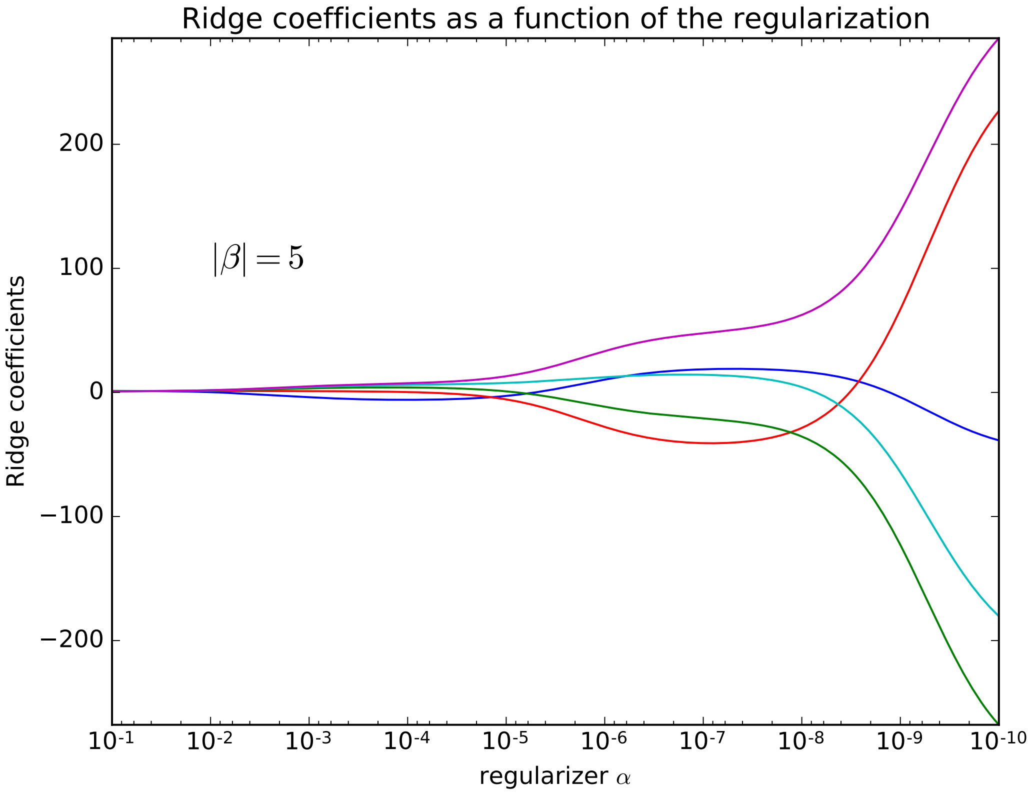

FIG. 35: Ridge coefficients as a function of the regularization

the collected dataset, producing a large variance. In real environment, physical quantities such as humidity and precipitation are usually highly correlated, this will render the ordinary least squares ill-functioned.

Ridge regression [40] addresses some problems of ordinary least squares by imposing a penalty on the magnitude of coefficient vector. The ridge coefficients minimize a penalized residual sum of squares,

$$\min_{\boldsymbol{\beta}} \|\mathbf{X}\boldsymbol{\beta} - \mathbf{P}\|_2^2 + \alpha \|\boldsymbol{\beta}\|_2^2 \tag{6.1}$$

Here, $\alpha \geq 0$ is a complexity parameter that controls the amount of $\boldsymbol{\beta}$ shrinkage: the larger the value of α, the greater amount of $\boldsymbol{\beta}$ shrinkage and thus the coefficients become more robust to collinearity illustrated in Figure 35.

In Figure 35, there is no intercept term for $\boldsymbol{\beta}$ and its length is 5, each color represents a different feature of the coefficient vector whose components are displayed as functions of the regularization parameter α. When α goes to zero, equation (6.1) goes to the ordinary least squares, and coefficients exhibit dramatic oscillations. Through cross-validation, a suitable regularizer can be found.

In wireless sensor network, the computing resource is limited. The lasso [93] is such a linear model for selecting features by estimating sparse coefficients. It has a l_1 norm as its regularizer:

$$\min_{\boldsymbol{\beta}} \frac{1}{2N} \|\mathbf{X}\boldsymbol{\beta} - \mathbf{P}\|_2^2 + \alpha|\boldsymbol{\beta}| \tag{6.2}$$

The lasso is useful in wireless sensor network due to its tendency preferring solutions with fewer parameter values which effectively reduces the number of features upon which the given solution depends. For this advantage, the lasso and its variants are widely used in compressed sensing and feature selection. Under certain conditions, it can recover the exact set of non-zero coefficients.

The lasso can be combined with the ridge to construct a synthesized linear regression model — the elastic-net [26] by controlling their convex combination of L1 from the lasso and L2 from the ridge using the $l1_ratio$ parameter. A model trained through this synthesis has both advantages such as few of the coefficients are non-zero like the lasso and advantages such as maintaining the regularization properties of the ridge. The elastic-net inherit some of the ridge's stability under rotation through the trading-off between the lasso and the ridge. When there are multiple features that are correlated mutually, the lasso is likely to pick one of them randomly while the elastic-net is likely to pick both. The synthesized objective function to minimize in the elastic-net is:

$$\min_{\boldsymbol{\beta}} \frac{1}{2N} \|\mathbf{X}\boldsymbol{\beta} - \mathbf{P}\|_2^2 + \alpha|\boldsymbol{\beta}| + \frac{\alpha(1-\rho)}{2} \|\boldsymbol{\beta}\|_2^2 \tag{6.3}$$

where ρ is the $l1_ratio$.

As we can see, using the lasso, the ridge and the elastic-net linear regressions, the probability of emergency can be predicted with fewer features, which make them suitable to wireless sensor network. Other regression algorithms [34], such as

- Nonlinear regression;
- Support vector machine regression;
- Gaussian process regression;
- Regression trees;
- Regression tress ensembles.

Their feasibility, utility and efficiency will be put into our future research. Once a fire happened in a patch, then the cluster to which the patch belongs is labeled as the area of interest, whose combination of features has the highest probability that there a fire will occur, and the cluster id is given 0; those clusters that are adjacent to cluster 0 are merged and labeled 1; those clusters that are adjacent to cluster 1 other than cluster 0 are labeled 2; and so on. Until now, all the clusters gotten through clustering got their first knowledgeable ids, and the number of clusters are reduced, new centroids are recalculated, training dataset are generated for the exploitation of future generations. The probability of a fire is then discretized according to the number of clusters, the lowest cluster id, the highest probability. How to select the most representative and sufficient training datasets to transfer to future generations is another challenge.

With training datasets, current and later generations can train themselves to create prediction models and test these models with datasets collected in the future. There are many classification algorithms [52] such as:

- *Linear classifiers:*
  - Fisher's linear discriminant;
  - Logistic regression.
- *Quadratic discriminant analysis*
- *Logic-based algorithms:*
  - Decision tree;
  - Random forests;
  - Rule-based classifier.
- *Perceptron-based techniques:*
  - Perceptron;
  - Neural networks;
  - FMM neural networks;
  - Deep convolution neural networks.
- *Statistical learning algorithm:*

  - Bayesian networks;
  - Naive Bayes classifier;
  - Instance-based learning;
  - Nearest neighbors classification;
  - Stochastic gradient descent.

- *Support vector machine:*
  - Least squares support vector machines.

- *Ensemble methods.*

We prefer support vector machine (SVM) [24] for our classification tasks because of its many advantages:

- It is a set of supervised learning algorithms applied for not only classification, but also regression and outliers detection.
- It is effective in high dimensional spaces even the number of dimensions is larger than the number of samples.
- It uses a subset of training samples for support vectors hence it is memory efficient.
- With different kernels for the decision function, it can be adapted to many situations. Typical kernel functions include:
  - linear $< x, x' >$.
  - polynomial: $(\gamma < x, x' > +r)^d$.
  - RBF: $\exp(-\gamma|x - x'|^2)$. where $\gamma > 0$.
  - sigmoid: $(\tanh(\gamma < x, x' > +r))$.

The disadvantages of SVM include:

- It likely performs poorly if the number of features is much larger than the number of samples. However, in our alert system, datasets are an abundant resource, so this disadvantage can be ignored.

- It has not probability estimates, even though these can be calculated through an expensive five-fold cross-validation. As we do not need strict segmentation between the clusters and patches, i.e. we do not need probability estimates for classification of sensors belonging to which cluster. So this disadvantage does not matter in our situation.

Increasing the number of clusters can shrink the area of interest to make the monitor more focused and efficient. Multiple clusters means sensors in later generations need multi-class SVM to assign cluster ids to sensors by using support vector machines. This single multi-class problem can be converted into multiple binary classification problems. Common methods for such conversion include [42]:

- *One-versus-all:* Binary SVMs are built to distinguish one class from all others. Classification of new samples for the one-versus-all case is done by a winner-takes-all strategy, in which the classifier with the highest output function assigns the class (it is important that the output functions be calibrated to produce comparable scores).

- *One-versus-one:* Binary SVMs are built between every pair of classes. Classification is done by a max-wins voting strategy, in which every classifier assigns the sample to one of the two classes, then the vote for the assigned class is increased by one vote, and finally the class with the most votes determines the sample classification.

