The Pennsylvania State University
The Graduate School

AUTONOMOUS PERCEPTION AND DECISION MAKING IN CYBER-PHYSICAL SYSTEMS

A Dissertation in
Mechanical Engineering
by
Soumik Sarkar

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

August 2011

The dissertation of Soumik Sarkar was reviewed and approved* by the following:

Asok Ray
Distinguished Professor of Mechanical Engineering
Dissertation Adviser, Chair of Committee

Alok Sinha
Professor of Mechanical Engineering

Shashi Phoha
Professor of Electrical Engineering
Division of Information Sciences and Technology, Applied Research Laboratory, Penn State

Mark Levi
Professor of Mathematics

Jainendra Jain
Erwin W. Mueller Professor of Physics

Thomas Wettergren
Senior Technologist, Operational and Information Science, Naval Undersea Warfare Center, United States Navy
Special Member

Karen A. Thole
Professor of Mechanical Engineering
Department Head of Mechanical and Nuclear Engineering

*Signatures are on file in the Graduate School.

Abstract

The cyber-physical system (CPS) is a relatively new interdisciplinary technology area that includes the general class of embedded and hybrid systems. CPSs require integration of computation and physical processes that involves the aspects of physical quantities such as time, energy and space during information processing and control. The physical space is the source of information and the cyber space makes use of the generated information to make decisions. This dissertation proposes an overall architecture of autonomous perception-based decision & control of complex cyber-physical systems.

Perception involves the recently developed framework of Symbolic Dynamic Filtering for abstraction of physical world in the cyber space. For example, under this framework, sensor observations from a physical entity are discretized temporally and spatially to generate blocks of symbols, also called words that form a language. A grammar of a language is the set of rules that determine the relationships among words to build sentences. Subsequently, a physical system is conjectured to be a linguistic source that is capable of generating a specific language. The proposed technology is validated on various (experimental and simulated) case studies that include health monitoring of aircraft gas turbine engines, detection and estimation of fatigue damage in polycrystalline alloys, and parameter identification.

Control of complex cyber-physical systems involve distributed sensing, computation, control as well as complexity analysis. A novel statistical mechanics-inspired complexity analysis approach is proposed in this dissertation. In such a scenario of networked physical systems, the distribution of physical entities determines the underlying network topology and the interaction among the entities forms the abstract cyber space. It is envisioned that the general contributions, made in this dissertation, will be useful for potential application areas such as smart power grids and buildings, distributed energy systems, advanced health care procedures and future ground and air transportation systems.

Table of Contents

List of Figures

Acknowledgments

This work would not have been possible without the guidance, inspiration and dedication of Prof. Asok Ray as a teacher and a mentor. I would like to thank Prof. Alok Sinha, Prof. Shashi Phoha and Prof. Jainendra Jain for their invaluable inputs and for being in my dissertation committee. I would like to extend special thanks to Prof. Mark Levi, my advisor for the M.A. degree in Mathematics and Dr. Thomas Wettergren, senior technologist in the U.S. navy for being in my dissertation committee. I am honored to be around some very talented individuals in my research group. Among them, I would like to mention Mr. Kushal Mukherjee, Dr. Abhishek Srivastav, Dr. Murat Yasar, Dr. Subhadeep Chakraborty, Dr. Chinmay Rao, Dr. Ishanu Chattopadhyay and Dr. Shalabh Gupta for all the insightful discussions and valuable suggestions that moved forward my research. I would also like to thank Mr. Dheeraj Sharan Singh, Mr. Xin Jin, Mr. Anthony Cascone, Dr. Eric Keller and Mr. Yicheng Wen for their help on various occasions. Numerous discussions with our NASA NRA technical monitor, Mr. Donald Simon helped me understand the practical aspects of my dissertation problem; I cordially thank him for the help. I thank my close friends in State College for their support and encouragement over the last five years, who made by graduate study experience fun and memorable. Last, but not the least, I could not have done it without the support of my entire family and all my teachers from my school and undergraduate days. A special thanks goes to my brother Mr. Soumalya Sarkar.

My work and study have been supported in part by NASA under Cooperative Agreement No. NNX07AK49A, the U.S. Office of Naval Research under Grant No. N00014-09-1-0688, and by the U.S. Army Research Office under Grant No. W911NF-07-1-0376.

Dedication

To my parents for their dedication, support and inspiration without expecting anything in return and to my grandfather for enkindling my passion for science and education ...