- *Decision directed acyclic graph (DDAG) SVM [83]:* The decision directed acyclic graph is used to combine many two-class classifiers into a multi-class classifier. The DDAG constructs $N(N-1)/2$ classifiers, one for each pair of classes in an N-class problem.

In the data space which is m dimensional, a support vector machine constructs a hyper-plane or set of hyper-planes to separate samples with different cluster id. A good separation is attained by the hyper-plane that has the largest distance to the nearest training sample data points of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier. For two clusters, suppose $X_i \in \mathbb{R}^m$, $i = 1,2,\ldots,N$ are samples collected by the former generations, and a vector of cluster ids $c \in \{1,-1\}^N$, the SVC solves the following primal problem:

$$\min_{w,b,\zeta} \frac{1}{2} w^T w + C \sum_{i=1}^{N} \zeta_i$$

$$\text{subject to } c_i(w^T\phi(X_i)+b) \geq 1-\zeta_i$$

where $\zeta_i \geq 0,\ i = 1,2,\ldots,N$. The dual form is:

$$\min_{\alpha} \frac{1}{2}\alpha^T Q\alpha - e^T\alpha$$

$$\text{subject to } c^T\alpha = 0$$

where $0 \leq \alpha_i \leq C,\ i = 1,2,\ldots,N$. e is the vector of all ones, $C > 0$ is the upper bound, Q is an $N \times N$ positive semidefinite matrix, $Q_{ij} \equiv c_i c_j K(X_i,X_j)$ in which $K(X_i,X_j) = \phi(X_i)^T\phi(X_j)$ is the kernel. The function ϕ determine the map from data space to a higher dimensional space.

The decision function is:

$$sgn\left(\sum_{i=1}^{N} c_i\alpha_i K(X_i,X)+\rho\right)$$

Reinforcement learning

Inspired by optimal adaptive control theory, reinforcement learning [17–19] constructs a feedback for the strategies or behaviors from their effects, loss or rewards when an agent acts in the environment. Here, the wireless sensor network is deployed in the environment, however, when an incident is predicted with a high probability that surpass preset threshold for alert, then an alert is triggered, and rescue teams come to the locale and clear the emergency, meanwhile, people around the spot are also evacuated. If consider the WSN, rescue teams and people around as agents. The actions are sensing, predicting and triggering alarm from sensors, deployment, movement and rescue from the rescue team, evacuation from people. Then how to minimize the loss becomes a multi-agent reinforcement learning problem.

In this paper, we only consider the wireless sensor network. The sensors sense the environment passively. They almost have no effect to this world. They are considered as the only agents. The actions they can take is:

- *Deep sleep:* A sensor turns into sleep to save energy;
- *Wake up:* A sensor can be waken up by external events from the environment such as received messages from other sensors or internal events such as a timer terminated;
- *Turn on/off a specified sensing capability:* such as temperature, humidity, etc.

- *Send/receive message* to/from other sensors;
- *Save/delete* sensed data for further processing/recovering memory space;
- *Computation:* A sensor carries out computing operations on datasets with certain algorithms for respective tasks.
- *Trigger alarm.* When an incident is predicted even detected, the sensor triggers an alarm.

The loss is the total energy of all sensors. The reward is the accuracy of predictions for incidents, the loss is energy depletion from sensors. Now we model the energy state $E(t)$ of the WSN as a stochastic finite state machine with inputs $A(t)$ (actions listed above carried out by sensors) and outputs $Y(t)$ (accuracy of predictions for emergencies and left energy of sensors):

- State transition function $F_T(E(t), E(t-1), A(t)) = Pr\{E(t)|E(t-1), A(t)\}$;
- Observation (output) function $F_O(E(t), A(t)) = Pr\{Y(t)|E(t), A(t)\}$;
- Reword function $F_R(t) = Pr\{R(t)|E(t), A(t)\}$.

A policy is a mapping from states to actions, reinforcement learning seeks to learn a policy that maximize the accuracy of predictions for emergencies and left energy of sensors in the long run for many generations. This research direction on reinforcement learning of WSN predicting emergencies is put into our future research.

Algorithm for AoI identification

Suppose there are N sensors in the WSN, after they got preprocessed data, the algorithm of AoI identification is carried out to find the hot spots.

```
Algorithm 4: AoI identification
  Input  :
        $X$ — the set of $N$ preprocessed data, each one resides in a sensor
        $S$ — the set of all sensors in the WSN
        $K_0$ — initial cluster number
        $d_b$ — diameter of circles for boundary determination
        $d_m$ — diameter of manageable area
  Output:
        $K$ — number of centroids
        $C$ — set of centroids
        $P$ — set of patches of sensors
        $D_A$ — set of diameters of AoIs
1 $K \leftarrow K_0$,    $d_A \leftarrow \infty$
2 $C \leftarrow InitCentroid(X, K)$
3 while $\max\{D_A\} > d_m$ do
4     $C \leftarrow$ k-Means$(X, K, C)$
5     $P \leftarrow Cluster2Patch(C, S, d_b)$
6     $D_A \leftarrow AoI(P)$
7     $K \leftarrow K + 1$
8 end
```

Algorithm 4 has four major operations:

- Line 2 in centralized version of *k-Means*, i.e, a single node has all data, carries out *k-Means* several times by choosing the initial centroids randomly in order to alleviate the probability that *k-Means* is trapped inside a local optimization. However, in our distributed version which is more suitable to WSN, algorithm *InitCentroid* is used to find the initial set of centroids.

- In line 4, we implemented a customized and distributed version of *k-Means* clustering algorithm to find the set of centroids. It is described in algorithm *k-Means*. After this step, each sensor got its cluster id from the set of $\{1, 2, \cdots, K\}$.

- In line 5, all clusters use *Cluster2Patch* to determine their patches.

- In line 6, the cluster with the highest incident probability, i.e. whose patches are hot spots, determines the maximum diameter $\max\{D_A\}$ of its patches with algorithm *AoI*.
- In line 7, the number of patches is increased, the while loop continues until the sizes of all hot spots decreased to a manageable size d_m (line 3).

In algorithm 5 *InitCentroid*, the sensors sequentially find the initial set of centroids $\{c_1, c_2, \cdots, c_K\}$. In the initial step, all sensors will broadcast to declare as c_1 with probability p_1, only the first one works as c_1, all others will receive and store its id, data and location, then step for c_2 begins, only those sensors far away from c_1 in both space will compete for c_2, the process continues as well until all K centroids are determined. In each step, if p_i is large then collision will be highly probable, otherwise, it will take longer time for the first sensor to broadcast its message, so there will exist an situation-dependent optimum p_i for each step.

Algorithm 5: Determining the initial set of centroids

```
Input  :
        $X$ — the set of $N$ preprocessed data, each one resides in a sensor
        $K$ — number of centroids
        $\{p_1, p_2, \cdots, p_K\}$ — each sensor's broadcasting probability for centroid
Output:
        $C$ — set of centroids
$i \leftarrow 1$
while $i \leq K$ do
    Each sensor broadcast itself as $c_i$ with probability $p_i$
    Only the first one will be accepted by all sensors
    $C \leftarrow C \cup \{c_i\}$
    $i \leftarrow i + 1$
end
```

K-Means [43] clustering algorithm is discovered almost half-century ago since then it has been used across a large range of application areas in many different fields. By minimizing a criterion known as the inertia or within-cluster sum-of-squares, it separates samples in K groups of equal variance in which K is the parameter needs to be specified. This algorithm scales well to large number of samples in a dataset.

The k-means algorithm separates a set of N samples $X = \{X_n,\ n = 1, 2, \ldots, N\}$ into K disjoint clusters $C = \{C_k, k = 1, 2, \ldots, K\}$, each cluster C_k is represented by the mean

μ_k of its consisted samples. The means are commonly called the cluster centroids which is similar to the centroid of a geometric shape consists of points; in general they are not necessarily points from X, although they reside in the same space. The K-means algorithm choose centroids by minimizing the inertia, or within-cluster sum of squared criterion. The squared error between μ_k and the points in cluster C_k is defined as

$$J(C_k) = \sum_{X_n \in C_k} \|X_n - \mu_k\|^2$$

. The goal of k-means is to minimize the sum of the squared error $J(C)$ of all K clusters.

$$J(C) = \sum_{k=1}^{K} \sum_{X_n \in C_k} \|X_n - \mu_k\|^2 \tag{6.4}$$

The within-cluster sum of squares criterion, or intuitively, inertia, can be recognized as a measure of how internally coherent each cluster is. It suffers from various drawbacks:

- Inertia assumes that clusters are convex and isotropic, which is not always the case. It responds poorly to elongated clusters, or manifolds with irregular shapes.

- Inertia is not a normalized metric so lower values are better and zero is optimal. However, in very high-dimensional spaces, Euclidean distances tend to become inflated (this is an instance of the so-called curse of dimensionality). Running a dimensionality reduction algorithm such as PCA prior to k-means clustering can alleviate this problem and speed up the computations.

Minimizing $J(C)$ defined in equation (6.4) is known to be an NP-hard problem. Though recent study has shown with a large probability K-means could converge to the global optimum when clusters are well separated, however, as a greedy algorithm, it can only converge to a local minimum. K-means starts with a random choose of K centroids from N samples and calculate new centroids iteratively. Since in every iteration the squared error always decreases, so it will converge within finite number of iterations.

In algorithm 6, the customized and distributed *k-Means*, it will find the new set of centroids from the initial set of centroids. Here, centroid refers to both the centroid of a cluster in the data space and the sensor that works as this centroid. The algorithm has a major loop, line 3, when the maximum of movements of centroids are less than a given error, the loop finished. The major loop contains three sub-loops:

- In line 4, centroid with lower id will broadcast before the higher through that a centroid begins broadcasting its data only after it received all data from centroids with cluster ids lower than itself. Each sensor will have all the centroids when this loop finished.

- In line 8, each sensor calculates its similarity to each centroid in data space and sets its cluster *id* the same as the most similar centroid. Here, we use Euclidean distance in the data space as the measure of similarity between data. It only considers moved centroids to save energy.

- In line 12, each centroid collects data from its cluster in the order of cluster *id* to alleviate collisions. When the turn for a centroid comes, it broadcasts a very short message to begin collection of data, the sensors in its cluster send data back to this centroid with a probability to alleviate collisions further. After a certain timeout if the centroid still receives nothing, it considers all sensors in its cluster have responded with data, then it broadcasts a very short finishing message so that the next centroid can begin collection. After data collection, each centroid calculates the arithmetic mean of collected data to get its new centroid and broadcast this result and the movement from its current centroid, after all new centroids and movements are collected, the maximum movement can be determined and be used to decide whether terminate the major loop. During this sub-loop, every sensor broadcast its information several times to all other sensors, each sensor also save its geometric neighbors for algorithm *Cluster2Patch*. Here, neighbors of a sensor are those sensors within a application-dependent geometric distance d_b to this sensor.

When the final centroids are found, as each centroid already has all information about its cluster, then it will choose a new sensor works as the new centroid along with several backups shown in line 20.

```
Algorithm 6: A customized and distributed k-Means
   Input  :
            $X_i,\ i \in \{1,2,\cdots,N\}$ — data set, each one resides in a sensor
            $C_0$ — initial set of $K$ centroids
            $\varepsilon$ — maximum centroid movement error in data space
   Output:
            Each sensor gets its cluster id
            $\max ||C_n - C_{n+1}|| < \varepsilon$
 1 $n \leftarrow 0,\ d \leftarrow \infty$
 2 ($n$ is round of k-Means algorithm, $d$ is the maximum centroid movement.)
 3 repeat
 4     foreach centroid $c$ in $C_n$ do
 5         $c$ broadcasts its data $X_c$ to the WSN
 6         Each sensor in the WSN stores $X_c$
 7     end
 8     foreach sensor $s$ (include centroids) in the WSN that collected $K$ $X_c$ do
 9         Calculate its similarity to each centroid in data space
10         Sets its cluster id the same as the closest centroid
11     end
12     foreach centroid $c$ in $C_n$ do
13         Collects all samples from its cluster
14         Calculate the new centroid $C_{n+1}$
15         $d = \max ||C_n - C_{n+1}||$
16         $C_n \leftarrow C_{n+1}$
17         $n \leftarrow n+1$
18     end
19 until $d < \varepsilon$;
20 Select closest senors work as final centroids
```

In algorithm *Cluster2Patch*, as each sensor already got its geometric neighbors in algorithm 6, if all its neighbors have the same cluster id as itself, then it declares as a inner sensor otherwise boundary sensor. However, those sensors at the outermost of the WSN also has a high probability to be declared as inner sensor, which is a special case.

In algorithm *AoI*, as each centroid has all the informations about other centroids, so it can calculate the incident probabilities with $p = f(X)$ for all centroids and sort them. The centroid with the highest incident probability declares all its patches as AoI of the WSN.

6.5 GENERATIONAL LEARNING

Consider a typical generational learning scenario:

- For a long term mission, all sensors of a WSN are distributed sleep at the beginning, each generation are programmed to wake up at their working time, for example, each generation work for one month, the first generation wake up right after deployed for one month, then the second generation wake up and inherit the knowledge from its previous generation. This process continues until the mission is complete. When the last generation come to their time, their data are collected before they stop working.

  A second way is to deploy sensors generation by generation. For example, a drone is used to deploy sensors, after one month, the drone will go out again to deploy the second generation and so on. Meanwhile, it will collect all the knowledge from the previous generation. After the second generation deployed and before they beginning to work, some learning skills and knowledges are inherited from their previous generation.

  Series of questions aroused:

  1. What kind of learning skills and knowledge are transferred from generation to generation?
  2. How to pass on knowledge collected by current generation to next generation?

- Usually, when sensors are sleep all the time, their lifespan is expected to be very long, say 1 year; when they are operating, their lifespan is likely to be very short, say 1 month. Under the constraint of keeping temporal and spatial coverage probability above some threshold of the WSN, How many generations does this WSN provide?

- Suppose each sensor is equipped with K bytes, how many bytes of information can they collected at the end?

- A wireless sensor network is randomly uniformly distributed across a monitored region with density of λ, then for any sub-region with area A in this region, the number of sensors $N(A)$ is a two dimensional *Poisson point process*:

$$P(N(A)=k)=\frac{(\lambda A)^k e^{-\lambda A}}{k!} \qquad k\in\mathbb{N}$$

- Connectivity requirement: for any sensor in the WSN, within its minimum or guaranteed transmission range, the probability that there exist at least one neighbor is above a preset threshold.

$$\begin{aligned} P(N(A)\geq 1) &= 1-P(N(A)=0) \\ &= 1-e^{-\lambda A} \\ &= 1-e^{-\lambda\pi[\mu(1-\nu)]^2} \end{aligned} \tag{6.5}$$

- Task achievability: For a long term task that lasts for T, the sensors are under previous assumption, what's probability that the task will be complete?

6.6 SURVIVAL PROBABILITY OF ONE GENERATION

Definition 6.6.1. *A partition of positive integer N into K parts is a tuple* $(n_1,n_2,\cdots,n_K)$ *under the constraints:*

1. $N\geq K$ *while* $N,K\in\mathbb{Z}^+$.
2. $\sum_{i=1}^{K} n_i = N$ *while* $n_i\in\mathbb{Z}^+$, $i=1,2,\cdots,K$.

Suppose

- The monitored field is divided into K patches and N sensors are deployed in total. In the i^{th} patch, there are $n_i\geq 1$ sensors, so $\sum_{i=1}^{K} n_i = N$, i.e. each allocation of N sensors among K patches is an integer partition of N into K parts.
- Each sensor's lifespan T is an identical independent distributed random variable with cumulative distribution function $F(t)$:

$$F(t)=Pr\{T\leq t\}.$$

- For the knowledge from i^{th} patch to be collected with a given generation, there should be at least one sensor survive the generation time, or redeploy cycle T_r, denote this event as E_{n_i}, if we denote the event that a single sensor S_m survives T_r as E_{S_m}, then $E_{n_i} = \cup_{m=1}^{n_i} E_{S_m}$.

Under such assumption, we have

$$\begin{aligned} Pr\{E_{S_m}\} &= Pr\{t > T_r\} = 1 - Pr\{t \leq T_r\} \\ &= 1 - F(T_r) \qquad (6.6) \\ Pr\{E_{n_i}\} &= Pr\{\cup_{m=1}^{n_i} E_{S_m}\} \\ &= 1 - F^{n_i}(T_r). \qquad (6.7) \end{aligned}$$

Equation (6.7) can be proved with mathematical induction.

Proof. Prove by mathematical induction on n_i.

Basis step: $n_i = 1$, by Equation (6.6), it holds.

Inductive step: Suppose $n_i = k$, $Pr\{E_k\} = 1 - F^k(T_r)$ holds. We need to prove $n_i = k+1$, $Pr\{E_{k+1}\} = 1 - F^{k+1}(T_r)$.

$$\begin{aligned} Pr\{E_{k+1}\} &= Pr\{\cup_{m=1}^{k+1} E_{S_m}\} = Pr\{(\cup_{m=1}^{k} E_{S_m}) \cup E_{S_{k+1}}\} \\ &= Pr\{(\cup_{m=1}^{k} E_{S_m})\} + Pr\{E_{S_{k+1}}\} - Pr\{(\cup_{m=1}^{k} E_{S_m})\} Pr\{E_{S_{k+1}}\} \\ &\quad (\because \quad \text{Each sensor survives independently}) \\ &= 1 - F^k(T_r) + 1 - F(T_r) - (1 - F^k(T_r))(1 - F(T_r)) \\ &= (1 - F^k(T_r))F(T_r) + (1 - F(T_r)) \\ &= (1 - F(T_r))(\sum_{j=1}^{k} F^j(Tr)) + (1 - F(T_r)) \\ &= (1 - F(T_r))(\sum_{j=0}^{k} F^j(T_r)) \\ &= 1 - F^{k+1}(T_r). \end{aligned}$$

□

For the event E_{WSN} that the WSN in the whole field survive T_r, i.e. each patch has at least one sensors survive T_r, as sensors in each patch survive independently, so $E_{WSN} = \cap_{i=1}^{K} E_{n_i}$:

$$Pr\{E_{WSN}\} = Pr\{\cap_{i=1}^{K} E_{n_i}\} = \prod_{i=1}^{K} [1 - F^{n_i}(T_r)]. \qquad (6.8)$$

Theorem 6.6.1. *Within all the integer partitions of N, the most even and unique partition $(\lfloor \frac{N}{K} \rfloor, \lfloor \frac{N+1}{K} \rfloor, \cdots, \lfloor \frac{N+K-1}{K} \rfloor)$ survives the longest.*

$$\prod_{i=1}^{K} \left[1 - F^{\lfloor \frac{N+i-1}{K} \rfloor}(T_r)\right] \geq \prod_{i=1}^{K} \left[1 - F^{n_i}(T_r)\right]. \tag{6.9}$$

We need three **Lemmas** to prove **Theorem 6.6.1**.