Chapter 1

Introduction

The industrial revolution of late 18^{th} and early 19^{th} century marked the beginning of a remarkable journey that redefined the life style of human society. The community of engineering physical systems played a central role to develop a sustainable modern civilization driven by technology and innovation. Around hundred and fifty years later, during the mid 20^{th} century, the world witnessed another technical revolution with the advent of computing technology that started changing the life style of human society once again. Today we cannot imagine our world without its cyber component. However, the community of engineering physical systems kept considering computers primarily as fast, powerful and flawless numerical calculators. This consideration simply ignores the issues of memory and energy management, software errors due to system complexity, etc. On the other hand, computer scientists and engineers did not care about physical systems in terms of stability, power consumption, computational requirements, etc [2]. Time has come to build a single unified framework that seamlessly integrates cyber and physical entities that will lead to safer, more efficient and sustainable technologies. To date, digital control systems are the most visible and successful applications of such a framework. However, we are far from realizing the enormous potential of cyber-physical systems (CPSs). With increasing size and complexity of human-engineered systems (e.g., national power grid, large air-traffic control systems, power generating plants with a large number components), the concept of CPS is becoming more and more relevant. Understanding the distributed dynamics of computation and plants in terms of emergent behavior, inducing order, anomaly

detection, robust and resilient control, are some of the crucial challenges for such systems.

1 Background and Literature Survey

With the progress of information science and communication technology, the trend of miniaturizing information processing units has flourished significantly. Furthermore, this advancement helped to embed information processing components inside physical products for in-situ analysis that leads to intelligent automation. Formally, the *embedded systems* can be defined as [3]

Definition 1.1. *Embedded systems are information processing entities embedded into enclosing products.*

Examples of such systems can be found in all aspects of today's world, e.g., in transportation, telecommunication, power and household systems. However, as the system complexity and performance requirements increase in such systems, understanding the physics of the physical components becomes more important for an efficient operation of the cyber components. This aspect drove the technical community towards a more robust and general framework of embedded systems, namely the *cyber-physical systems*. In words, cyber-physical Systems (CPSs) are integrations of computation and physical processes [4] that emphasizes on the aspects of physical quantities such as time, energy and space during information processing and control.

1.1 Cyber-physical System: Definition and Examples

Cyber-physical systems are essentially composed of the physical space (generator of information) and the cyber space (abstract information/event space). Mathematically, an observation of the physical space can be expressed as a three tuple, $O(M, L, T)$. In this definition, M denotes the type of information or modality, L denotes location of the observation in the physical space and T denotes the (fast-scale) time domain of the physical space dynamics. The observations are made by sensors (a physical measurement device, hard or soft, such as an acoustic

transducer or instrumented code) that can be defined as a mapping

$$S : M \times L \times T \rightarrow V \tag{1.1}$$

While, M, L, and T have the same meaning as before, V is a vector space over the real field $\mathbb{R}$. Modern developments in sensor technology make it possible to measure virtually any physical quantity. Apart from classical sensors (e.g., pressure, speed, acceleration, electrical current, voltage, temperature), a wide variety of advanced sensors are helping in fast progress of smart system technology. Such sensors may use different transducers (e.g., magnetic, sonar, acoustic and chemical sensors). However, due to the extreme heterogeneity among sensors, sophisticated information fusion techniques are also required more than ever before. As shown in Fig. 1.1, the information from sensors is processed, interpreted and fused to generate abstract events without losing its semantics. Formally, an event in the information space can be defined as a three tuple as well: $\varepsilon(\Gamma, \mathcal{L}, \tau)$, where Γ denotes the event class; $\mathcal{L}$ denotes the event location; and τ denotes the (slow-scale) time domain of the event dynamics in the cyber space.

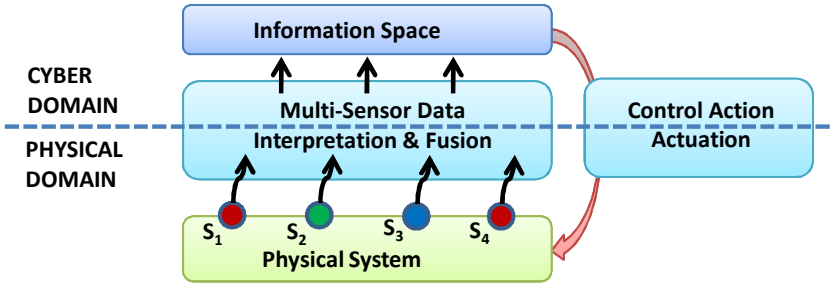


Figure 1.1. Outline of a Cyber-Physical System (CPS)

Potential Application Areas of Cyber-physical Systems

Recently, in a national summit on CPS [2], quite a few potential application areas have been identified, which include: (i) Large-scale autonomous manufacturing processes, (ii) Intelligent sensor networks for monitoring the physical world essentially to avoid/prevent environmental and human threats, (iii) Smart and green buildings, (iv) Advanced robotic applications including disaster management, context-aware help for humans. Few application areas are discussed below.