Lemma 6.6.2. *For positive integers, a, b, $(0 < a < b)$, and $b - a \geq 2$, $(b-1)(a+1) > ab$.*

Proof.

$$\begin{aligned} (b-1)(a+1) - ab &= ab + (b-a) - 1 - ab \\ &= (b-a) - 1 \geq 1 \qquad (\because\ b - a \geq 2) \end{aligned}$$

□

Lemma 6.6.3. *For positive integers, a, b, $(0 < a < b,\ b - a \geq 2)$, and real number F, $(0 < F < 1)$, $(1 - F^{b-1})(1 - F^{a+1}) > (1 - F^{a})(1 - F^{b})$.*

Proof.

$$\begin{aligned} &(1 - F^{b-1})(1 - F^{a+1}) - (1 - F^{a})(1 - F^{b}) \\ &= (1 - F^{a+1} - F^{b-1} + F^{a+b}) - (1 - F^{a} - F^{b} + F^{a+b}) \\ &= F^{a} + F^{b} - F^{a+1} - F^{b-1} \\ &= (1 - F)(F^{a} - F^{b-1}) > 0 \qquad (\because\ b - a \geq 2 \text{ and } 0 < F < 1). \end{aligned}$$

□

Lemma 6.6.4. *For the most even partition of positive integer N into K parts, each part is either $\lfloor \frac{N}{K} \rfloor$ or $\lceil \frac{N}{K} \rceil$.*

Proof. By the integer division theorem,

$$N = K \lfloor \frac{N}{K} \rfloor + r,$$

with $0 \leq r < K$.

If $r = 0$, then each part of K gets $\lfloor \frac{N}{K} \rfloor$; otherwise, divide r into r parts and add them to r parts of the previous K parts, this results in r parts out of K each equals $\lfloor \frac{N}{K} \rfloor + 1 = \lceil \frac{N}{K} \rceil$ and $K - r$ parts each equals $\lfloor \frac{N}{K} \rfloor$.

This partition can be noted in a general uniform format $(\lfloor \frac{N}{K} \rfloor, \lfloor \frac{N+1}{K} \rfloor, \cdots, \lfloor \frac{N+K-1}{K} \rfloor)$.

□

Corollary 6.6.5. *For an arbitrary partition $(n_1, n_2, \cdots, n_K)$ of positive integer N into K parts, if there is a $n_i > \lceil \frac{N}{K} \rceil$ $(i = 1, 2, \ldots, K)$, then there exist at least one n_j such that $n_j < \lceil \frac{N}{K} \rceil$ $(j \neq i,\ j = 1, 2, \ldots, K)$.*

Now, we can give the proof for **Theorem 6.6.1**:

Proof. We use **Algorithm 7** to prove **Theorem 6.6.1**.

Algorithm 7: Find the optimum allocation of sensors to patches

```
   Input: $0 < K \leq N$
   Output: p
 1 $P \leftarrow (n_1, n_2, \ldots, n_K)$ is an arbitrary partition of $N$ into $K$ parts
 2 $p \leftarrow \prod_{i=1}^{K} (1 - F^{P[i]})$
 3 Sort $P$ in ascending order
 4 $i \leftarrow 1$
 5 $j \leftarrow K$
 6 while ($P[j] > \lceil \frac{N}{K} \rceil$ or $P[i] < \lceil \frac{N}{K} \rceil$ ) and $i < j$ do
 7     $P[j] \leftarrow P[j] - 1$
 8     $P[i] \leftarrow P[i] + 1$
 9     while $P[j] = \lfloor \frac{N}{K} \rfloor$ do
10         $j \leftarrow j - 1$
11     end
12     while $P[i] = \lceil \frac{N}{K} \rceil$ do
13         $i \leftarrow i + 1$
14     end
15 end
16 $p \leftarrow \prod_{i=1}^{K} (1 - F^{P[i]})$
```

In **Algorithm 7**, **line 6**, if the condition is not satisfied, then the partition P is already the most even partition. Otherwise, by **Corollary 6.6.5**, the body of the **while** loop will be executed. By **Lemma 6.6.3**, each iteration only moves toward or generates term of $\lceil \frac{N}{K} \rceil$ and/or $\lfloor \frac{N}{K} \rfloor$ and increase or keep the product p, so it will reach the most even partition when the loop exit and get the maximum product at **line 16**. □

6.7 ACHIEVABILITY OF A LONG TERM TASK

If a long term task lasts for G generations, under previous assumptions, we say the task is achievable if there are at least one sensor survive for each patch at the end of the G^{th} generation.

First, consider a simple case for a single patch k of the monitored area, there are N_k sensors in total for this patch, G generations needed, each generation is deployed by drone, this case is similar to the case of K patches for one generation discussed in section 6.6. If the allocation of sensors for each generation is n_{ki}, $\sum_{i=1}^{K} n_{ki} = N_k$, $i \in \{1,2,\dots,G\}$, $k \in \{1,2,\dots,K\}$, then the probability of the event that there are at least one sensor of the first generation survive T_r is:

$$P_{k1} = 1 - F^{n_{k1}}(T_r). \tag{6.10}$$

For the second generation, those from the first generation that survived T_r will join them and work together, then, at the end of the second generation, the survival event is there are at least one sensor from the second generation survive T_r as event $E_{G_{k2}^{T_r}}$ or there are at least one sensor from the first generation survive $2T_r$ as event $E_{G_{k1}^{2T_r}}$, $E_{G_{k2}^{T_r}}$ and $E_{G_{k1}^{2T_r}}$ are independent, so the corresponding probability is:

$$\begin{aligned}
P_{k2} &= Pr\{E_{G_{k1}^{2T_r}} \cup E_{G_{k2}^{T_r}}\} \\
&= Pr\{E_{G_{k1}^{2T_r}}\} + Pr\{E_{G_{k2}^{T_r}}\} - Pr\{E_{G_{k1}^{2T_r}}\}Pr\{E_{G_{k2}^{T_r}}\} \\
&= 1 - F^{n_{k1}}(2T_r) + 1 - F^{n_{k2}}(T_r) - (1 - F^{n_{k1}}(2T_r))(1 - F^{n_{k2}}(T_r)) \\
&= 1 - F^{n_{k1}}(2T_r)F^{n_{k2}}(T_r).
\end{aligned} \tag{6.11}$$

It can be proved by mathematical induction that the survival probability of the g_{th} generation is:

$$\begin{aligned}
P_{kg} &= 1 - F^{n_{k1}}(gT_r)F^{n_{k2}}((g-1)T_r)\cdots F^{n_{k(g-1)}}(2T_r)F^{n_{kg}}(T_r) \\
&= 1 - \prod_{i=1}^{g} F^{n_{ki}}((g-i+1)T_r).
\end{aligned} \tag{6.12}$$

Proof. Prove by mathematic induction on non-zero natural number g.

Basis step:

$$P_{k1} = 1 - \prod_{i=1}^{1} F^{n_{ki}}((1-i+1)T_r) = 1 - F^{n_{k1}}(T_r)$$

which is the same as equation (6.10).

Induction step: We need to prove $P_{kg} \rightarrow P_{k(g+1)}$.

Based on the assumption that $E_{G_{k(g+1)}^{T_r}}$, $E_{G_{kg}^{2T_r}}$, $\ldots, E_{G_{k2}^{gT_r}}$ and $E_{G_{k1}^{(g+1)T_r}}$ are independent,

$$\begin{aligned}
P_{k(g+1)} &= Pr\{\cup_{i=1}^{g+1} E_{G_{ki}^{((g+1)-i+1)T_r}}\} \\
&= Pr\{(\cup_{i=1}^{g} E_{G_{ki}^{((g+1)-i+1)T_r}}) \cup E_{G_{k(g+1)}^{T_r}}\} \\
&= Pr\{\cup_{i=1}^{g} E_{G_{ki}^{((g+1)-i+1)T_r}}\} + Pr\{E_{G_{k(g+1)}^{T_r}}\} - Pr\{\cup_{i=1}^{g} E_{G_{ki}^{((g+1)-i+1)T_r}}\} Pr\{E_{G_{k(g+1)}^{T_r}}\} \\
&= 1 - \prod_{i=1}^{g} F^{n_{ki}}(((g+1)-i+1)T_r) + 1 - F^{n_{k(g+1)}}(T_r) \\
&- \left(1 - \prod_{i=1}^{g} F^{n_{ki}}(((g+1)-i+1)T_r)\right)(1 - F^{n_{k(g+1)}}(T_r)) \\
&= 1 - F^{n_{k(g+1)}}(T_r) \prod_{i=1}^{g} F^{n_{ki}}(((g+1)-i+1)T_r) \\
&= \prod_{i=1}^{g+1} F^{n_{ki}}(((g+1)-i+1)T_r).
\end{aligned}$$

□

Combine all G generations, the survival probability of the whole lineage in the k^{th} patch is:

$$P_k = \prod_{g=1}^{G} \left(1 - \prod_{i=1}^{g} F^{n_{ki}}((g-i+1)T_r)\right). \tag{6.13}$$

If none of the sensors can survive more than $2T_r$, i.e. $F(2T_r) = F(3T_r) = \cdots = F(GT_r) = 1$, then equation (6.13) degenerated into the following format:

$$P_k = \prod_{g=1}^{G} [1 - F^{n_{kg}}(T_r)] \tag{6.14}$$

which is the same as equation (6.8).

For K patches, the achievability P_A of the long term task can be modeled by equation (6.15) based on the fact that each patch survives independently and equation (6.13):

$$P_A = \prod_{k=1}^{K} P_k = \prod_{k=1}^{K} \left(\prod_{g=1}^{G} \left(1 - \prod_{i=1}^{g} F^{n_{ki}}((g-i+1)T_r)\right)\right), \tag{6.15}$$

where $\sum_{i=1}^{G} n_{ki} = N_k$ and $\sum_{k=1}^{K} N_k = N$. Due to the symmetry of P_k, P_A can be maximized by allocating N sensors evenly into K parts. Which can be proved in a similar way as Proof (6.6). So the next step to maximize P_A is to maximize all P_k, i.e. what is the best way to allocate sensors to each generation of a lineage? In the degenerated situation described in equation (6.14), the best way is allocating sensors evenly into each generation.

In order to find the maximum of P_k, first consider two generations of the first patch, which is a relatively simple case. Equation (6.13) can be specialized and rewritten as:

$$P_1 = \left[1 - F^{n_{11}}(T_r)\right]\left[1 - F^{n_{11}}(2T_r)F^{N_1 - n_{11}}(T_r)\right], \tag{6.16}$$

with $n_{11} \in \{1, 2, \ldots, N_1\}$. To determine the characteristics of function $P_1(n_{11})$, suppose a typical situation, the sensors' working lives are normal distributed with expectation 30 days and standard deviation from 3 days(relatively good uniformity) to 12 days(very bad uniformity). If taking T_r as the expectation, then $F(T_r)$ and $F(2T_r)$ can be calculated versus the standard deviation σ, the results are shown in Table (9). For those two best cases which $F(2T_r)$ are 1.0 under the precision of IEEE-754 double, they can be approximated very well by corresponding degenerated form, i.e, P_1 reaches its maximum by allocating sensors evenly to each generation. Take the worst case, let $F(2T_r) = 0.99$ and $F(T_r) = 0.5$, substitute back into equation (6.16), get $P_1 = \left[1 - 0.5^{n_{11}}\right]\left[1 - 0.99^{n_{11}} 0.5^{N_1 - n_{11}}\right]$, whose curve is shown in Figure 6.7. Calculation shows that P_1 reaches its max 0.998141114365831 at n_{11}=10, even though P_1 has values very close to this maximum at values $\{7, 8, 9, 11, 12, 13\}$, all above 0.99! The best way to maximize P_1 is still even distribution of sensors to each generation.

TABLE 9: $F(T_r)$ and $F(2T_r)$ vs standard deviation σ

σ	$F(T_r)$	$F(2T_r)$
3	0.5	1.0
4	0.5	1.0
5	0.5	0.999999999013
6	0.5	0.999999713348
7	0.5	0.999990892351
8	0.5	0.999911582715
9	0.5	0.999570939667
10	0.5	0.998650101968
11	0.5	0.996806988359
12	0.5	0.993790334674

For multiple generations more than two, $F(GT_r)$ turns very quickly to 1.0 under the

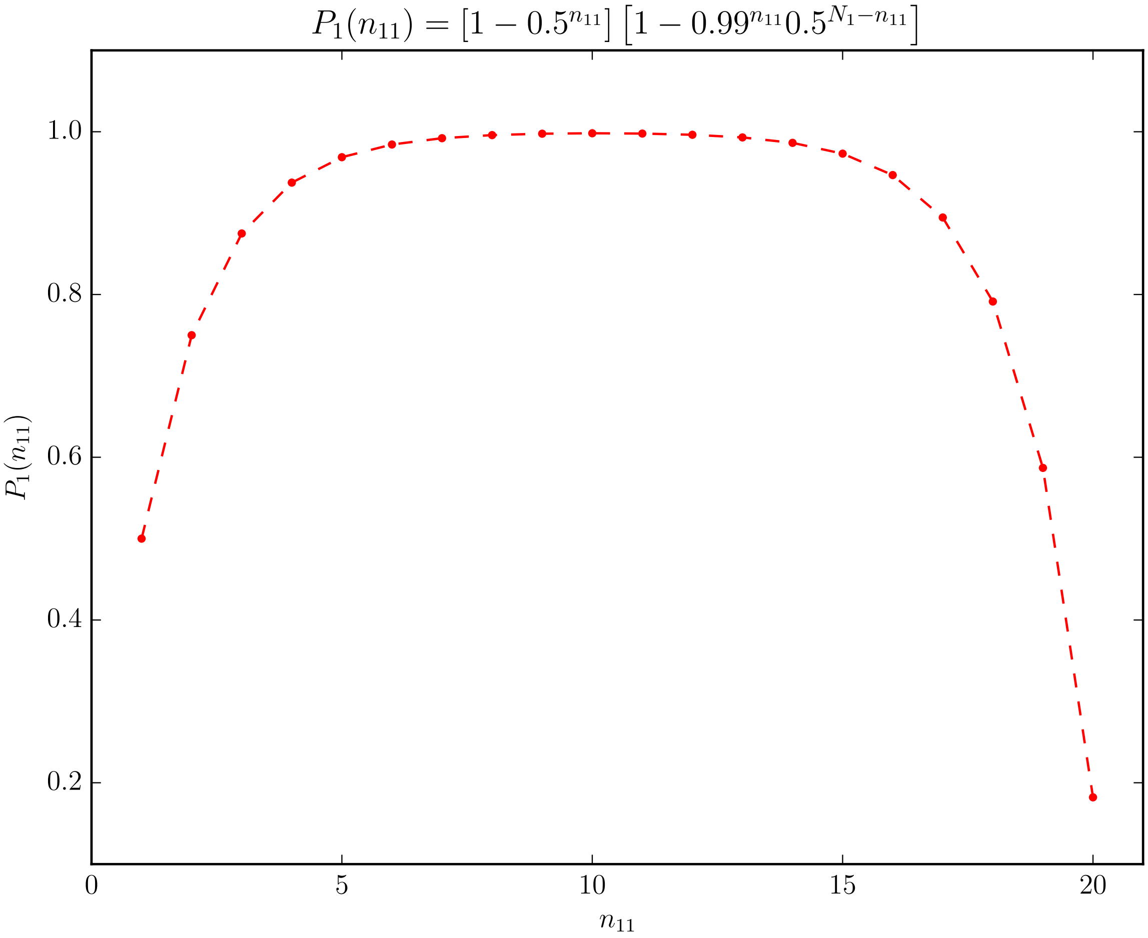

FIG. 36: The curve of $P_1(n_{11})$

precision of IEEE-754 double with $G = 5$ at the worst case as above, as well, calculation shows P_k is maximized with even distribution of sensors to each generation with $G = 5$, so it is reasonable to use even distribution as a best way to maximize P_k.

For generation renewal with scheme of sleep and wake up, by choosing sleep survival expectation doubles the entire period of the long term task, for example, for one year task, choose sensors with sleep survival expectation of two year and standard deviation of one month(roughly bad uniformity), then these sensors survive one year with almost certainty — a probability of $1 - 1.776482112077653 \cdot 10^{-33}$, however, the energy will only left a half for the last generation to work since energy dissipates during sensors' sleep. Even for generations other than the first generation, the dissipation of energy due to sleep also affect the survival distribution of working generations, in general, it is hard to find the analytical optimization distribution of sensors to each generation to maximize the achievability of the task.

In real world, a non-rechargeable sensor's sleep lifespan are dominantly determined by the equipped energy capacity, and its working lifespan depends not only the equipped power but also its working intensity or pattern. It is reasonable to assume a sensor's energy dissipates linearly with time for sleep state. For simplicity, we also assume a sensor's energy dissipates linearly with time for working in full load. Denote a sensor's sleep lifespan expectation as T_s, and full load working lifespan expectation as T_w, then a sensor's full load working lifespan expectation of the G^{th} generation denoted as T_w^G is:

$$T_w^G = \frac{T_w}{T_s}\left[T_s - (G-1)T_w\right] \tag{6.17}$$

where, $T_w^1 = T_w$. If $T_s = GT_w$, under previous assumption, then the expected full load working time for total G generations denoted as T_f^G is:

$$T_f^G = \sum_{i=1}^{G} T_w^i = \sum_{i=1}^{G} \frac{T_w}{T_s}\left[T_s - (i-1)T_w\right] = \frac{T_s + T_w}{2} = \frac{G+1}{2}T_w \tag{6.18}$$

In order to get the theoretical boundaries of T_f^G, suppose sensors' each kind of lifespan, by themselves, are identically independently distributed as a truncated normal distribution $T \sim N(\mu, \sigma^2)$ on the support of $[\mu - \nu, \mu + \nu]$, for etample, consider full load working lifespan, $\mu = 30$, $\nu = 3$ means all sensors have an expected lifespan of 30 days in the range from 27 to 33 days. The corresponding *cumulative distribution function* is:

$$F(t;\mu,\sigma,\mu-\nu,\mu+\nu) = \frac{\Phi\left(\frac{t-\mu}{\sigma}\right) - \Phi\left(\frac{-\nu}{\sigma}\right)}{\Phi\left(\frac{\nu}{\sigma}\right) - \Phi\left(\frac{-\nu}{\sigma}\right)}$$

for $\mu-\nu \le t \le \mu+\nu$ and 0 elsewhere. Here, $\Phi(.)$ is the *CDF* of standard normal distribution. It is clear that $F(t)=0$ for all $t \le \mu-\nu$ and $F(t)=1$ for all $t \ge \mu+\nu$. The corresponding lower boundary and upper boundary are denoted as $\underline{T} = \mu-\nu$ and $\overline{T} = \mu+\nu$ correspondingly. Then the theoretical boundaries of T_f^G are:

$$\underline{T_f^G} = \frac{\underline{T_s}+\underline{T_w}}{2} \le T_f^G \le \frac{\overline{T_s}+\overline{T_w}}{2} = \overline{T_f^G}$$

while, $\underline{T_s} = \underline{G}\cdot\underline{T_w}$ and $\overline{T_s} = \overline{G}\cdot\overline{T_w}$.