Energy Technology: CPSs are expected to play a very important role in man-

aging the future energy initiatives. Although around the world, a desperate search for green and renewable energy sources is on, there is no such single candidate to replace fossil fuels yet. Therefore, the smart power grids of near future need to coordinate and control the supply-demand of a multi-source, multi-regional energy distribution. This will require context-aware optimization of energy generation, storage, transmission, and use, to achieve reductions in waste, carbon footprint and costs. On a smaller scale, the CPS technology will also serve green buildings and cars.

Health care Industry: Human body is a physical system that continuously generates information. Therefore autonomous monitoring of human body on a regular basis can prevent critical/fatal diseases (e.g., early diagnosis of cancer). CPS technologies will develop autonomous health care diagnosis and safe/flawless drug delivery (e.g., monitoring blood sugar level and injecting insulin in the blood stream). Furthermore, implants of cyber components in the bodies of paralyzed patients can act as neural prosthesis.

Future Transportation Systems: With advancement of technology and progress of economy around the world, both ground and air-traffic systems are getting more and more stressed. Tremendous congestion in traffic systems not only causes loss of time and productivity, it also results in wastage of valuable energy. Smart transportation systems of future will require extreme advancements in CPS technology enabling robust and resilient control/optimization in real time. Analyzing highly nonlinear and emergent behaviors of complex traffic systems and ensuring safety, stability, reliability and efficiency during control are among the toughest problems in CPS research.

Performance Requirements of Cyber-physical Systems

The core idea of cyber-physical systems is not to consider the physical components as a black-box from the cyber perspective. Therefore, the design process of a cyber-physical system will require in-depth understanding of the physics of the application area. However, the objective functions used for design optimization can be fairly general. Typically, a large number of objective functions need to be considered due the high-complexity and performance expectation of CPSs. Such design objectives include [5] (i) average and worst-case performance, (ii) energy consumption, (iii) safety and reliability, (iv) robustness and inter-operability (v)

cost and environmental friendliness. While high average/worst-case performance and related costs are regarded as general requirements for any designed system, energy consumption, safety and reliability are more important for CPSs due to the implied high degree of autonomy. Other than that, a CPS design should not be completely platform-specific and hazardous to environment. Due to the on-board, real-time requirements of CPSs, fast optimization techniques are required with low-memory requirements. However, instead of combining all the objectives into a single objective function, use of multi-objective optimization techniques should be preferred in order to prevent loss of critical information. That will provide design options as individual points on the Pareto surface [6].

1.2 Modeling approaches for Cyber-physical Sytems

Modeling approaches for CPSs differ in terms of domain characteristics (continuous/discontinuous time), range characteristics (continuous/discontinuous states), operational logic (time-driven/event-driven) and obviously modeling complexity.

Continuous State Systems:

Continuously varying system models are commonly used in modeling physical systems in engineering. In continuous time setting, ordinary and partial differential equations are used. A continuous state dynamical system is defined as $P = (X, U, f)$, where $X \subseteq \mathbb{R}^n$, $n \in \mathbb{N}$ is the state-space dimension, $U \subseteq \mathbb{R}^m$, $m \in \mathbb{N}$ is the input space dimension and f is the continuous mapping. The continuous time system is governed by the differential equation:

$$\dot{x} = f(x, u, t) \tag{1.2}$$

where $t \in \mathbb{R}$ is the time index ($t \geq t_0$) and the initial condition is $x(t_0) = x_0$. When $\dot{x}$ is an explicit function of the independent variable t, then the system is called a non-autonomous system, otherwise if it depends on time implicitly through x and u, then the system is called an autonomous system. $y \in \mathbb{R}^l$ is the l-dimensional output/sensor observation of the system that follows the differential equation:

$$y = g(x, u, t) \tag{1.3}$$

For discrete-time systems (i.e., time index is discrete, $t \in \mathbb{N}$), differential equations are replaced by difference equations.

Discrete State Systems:

Although continuous state modeling is more traditional, its discrete counterpart is more useful in terms of computation in the cyber domain. In this context, the systems exhibit discrete (logical) states and are often known as discrete-event systems (DESs). Ramadge and Wonham [7] initiated the idea of supervisory control of DESs to analyze and control event-driven dynamical systems (asynchronous in time). This approach is especially useful for controlling large human-engineered complex systems, such as networked robotic systems, military command/control missions, distributed power-grids, integrated aircraft systems for enhanced reliability, performance, and operating range [8].