Under the assumption of truncated normal distribution and suppose ν is independent of generations, take T_w^G as the generation replace time denoted as T_r^G, in order to make $F(2T_r) = F(3T_r) = \cdots = F(GT_r) = 1$, only let

$$T_w^i + \nu = \frac{T_w}{T_s}\left[T_s - (i-1)T_w\right] + \nu \le 2T_w^i$$

solve i from this inequality,

$$i = \lfloor 1 + \frac{T_s}{T_w}(1-\frac{\nu}{T_w})\rfloor = 1 + \lfloor \frac{T_s}{T_w}(1-\frac{\nu}{T_w})\rfloor \tag{6.19}$$

For a typical setting, $T_w = 30$ days, $T_s = 360$ days, $\nu = 3$ days, by equation (6.19), $i = 11$ generations, which last half a year by equation (6.18), i.e. distributing sensors evenly into each generations maximize the achievability of a long term task described in equation (6.15). Now, for a long term task, we have the theoretical basis to make our choices for the sleep-wakeup scheme.

CHAPTER 7

PROPOSAL OF IMPLEMENTATION

Unmanned aerial vehicles(UAVs) are used for various purposes in wireless sensor network. In areas where communication between grounds node is difficult such as isolated areas, UAVs working as mobile sensors are used [92]. G Tuna *et al.* [94] use UAVs to deploy WSN for post-disaster monitoring. J. Leng [59] uses UAVs to wirelessly charge the sensor nodes and prolong the sensor network lifetime. Villas *et al.* [95] use GPS-equipped UAVs to addresses the problem of 3D localization in WSNs. Here, we will use UAVs to deploy generations of wireless sensors and localization.

In most wireless sensor networks deployed several years ago, their sensor's power capacity, computing and storage resource and transmitting are very limited [14, 15, 50]. Usually, their transmitting range are about 50 to 100 meters, in order to monitor an area with 100 km^2, it will needs at least 10,000 sensors, it is impractical to run distributed clustering algorithm with multi-hops and uncertainty in convergence of the algorithm in such a relatively large scale. This may be the major reason for the scarcity of deployment of WSNs. In our research, we will propose using cutting-edge wireless MCU CC1310 [22] from Texus Instruments to construct deterministically reactive WSN, which is vital to monitor and react to emergencies. CC1310 can be equipped with long-range narrow band RF transceiver CC1120 and ranger extender CC1190 which have an amazing transmitting range from 20km to 100km, it also can interface to CC1190 directly to improve the transmission range of its integrated RF transceiver module. It can sleep up to 20 years through utilizing lower-power shutdown at only 185nA, ultra-low power radio and smart sensor control. With such long range transmission capability, it can be used to construct WSN that covers cities and mountains.

For example, the city of Norfolk, VA covers about 250 km^2 [4], the diameter is about 16km, which can be covered by the transmission range of long-range narrow band RF transceivers. If the sensing range or required resolution is 100m, then we need 8,000 CC1310s to cover the whole city of Norfolk. If the CC1310s are equipped with sensors for measuring temperature, humidity, precipitation, wind-speed and atmospheric pressure, then a detailed map of the micro-weather of Norfolk can be provided on real-time.

For another example, monitoring possible wildfire in a forest, such as Sierra national forest [5] in California, it covers 5,300km^2, its diameter is about 75km, which is also in the transmission range of long-range narrow band RF transceivers. If the sensing range or required resolution is also 100m, then we need 168,789 CC1310s to cover the whole forest. The CC1310s are equipped with sensors for measuring temperature, humidity, precipitation, wind-speed and atmospheric pressure to monitor the whole forest in real-time. Here the number of sensors is a large number, the combination of wireless micro-controllers with different transmission ranges would be a better choice.

CC1310 is a member of the CC26xx and CC13xx family of cost-effective, ultralow power, 2.4 GHz and sub-1-Ghz RF micro-controllers from Texas Instruments. It includes many features:

- *High performance:* A powerful ARM Cortex-M3 core runs up to 48 MHz clock speed.

- *Advanced power management:* In active mode, the MCU running at 48 MHz consumes 2.5 mA (51 $\mu A/MHz$); in standby mode in which RTC is running and RAM and CPU keep retention, 0.6 μA is consumed; in shutdown mode, which can be waken up on external events, only 185 nA is consumed.

- *Dedicated ultralow power sensor controller:* can run autonomously from the rest of the system. It has separate 2KB of ultralow leakage SRAM for code and data.

- *Abundant peripherals:* It has four general-purpose timers; 12-Bit ADC with 8-channel analog MUX capable of 200 $ksamples/s$; 1 continuous time comparator and ultralow power clocked comparator; 1 programmable current source, UART, I2C, I2S, RTC and 2 SSI.

- *Powerful RF:* It has excellent receiver sensitivity of -124 dBm using long-range mode and -110 dBm at 50 $kbps$, selectivity of 52 dB, blocking performance of 90 dB, programmable output power up to 14 dBm. It can seamlessly integrated with CC1190 range extender.

- *High security:* It has a true random number generator and an AES-128 security module.

- *Ease of use:* It integrated TI-RTOS, drivers and bootloader in its ROM. It supports over-the-air upgrade which simplifies upgrading software.

CHAPTER 8

CONTRIBUTIONS AND FUTURE RESEARCH DIRECTIONS

8.1 CONTRIBUTIONS

Our research provided a formal way of looking at aggregation of information in networks where individual actors possess information whose value decays over time. We offered a formal model for the valuation of time-discounted information and of the algebra of its aggregation.

We allowed aggregation of time-discounted information to proceed in an arbitrary, not necessarily pairwise, manner. We have shown that the resulting value of the aggregate does not depend on the order in which aggregation of individual values take place.

Our results suggest natural thresholding strategies for the aggregation of the information collected by sets of network actors. Our theoretical predictions were confirmed by extensive simulation.

With formal models set up for the continuity of sensors, patched, clusters and generations, the dissertation provides an optimum way to allocate sensors over patches, clusters and generations to maximize the achievability of long-term tasks.

8.2 FUTURE RESEARCH DIRECTIONS

With the research results we have gotten, there are still many relative interesting problems remain in the assessment of the value of information and the dynamic discounting process of this value, and aggregation these values across generations autonomously. Here are some interesting directions:

Formal assessment of value: There are many factors that have effects on the value of information such as the type, source and amount of the value, time and location when and where the value is collected and who are users. In this subtask, we are trying to come up with a formal description of the dependence of the value of information on those effect factors. Especially some families of functions that map data into a value space such as utility, probability and confidence, etc will be given. Algorithms for this assessment are also in our consideration.

Other information discounting: Time discounting regimens other than those described in Chapter 3 such as multiplicative discounting by a factor that decreases with time in the interval $[0,1]$ and additive discounting by a value from the same value space as the being discounted may implicate theoretical and practical values.

Examples of such regimens are known to exist, step functions being a prime example. The question that we are addressing here is that of approximating a step function and, indeed, other similar decay functions by polynomials. It is a classic result of Weierstrass that every real function can be approximated by a suitable sequence of polynomials [70]. This approach seems natural as every exponential can be approximated by a polynomial consisting of the first few terms in its Taylor expansion.

For those factors that discount the value of information other than time such as location and users, as well as the effect and the way of these factors on the underline value of information, we will set up some families of functions to expound those factors, some families of operators to explain the effecting way exerted by those factors on the value of information. Other two trends of value evolution — keeping constant and increasing will be discussed in this subtask.

Aggregation of data and its interaction with discounting: The three types of aggregators explained in Chapter 4, are derived from the comparison of the aggregated values of two permutation processes: one is aggregate individual values first then use a discounted aggregated value later, the other is aggregate individual values that are discounted. Further study on them constitute this direction which has following areas:

Generalization of type 1 aggregation operators: The special class of Type 1 aggregation operators explained in Section 4.7 can be generalized as

$$X \Diamond Y = X + Y - \alpha XY$$

where α is a real number and $\alpha \in [0,2]$. The meaning and implication of this generalization require exploration, especially when α is other than 1 which has been studied.

Special classes of Type 2 aggregation operator such as min, max and weighted average seems trivial, however, it is a big challenging to model the dynamics of these operators carried out by a set of sensors distributively. We are considering modeling this dynamic process with Markov chain.