Discrete-event based languages: The discrete event-based model of computation is based on the idea of executing a sequence of discrete events. Event generation is essentially a recursive process as an event is generated as a consequence of other generated events. An event is removed when its corresponding action is performed and another event takes its place in the queue. Hardware description languages (HDLs) are typically based on discrete-event models, e.g., VHDL [9], Verilog [10] and SystemC [11]. In general, from a computational perspective, a formal language is a set of words, i.e., finite strings of letters, symbols, or tokens. The letters are taken from a set, called the alphabet, over which the words of the language are constructed. A formal language is defined by means of a formal grammar that permits certain words to be parts of a formal language. Formal languages play a crucial role in the development of compilers and structure of corresponding machines. Chomsky hierarchy [12] describes the family of formal languages in terms of model complexity. The most general one is called an unrestricted grammar or a recursively enumerable language that is equivalent to von Neumann machines, Turing machines, C++ and Java. Context-sensitive grammars form a subset of unrestricted grammars and are recognized by linear bounded automata. Similarly, context-free grammars (recognized by non-deterministic Push down automata (PDA)) form a subset of context-sensitive grammars and regular grammars (recognized by finite state automata (FSA)) form a subset of context-free grammars. More details on FSA are provided below as this dissertation uses this modeling approach.

Finite state automata (FSA): A finite state automaton is a model (simplest in terms of complexity) composed of states and transitions that can be used as one of several mathematical representations of discrete state systems. In the automata theory, deterministic finite state automata (DFSA) and nondeterministic finite state automata (NFSA) are equivalent in the sense that both generate and recognize only the set of regular languages [12]. Consequently, discrete event dynamical behavior of physical plants (e.g., gas turbine engines [13], nuclear power plants [14]) can be modeled as generators of regular languages that are realized by finite state automata [7, 15].

Mathematically, an FSA is described as 5-tuple $G = (Q, \Sigma, \delta, q_0, Q_m)$, where Q is the finite set of states with $|Q| = n$ and $q_0 \in Q$ is the initial state; Σ is the (finite) alphabet of events with $|\Sigma| = m$; Σ^* is the set of all finite-length strings of events including the empty string ε. The (possibly partial) function $\delta : Q \times \Sigma \to Q$ represents state transitions and $\hat{\delta} : Q \times \Sigma^* \to Q$ is an extension of δ and $Q_m \subseteq Q$ is the set of marked (also known as accepted) states which have some importance (positive or negative) on the model. Given an initial state $q_0 \in Q$, the behavior of the (deterministic) FSA G is a sequence of states such that events trigger the state transitions. The generating language of G is

$$L(G) = \{s \in \Sigma^* | \hat{\delta}(q_0, s) \in Q\} \tag{1.4}$$

and the marked language of G is

$$L_m(G) = \{s \in \Sigma^* | \hat{\delta}(q_0, s) \in Q_m\} \tag{1.5}$$

Many variations of FSA modeling have been used in literature for different purposes. For example, concept of communicating finite state automata has been developed to facilitate collaborative decision making [5]. Also, in order to include timing information, classical automata are extended to timed automata [16]. The concept of probabilistic finite state automata (PFSA) has been used extensively [17] to model dynamical systems for anomaly detection. While FSA determines whether a string is allowed in a language or not, PFSA provides the probability of occurrence of individual strings. Thus, PFSA (as a generator of stochastic regular languages) is able to capture more dynamical characteristics

compared to FSA.

Petri nets: Similar to PFSA, Petri net (PN) models [18] also have more descriptive power than finite state machines in the sense that the set of PN languages is a superset of regular languages and they allow a more concise model description. Petri nets are essentially computational graphs that are efficient in modeling control flows and causal dependencies and admit concurrency. They are especially suited for modeling distributed systems as there is no assumption of global synchronization. The three interconnected ingredients of PN logic are conditions, events and a flow relation. The flow relation describes the conditions in which events can happen and also resulting conditions as consequences of event occurrences.

Hybrid Systems:

It is evident from the discussion thus far that one needs to capture different types of continuous and discrete dynamics to successfully model and control a CPS. In this context, the modeling language has to be *descriptive* (bringing continuous and discrete dynamics together), *composable* (to allow building large model by composition of simple sub-components) and *abstractable* (to allow inter-operability and facilitate design) [19, 20]. Different languages have been developed emphasizing different aspects of the problem in the hybrid systems literature. Also, examples of hybrid systems emerge in many areas including automatic control systems [21, 22], robotic systems [23, 24], intelligent vehicle and highway systems [25, 26]. In line with the earlier discussions, one such language is described here that is known as *hybrid automata* [19].

A hybrid automaton describes the time evolution of a dynamical system with a combination of continuous and discrete state variables. Mathematically, it can be defined as a 8-tuple, $H = (Q, X, f, \chi_0, D, E, G, R)$, where

- $Q = \{q_1, q_2, \cdots\}$ is a set of discrete states. In most applications, Q is assumed to be finite.
- $X \subseteq \mathbb{R}^n$ is a set of continuous states and $\mathcal{P}(X)$ denotes the power set of X.
- $f(\cdot, \cdot) : Q \times X \to \mathbb{R}^n$ is a vector field (similar to that for continuous state systems)
- $\chi_0 \subseteq Q \times X$ is a set of initial states; thus, an initial condition of the system

is denoted by $(q_0, x_0) \in \chi_0$. Similarly, in general a state of H is denoted by $(q, x) \in \chi$.