Type 3 aggregation operators: For a Type 3 aggregation operator, the best strategy is to aggregate as early as the data is available and/or the semantics of the application permit. It is interesting to find some special applicable candidates and their applications.

Interaction between aggregating and discounting: The interaction between multiplicative discount with dimensionless factors and aggregation has been discussed in Section 4.4. It is interesting to look into other types of discount such as additive discount with certain amount of value in this direction.

Handling of dynamic evolution: Long-term deployed large scale sensor networks are used to catch the dynamic evolution of long-term phenomenons. Renewal theory is promising to be used to model this evolution and find economical approaches to renew faulted sensors and recall obsolete ones. The information discounting and aggregating across consecutive generations can be described formally in a similar way. If there are many consecutive tasks scheduled in a long term mission, sensor retask will be one of the optional approaches to handle this challenge. Retasking may involve moving through various sets of sensed attributes in order to infer a derived attribute for which the sensor does not have a direct sensing capability.

Simulation and verification: We discuss information discounting and aggregating in the context of sensor network. Usually, each sensor has very limited resources such as computing power, small storage space, low capacity battery and short range radio, etc. Such a context implicates various aggregation strategies such as algorithms for implementation of special classes of aggregation operators will be constrained by these characteristics. The context overlaid by sensor network can be extended to sets of actors, whether sensors, robots, or people, similar result will be gotten as we only concern the data they possess except their concrete dynamics. It is interesting to develop decision strategies for the aggregation of the data collected by sets of actors and apply these strategies to fields in which hybrid networks of humans and sensors need to be formed, such as sensor networks deployed in support of emergency response and tactical applications. Due to the limitation of real platform, it is promising to develop a series of related algorithms and simulation models to implement, verify and confirm our theoretical models.

References

[1] AMBER Alert. https://en.wikipedia.org/wiki/AMBER_Alert.

[2] Exponential decay. https://en.wikipedia.org/wiki/Exponential_decay.

[3] GDACS (Global Disaster Alert and Coordination System). http://www.gdacs.org/.

[4] Norfolk of Virginia. https://en.wikipedia.org/wiki/Norfolk_Virginia.

[5] Sierra national forest. https://en.wikipedia.org/wiki/Sierra_National_Forest.

[6] Wireless emergency alerts. https://en.wikipedia.org/wiki/Wireless_Emergency_Alerts.

[7] Worst natural disasters of 2015. http://www.worldvision.org.

[8] M. Abu Alsheikh, S. Lin, D. Niyato, and H.-P. Tan. Machine learning in wireless sensor networks: Algorithms, strategies, and applications. *Communications Surveys & Tutorials, IEEE*, 16(4):1996–2018, 2014.

[9] N. Ahituv. A systematic approach toward assessing the value of an information system. *MISQ*, 4(4):61–75, 1980.

[10] N. H. Anderson. *Foundations of information integration theory*. Academic Press, New York, 1981.

[11] N. H. Anderson. *Contributions To Information Integration Theory: Volume 1: Cognition*. Psychology Press, 2014.

[12] K. J. Arrow. The value of and the demand for information. In C. B. McGuire and R. Radner, editors, *Decision and organization. A volume in honor of Jacob Marschak*, pages 131–140. North-Holland Pub. Co., Amsterdam, 1972.

[13] A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[14] J. Beutel, K. Römer, M. Ringwald, and M. Woehrle. Deployment techniques for sensor networks. In *Sensor Networks*, pages 219–248. Springer, 2010.

[15] Z. Bojkovic and B. Bakmaz. A survey on wireless sensor networks deployment. *WSEAS Trans. on Communications*, 7(12):1172–1181, 2008.

[16] T. Brody, S. Harnad, and L. Carr. Earlier web usage statistics as predictors of later citation impact. *Journal of the American Society for Information Scienced and Technology*, 57(8):1060–1072, 2006.

[17] L. Busoniu, R. Babuska, and B. De Schutter. Multi-agent reinforcement learning: A survey. In *Control, Automation, Robotics and Vision, 2006. ICARCV'06. 9th International Conference on*, pages 1–6. IEEE, 2006.

[18] L. Busoniu, R. Babuska, and B. De Schutter. A comprehensive survey of multiagent reinforcement learning. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 38(2):156–172, 2008.

[19] L. Busoniu, R. Babuska, and B. De Schutter. A comprehensive survey of multiagent reinforcement learning. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 38(2):156–172, 2008.

[20] C.-h. Chen, L.-F. Pau, and P. S.-p. Wang. *Handbook of pattern recognition and computer vision*, volume 27. World Scientific, 2010.

[21] J.-Y. Chen, G. Pandurangan, and D. Xu. Robust computation of aggregates in wireless sensor networks: distributed randomized algorithms and analysis. *Parallel and Distributed Systems, IEEE Transactions on*, 17(9):987–1000, 2006.

[22] J. Copley. Diversifying the IoT with Sub-1 GHz technology. http://www.ti.com/lit/wp/swry017/swry017.pdf, 10 2015.

[23] A. Criminisi, J. Shotton, and E. Konukoglu. Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning. *Foundations and Trends in Computer Graphics and Vision*, 7(2–3):81–227, 2012.

[24] N. Cristianini and J. Shawe-Taylor. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[25] K. Crowston et al. *A taxonomy of organizational dependencies and coordination mechanisms*. Center for Coordination Science, Alfred P. Sloan School of Management, Massachusetts Institute of Technology, 1994.

[26] C. De Mol, E. De Vito, and L. Rosasco. Elastic-net regularization in learning theory. *Journal of Complexity*, 25(2):201–230, 2009.

[27] K. Deutsch. Communication theory and political integration. In P. E. Jacob and J. V. Toscano, editors, *The integration of political communities*. Lippincott, Philadelphia, 1964.

[28] M. Di and E. M. Joo. A survey of machine learning in wireless sensor netoworks from networking and application perspectives. In *Information, Communications & Signal Processing, 2007 6th International Conference on*, pages 1–5. IEEE, 2007.

[29] J. F. Elder IV. Machine learning, neural, and statistical classification. *Journal of the American Statistical Association*, 91(433):436–439, 1996.

[30] P. A. Forero, A. Cano, and G. B. Giannakis. Distributed clustering using wireless sensor networks. *Selected Topics in Signal Processing, IEEE Journal of*, 5(4):707–724, 2011.

[31] S. Frederick, G. Loewenstein, and T. O'Donoghue. Time discounting and time preference: A critical review. *Journal of Economic Literature*, XL:351–401, 2002.

[32] A. A. Freitas. A survey of evolutionary algorithms for data mining and knowledge discovery. In *Advances in evolutionary computing*, pages 819–845. Springer, 2003.

[33] N. Friesen. Interoperability and learning objects: An overview of e-learning standardization. *Interdisciplinary Journal of Knowledge and Learning Objects*, 1(1):23–31, 2005.

[34] W. A. Fuller. Regression analysis for sample survey. *Sankhya Series C*, 37:117–132, 1975.

[35] M. Goebel and L. Gruenwald. A survey of data mining and knowledge discovery software tools. *ACM SIGKDD explorations newsletter*, 1(1):20–33, 1999.

[36] D. E. Goldberg and J. H. Holland. Genetic algorithms and machine learning. *Machine learning*, 3(2):95–99, 1988.

[37] N. Grira, M. Crucianu, and N. Boujemaa. Unsupervised and semi-supervised clustering: a brief survey. *A review of machine learning techniques for processing multimedia content, Report of the MUSCLE European Network of Excellence (FP6)*, 2004.

[38] P. Guo, T. Jiang, Q. Zhang, and K. Zhang. Sleep scheduling for critical event monitoring in wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 23(2):345 –352, february 2012.

[39] H. C. Hartmann, R. Bales, and S. Sorooshian. Weather, climate, and hydrologic forecasting for the us southwest: a survey. *Climate Research*, 21(3):239–258, 2002.

[40] A. E. Hoerl and R. W. Kennard. Ridge regression: applications to nonorthogonal problems. *Technometrics*, 12(1):69–82, 1970.

[41] R. Howard. Information value theory. *IEEE Transactions on Systems Science and Cybernetics*, 2(1):22 –26, August 1966.

[42] C.-W. Hsu and C.-J. Lin. A comparison of methods for multiclass support vector machines. *Neural Networks, IEEE Transactions on*, 13(2):415–425, 2002.

[43] A. K. Jain. Data clustering: 50 years beyond K-means. *Pattern recognition letters*, 31(8):651–666, 2010.

[44] K. H. Jones, K. Lodding, A. Wadaa, S. Olariu, L. Wilson, and M. Eltoweissy. Biomimetic models for sensor networks: Towards a social sensor network. *Handbook of Bio-Inspired Algorithmic Techniques, CRC Press, Boca Raton*, 2005.

[45] K. H. Jones, K. N. Lodding, S. Olariu, L. Wilson, and C. Xin. Biology-inspired distributed consensus in massively-deployed sensor networks. In *Ad-Hoc, Mobile, and Wireless Networks*, pages 99–112. Springer, 2005.

[46] K. H. Jones, K. N. Lodding, S. Olariu, L. Wilson, and C. Xin. Energy usage in biomimetic models for massively-deployed sensor networks. In *Parallel and Distributed Processing and Applications-ISPA 2005 Workshops*, pages 434–443. Springer, 2005.

[47] K. H. Jones, K. N. Lodding, S. Olariu, L. Wilson, and C. Xin. Sensor networks for situation management: a biomimetic model. In *Military Communications Conference, 2005. MILCOM 2005. IEEE*, pages 1787–1793. IEEE, 2005.

[48] K. H. Jones, K. N. Lodding, S. Olariu, L. Wilson, and C. Xin. Communal cooperation in sensor networks for situation management. In *Information Fusion, 2006 9th International Conference on*, pages 1–8. IEEE, 2006.

[49] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, pages 237–285, 1996.

[50] I. Khoufi, P. Minet, A. Laouiti, and S. Mahfoudh. Survey of deployment algorithms in wireless sensor networks: coverage and connectivity issues and challenges. *International Journal of Autonomous and Adaptive Communications Systems (IJAACS)*, page 24, 2014.

[51] K. Koperski, J. Adhikary, and J. Han. Spatial data mining: progress and challenges survey paper. In *Proc. ACM SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery, Montreal, Canada*, pages 1–10. Citeseer, 1996.

[52] S. B. Kotsiantis. Supervised machine learning: A review of classification techniques. In *Proceedings of the 2007 Conference on Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in eHealth, HCI, Information Retrieval and Pervasive Technologies*, pages 3–24, Amsterdam, The Netherlands, The Netherlands, 2007. IOS Press.

[53] P. Kozma-Wiebe, S. M. Silverstein, A. Feher, I. Kovacs, P. Ulhaas, and S. M. Wilkniss. Development of a world-wide web based contour integration test. *Computers in Human Behavior*, 22(6):971–980, 2006.

[54] M. Kuczma. *An introduction to the theory of functional equations and inequalities: Cauchy's equation and Jensen's inequality*. Springer Science & Business Media, 2009.

[55] V. Lakshmanan, E. Gilleland, A. McGovern, and M. Tingley. *Machine Learning and Data Mining Approaches to Climate Science*. Springer, 2015.

[56] K. Lakshminarayan, S. A. Harp, R. P. Goldman, T. Samad, et al. Imputation of missing data using machine learning techniques. In *KDD*, pages 140–145, 1996.

[57] P. R. Lawrence and J. W. Lorsch. *Organization and environment; managing differentiation and integration*. Harvard University, Boston, 1967.

[58] S. Laxman and P. S. Sastry. A survey of temporal data mining. *Sadhana*, 31(2):173–198, 2006.

[59] J. Leng. *Using a uav to effectively prolong wireless sensor network lifetime with wireless power transfer*. PhD thesis, University of Nebraska, 2014.

[60] Z. Li, Y.; Wang and Y. Song. Wireless sensor network design for wildfire monitoring. *Proc. Sixth World Congress on Intelligent Control and Automation WCICA 2006*, 1:109–113, 2006.

[61] Y. Liu and R. H. Weisberg. *A review of self-organizing map applications in meteorology and oceanography*. INTECH Open Access Publisher, 2011.

[62] J. L. Lu, S. J. Williams, and L. Kaufman. Behavioral lifetime of human auditory sensory memory predicted by physiological measures. *Science*, 258:1668–1670, 1992.

[63] A. C. MacKinlay. Event studies in economics and finance. *Journal of Economic Literature*, 35(1):13–39, 1997.

[64] T. W. Malone and K. Crowston. Interdisciplinary study of coordination. *ACM Computing Surveys*, 26(1):87–119, 1994.

[65] T. W. Malone, K. Crowston, J. Lee, B. Pentland, C. Dellarocas, G. Wyner, J. Quimby, C. S. Osborn, A. Bernstein, G. Herman, M. Klein, and E. O'Donnell. Tools for inventing organizations: Toward a handbook of organizational processes. *Management Science*, 45(3):425–443, 1999.

[66] B. Manoj and A. H. Baker. Communication challenges in emergency response. *Communications of the ACM*, 50(3):51–53, March 2007.

[67] E. Manolakos and G. Xanthopoulos. Temperature field modelling and simulation of wireless sensor network behaviour during a spreading wildfire. *Proc. European Signal Processing Conference (EUSIPCO 2008)*, 2008.

[68] J. Marschak. Economics of information systems. *Journal of the American Statistical Association*, 66(333):192–219, 1971.

[69] J. Marschak and R. Radner. *Economic theory of teams*. Yale University Press, New Haven, 1972.

[70] J. E. Marsden and M. J. Hoffman. *Elementary classical analysis*. Macmillan, 1993.

[71] MemsIC. IRIS datasheet. http://www.memsic.com/userfiles/files/Datasheets/WSN/IRIS_Datasheet.pdf.

[72] T. M. Mitchell. *The discipline of machine learning*, volume 9. Carnegie Mellon University, School of Computer Science, Machine Learning Department technical report CMU-ML-06-108, 2006.

[73] J. V. Nickerson. A concept of communication distance and its application to six situations in mobile environments. *IEEE Transactions on Mobile Computing.*, 5(4):409–419, 2005.

[74] J. V. Nickerson. Flying sinks: Heuristics for movement in sensor networks. In *Proceedings of the 39th Annual Hawaii International Conference on System Sciences*, 2006.

[75] J. V. Nickerson and S. Olariu. A measure for integration and its application to sensor networks. In *Workshop on Information Technology and Systems (WITS)*, 2005.

[76] J. V. Nickerson and S. Olariu. Courier assignment in social networks. In *Proceedings of the 40th Annual Hawaii International Conference on System Sciences*, HICSS '07, pages 46–, Washington, DC, USA, 2007. IEEE Computer Society.

[77] J. V. Nickerson and S. Olariu. Protecting with sensor networks: Attention and response. In *Proceedings of the 40th Annual Hawai'i International Conference on System Sciences*, 2007.

[78] S. Olariu, M. Eltoweissy, and M. Younis. ANSWER: AutoNomouS netWorked sEnsoR system. *Journal of Parallel and Distributed Computing*, 67(1):111–124, 2007.

[79] S. Olariu, S. Mohrehkesh, X. Wang, and M. C. Weigle. On aggregating information in actor networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 18(1):85–96, Feb. 2014.

[80] S. Olariu, S. Mokhrekesh, and M. Weigle. Toward agregating time discounted information. In *MiSeNet'2013*, September 2013.

[81] S. Olariu and J. V. Nickerson. Protecting with sensor networks: Perimeters and axes. In *MILCOM*, 2005.

[82] S. Olariu and J. V. Nickerson. A probabilistic model of integration. *Decision Support Systems*, 45(4):746 – 763, 2008.

[83] J. C. Platt, N. Cristianini, and J. Shawe-Taylor. Large margin dags for multiclass classification. In S. A. Solla, T. K. Leen, and K. Müller, editors, *Advances in Neural Information Processing Systems 12*, pages 547–553. MIT Press, 2000.

[84] M. E. Porter. *Competitive strategy : techniques for analyzing industries and competitors*. Free Press, New York, 1980.

[85] D. R. Raban and S. Rafaeli. The effect of source nature and status on the subjective value of information. *JASIST*, 57(3):321–329, 2006.

[86] V. Sachidananda, A. Khelil, and N. Suri. Quality of information in wireless sensor networks: A survey. In *Proceedings of the International Conference on Information Quality*, 2010.

[87] P. Sasikumar and S. Khara. K-means clustering in wireless sensor networks. In *Computational intelligence and communication networks (CICN), 2012 fourth international conference on*, pages 140–144. IEEE, 2012.

[88] R. E. Schapire. The boosting approach to machine learning: An overview. In *Nonlinear estimation and classification*, pages 149–171. Springer, 2003.

[89] J. Simpson, R. F. Adler, and G. R. North. A proposed tropical rainfall measuring mission (TRMM) satellite. *Bulletin of the American meteorological Society*, 69(3):278–295, 1988.

[90] P. Smyth et al. Clustering sequences with hidden markov models. *Advances in neural information processing systems*, pages 648–654, 1997.

[91] E. A. Stohr and J. V. Nickerson. Intra enterprise integration. In J. Luftman, editor, *Competing in the Information Age: Align in the Sand*, pages 227–251. Oxford University Press, New York, 2002.

[92] S. K. Teh, L. Mejias, P. Corke, and W. Hu. Experiments in integrating autonomous uninhabited aerial vehicles(uavs) and wireless sensor networks. In *2008 Australasian Conference on Robotics and Automation (ACRA 08)*, Canberra, 2008. The Australian Robotics and Automation Association Inc.

[93] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[94] G. Tuna, T. V. Mumcu, K. Gulez, V. C. Gungor, and H. Erturk. Unmanned aerial vehicle-aided wireless sensor network deployment system for post-disaster monitoring. In *Emerging Intelligent Computing Technology and Applications*, pages 298–305. Springer, 2012.

[95] L. Villas, D. L. Guidoni, J. Ueyama, et al. 3D localization in wireless sensor networks using unmanned aerial vehicle. In *Network Computing and Applications (NCA), 2013 12th IEEE International Symposium on*, pages 135–142. IEEE, 2013.

[96] E. R. Williams, S. Geotis, N. Renno, S. Rutledge, E. Rasmussen, and T. Rickenbach. A radar and electrical study of tropical hot towers. *Journal of the atmospheric sciences*, 49(15):1386–1395, 1992.

[97] X. Zhu. Semi-supervised learning. In *Encyclopedia of Machine Learning*, pages 892–897. Springer, 2011.