- $D(\cdot) : Q \to \mathcal{P}(X)$ is a domain that assigns a set of continuous states $D(q) \subseteq \mathbb{R}^n$ to each discrete state $q \in Q$.
- $E \subseteq Q \times Q$ is a set of edges that form pairwise combinations of discrete states.
- $G(\cdot) : E \to \mathcal{P}(X)$ is a guard condition that determine boundaries in the continuous state space with respect to the discrete state space.
- $R(\cdot, \cdot) : E \times X \to \mathcal{P}(X)$ is a reset map that reset the continuous system state upon crossing a boundary in terms of the discrete states.

As an example, starting from an initial condition $(q_0, x_0) \in \chi_0$, the continuous state x evolves according to the differential equation

$$\dot{x} = f(q_0, x) \tag{1.6}$$

Note that the discrete state remains as q_0 while the continuous state evolves starting from x_0. This evolution goes on as long as condition $x \in D(q_0)$ is satisfied. If x reaches the guard $G(q_0, q_1) \subseteq \mathbb{R}^n$ of some edge $(q_0, q_1) \in E$, the discrete state may change to q_1 and the continuous state gets reset by the reset map as $R(q_0, q_1, x) \subseteq \mathbb{R}^n$. After this transition, the above process repeats again.

Concepts of hybrid systems bring the merits of both continuous and discrete sides together and therefore, it is a very powerful tool for modeling CPSs. However, significant research issues still remain in terms of switching, stability etc. In this dissertation, underlying physical systems are abstracted in the cyber space and then modeled using PFSA for analysis and eventually for control.

1.3 Issue of Complexity in Cyber-physical Systems

In general, a system is called complex if its overall behavior may not be identifiable when the behaviors of the individual components are known. In the context of CPS, issues of distributed sensing, computation and control come under the purview of so

called multi-agent systems (MAS). While the next section describes these aspects in detail, issues of complex MAS in terms of modeling emergence and inducing order are discussed here.

Human beings are social animals. We all have our limitations; one can not perform all necessary tasks that need to be done for our prosperity. Yet, everybody contributes as per their limited capacity and the society thrives and progresses as a whole. Social science tries to figure out the underlying mechanism of this fascinating phenomena where conglomeration of simple tasks done by individuals leads to the emergence of a social order. Social insects, such as ants and bees exhibit even more fascinating collaborative behaviors. Despite their limited intelligence, ants collaboratively find out optimal paths from their nest to the food source. By implementing a proper division of labor, bees form complex structured combs which are organized in concentric rings of brood, pollen and honey. Complex systems in nature inspired researchers for more than two decades now, to build multi-agent (where, an agent is an autonomous entity that reacts according to its own perception of the environment) human-engineered systems to solve certain problems, which are otherwise unsolvable or inefficiently solved by monolithic (single agent) systems. Typically, MAS offer computationally inexpensive and faster solution due to their asynchronous and parallel computation capability. Robustness, reliability and adaptivity are among the other compelling features that provide the rationale of using MAS across numerous disciplines of engineering, such as swarm-robotics, network science, distributed data mining, software engineering (artificial life, games), operation research, transportation etc. Also, reusability and multi-functionality nature of the agents are advantageous from the point of view of cost and sustainability. On the other hand, as mentioned earlier, the subject matter of several fields of science, e.g. physics, social science, biology, economics etc. inherently involve multi-agent behavior. These fields of science try to characterize the complex emergent behaviors shown by the multi-agent systems in their respective fields. Although all the above application areas may have completely different underlying systems, similar emerging behaviors are observed across them, that inspires researchers to adapt and exchange ideas among different fields. Thus the study of MAS has been immensely interdisciplinary in nature from its inception.

From the perspective of engineering, the problem of complex systems can be

viewed as a two part problem, namely *(i) modeling emergence (analysis)* and *(ii) inducing order (synthesis)*.

1.3.1 Modeling emergence

Given a system of multiple agents with a specific agent-level behavior and their interaction characteristics, predicting the overall global behavior falls under the realm of the analysis step. In literature, this problem is tackled primarily in two different ways, namely macroscopic modeling and microscopic modeling as discussed in detail in Appendix-B. However, (reliable) average behavior based global analytical models exist for very few problems. Even if they do exist, closed form analytical solution actually exist for only a small subset of such models. Therefore, (monte-carlo) simulation based investigations are very important to understand these complex behaviors. The primary problem in modeling emergence is introduced by the often intractable inter-agent interactions. In this regard, the field of *Statistical Mechanics* presents modeling techniques of such interactions that lead to many reliable models for numerous natural multi-agent systems. Recently, researchers started to use ideas of statistical mechanics to understand complex behaviors of a human-engineered MAS [27]. In the sequel, specific issues related to analysis of MAS, are discussed using concepts of Statistical Mechanics.

Critical Behavior: An emergent behavior of an MAS is known as the critical behavior when, the global system behavior changes drastically with small changes of some system parameters. Thus it is of extreme importance to identify and characterize such behaviors to be able to avoid such phenomena or in some cases to exploit them to advantage. The field of statistical mechanics elaborately explored critical behaviors in natural multi-agent systems, such as liquid-gas phase transition in fluids, magnetic phase transition across curie point temperature etc. Sophisticated tools are developed and important observations are made which are not necessarily specific to any application. As a consequence, statistical mechanics approach can also be used for characterization of critical behavior of many human-engineered complex systems.

Finite Size Scaling: Identification of size scaling laws is another issue for MAS, which is extremely important from the point of view of engineering applications. Size scaling laws characterize the overall system behavior as a function

of number of agents, that is not necessarily straight forward for most systems as a consequence of emergent characteristics. Again, finite size scaling (FSS) ansatz of statistical mechanics is one of the most prominent approach to systematically identify the scaling laws in multi-agent systems.

1.3.2 Inducing order

Given a desired overall global behavior, the challenge of this synthesis step is to determine the agent level behaviors and their interaction characteristics. While, a brief overview of the design and control problem of multi-agent systems is provided below from an engineering perspective, a detail survey of specific techniques is conducted in Appendix-B.

Control of MAS - Centralized vs. Decentralized: Control for inducing desired order in an MAS can be executed either in a centralized or in a decentralized fashion, or a combination of them. Centralized control strategies compute (possibly optimal) control actions for each agent centrally. This procedure may guarantee definite function of each agent while reaching the desired overall goal. However, as the number of agents increases, this method becomes increasingly computationally expensive. Therefore, it is not ideal for real-time control of large MAS. Again, robustness and adaptiveness are among the potential problems of centralized strategies. Malfunction of a single central controller or a communication problem with it, shuts down the whole MAS (single-point failure), thus making the MAS fragile, especially in adverse conditions. Therefore, decentralized control strategies are becoming more and more popular in literature (see Appendix-B). These strategies necessarily alleviate many of the problems of its centralized counterpart. In a decentralized control strategy, each agent computes its own action upon its perception of the (possibly local) environment. The agents may be partially or fully aware of the global objective. Due to low dimensionality and parallel computation capability, these strategies are feasible for real time applications. The most important features of these strategies are their inherent robustness (fault tolerance capability: failure of a few agents do not necessarily shuts down the whole system) and adaptivity (flexibility). However, in most cases a decentralized strategy only guarantees the average behavior of agents, i.e., behavior of a particular agent may not be known a priori. Furthermore, limited existence of closed form

mathematical results (e.g. proofs of stability, convergence etc.) in this field makes it very much dependent on large simulations. On the other hand, as discussed earlier, complete understanding of emergent behavior is still an unsolved (may be even intractable) problem for many types of multi-agent systems.

Adaptation and Learning: For autonomous human engineered multi-agent systems, capability to learn the changing conditions (e.g. change in environment or change in global objective) and adapting to it are two important requirements of the control system. Learning primarily involves situational awareness based change in the controller or agent behavior, where as general adaptation can be triggered by both changing environment or any supervisory command. For single agent systems, sophisticated learning algorithms are well established in artificial intelligence literature (see Appendix-B). However, necessary modifications of those tools for MAS applications are still topics of active research. In general, adaptation of decentralized control strategies based on change in environment and desired objective can be considered as a problem of designing robust and resilient decentralized controllers for MAS. A robust controller should be able to accommodate small uncertainties in both environment and desired goal, where as resilient control is necessary for drastic changes in condition that may also lead to a change in the controller objective function.

2 Motivation and Research Issues

It is evident from the discussion above that complex cyber-physical systems have immense engineering relevance in the context of development of new technology as well as improvement of the existing ones. However, significant research efforts are required before their wide spread use. Irrespective of application areas, following research issues are identified related to complex CPSs [2].

2.1 Information Processing: From Sensor Data to Knowledge

Across all the application areas of CPSs, an important aspect of the problems is to discover knowledge about the system from sensor observation and then use of

the knowledge to adaptively control the underlying physical system. A synergistic combination of cyber and physical entities is required for this step of information processing. Different issues in this area are: sensor data abstraction, feature extraction, pattern discovery, information fusion and classification. New theories and models are required to realize this chain of processing. Also, the effect of sensor properties, including calibration, context information, and real world uncertainties, must be accounted for in developed algorithms. Reliability and trust are also very important for the discovered knowledge as the assessments of the created knowledge will be fed back to the physical layer. Dynamic adaption to changing environment and signal conditions is absolutely necessary to maintain reliability and trust. Adaptation may include increase in sensing rate, activation of additional sensors or sensor types at the physical layer or change in specifications of signal processing unit, feature extractor or classifier at the cyber layer.

2.2 Issue of Modeling and Analysis

CPSs are typically composed of massively heterogeneous components in terms of response time-scale and operational logic. At one end, physical components operate in terms of lower level feedback control models. On other end, cyber components mostly operate in an event-driven setting (asynchronous) that is not coherent with time driven models (synchronous). Since the role of both event and time are critical in a CPS, new hybrid models are required. Another significant modeling challenge is integration of multiple scales of temporal and spatial resolutions within a heterogeneous, context-aware CPS. Efficient abstraction will be key to this approach. The uncertainties in sensing, communications, noise, and physical mobility often affect the cyber design, implementation, and performance to a great extent. The impact of the physical components via these properties is so profound that they should be taken care of at the design level itself while building a CPS. In order to have real-time operability and safety, novel notions of robust control has to be developed to operate in this type of non-deterministic, probabilistic, and delay-induced environment. As explained earlier, handling emergent behaviors of a CPS due to the increased complexity is another major issue. The notion of complexity arises from the fact that many explicit global coupling behaviors of (typically)

multi-agent CPSs cannot be anticipated from the individual behaviors of the components. Therefore, inducing order in a CPS requires fundamental research in science and engineering.

2.3 Distributed Sensing, Computation and Control

Despite the increase in modeling complexity, distributed implementations of CPSs are better in most operations compared to centralized ones in terms of robustness to failure and performance in time-critical missions. This issue has already been discussed in the previous section. Furthermore, distributedness allows in-situ computation and processing of sensed data leading to an efficient utilization of CPS framework. However, many aspects regarding distributed control, sensing, computation and communication are yet to be understood properly. Key issues include: collection of adequate information in a distributed manner, distribution of computation authority among entities, collaborative agreement and decision making. In a networked cyber-physical system, latency in computation and control actuation is a critical problem that is not very apparent in traditional models of sensing, computation and decision making acting typically in low-latency environments. Therefore, fast and reliable communication techniques are very crucial. In order to develop robust and resilient distributed decision making mechanisms, efficient techniques of distributed learning, reinforcement and adaption are needed. Above all as mentioned earlier, distributed systems exhibit complex ensemble behavior, such as global cascading and phase transition. Therefore, new science and theory are needed to design safe, context-aware, robust and resilient distributed systems.

2.4 Issue of Seamless Integration

Recently, the research community is increasingly moving away from the conventional (ultimate) goal of artificial intelligence, i.e., *complete autonomy*. Importance of augmenting human capabilities is getting more and more apparent. Systems with physical contacts include mobile sensor platforms, medical devices, e.g., laparoscopic instruments, protein manipulators, and human exo-skeletons. On the other hand, most systems have to be compatible with human end users. Therefore, research is needed to interface the modeling, control, and adaptation of a

CPS with human influence and perception. One aspect of this problem is to mesh the time scales of system adaptation and human response for seamless integration during on-line real-time operations. Human supervisors are particularly useful in understanding the context of the situation that is essential for optimal operation of a CPS. However, efficient communication of context still remains a difficult problem. Moreover, in the event of any anomalous system behavior, human intervention provides the required resilience, for example in a pilot/autopilot scenario. In environments where CPSs are allowed to operate semi-autonomously, issues of authority management, tolerance of operator neglect come up. In this context, the ultimate research goal is to build a unified modeling framework for a seamless human-CPS integration.

The other integration issue has to do with integration between cyber and physical components that is mentioned before as well. In a general CPS architecture this integration is needed at different levels of the hierarchy with varying levels of resolution and degrees of freedom. Therefore, a unified mathematical framework will be useful to model this hierarchy using algorithms with known performance properties while guaranteeing co-stability of the cyber and the physical system.

3 Research Objectives and Contributions

A key requirement of cyber-physical systems is to be able to assess situations (e.g., environmental condition, system health, and current mission objectives). The capability of extracting patterns of critical physical phenomena from local sensory data and other sources (e.g., text messages, and procedure manuals) will allow subsystems to react to evolving situations in time and affect optimal modification of the control and actuation parameters. This is essentially a framework of perception-based decision & control as depicted in Fig. 1.2.

The architecture exhibits two interacting blocks: (i) layered sensing of the physical and informational system, whose output is a quantified form of situational assessment and supporting information, and (ii) hierarchical hybrid (i.e., coupled continuously-varying and discrete-event) control of joint physical and informational states at different layers. Large amounts of data of possibly disparate datatypes and sources need to be analyzed, and the extracted information effec-

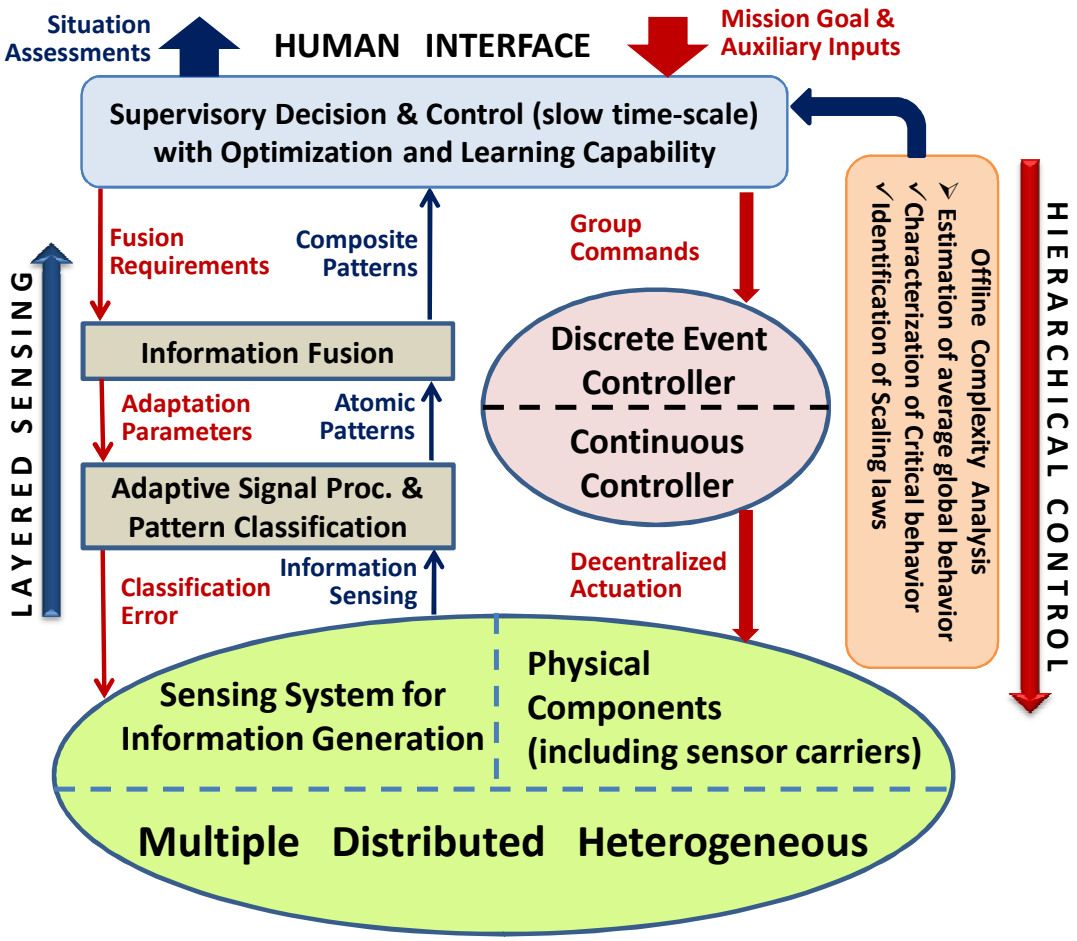


Figure 1.2. Architecture of Perception-based Decision & Control of Complex Cyber-physical Systems

tively fused, in-situ, to identify and discover emerging patterns relevant to the physical reality at hand. During fusion, the atomic patterns should be fused to form composite patterns that capture spatiotemporal dynamics of the environment observed through heterogeneous sensing. Quantification of situational awareness may involve causal graphical modeling among composite patterns inspired from Bayesian Network and Entity Relationship techniques [28]. This crucial task is immensely complicated due to the lack of unified modeling paradigms that allow the use of sensed physical information in existing planning and decision-making algorithms, where acquisition, usage, and sharing of distributed and disparate information are essential for quantification of situational awareness. This explains the call for execution of network-centric Intelligence, Surveillance, and Reconnaissance

(ISR) operations by relying on layered sensing with spatially distributed sensors of different modalities and temporal scales, which are expected to be deployed on heterogeneous platforms in air, land, ocean surface, and undersea environments.

In the hierarchical control scheme depicted in Fig. 1.2, a (possibly) large multi-agent system operates in a decentralized manner at the bottom layer and in the top layer, a supervisor operates to modify/adapt the decentralized behavior based on any situational awareness or change in global objective. It should be noted that physical location wise the supervisor(s) can be placed among the other agents, e.g. for a group or cluster of agents, one of them can be the group leader or cluster heads which will act as a supervisor of the group. However, they will be different in terms of knowledge, perception, information processing capability and authority from the rest of the group. The primary function of the supervisor will be changing the bottom layer group behavior upon a major change in environment and/or change in global objective. The supervisor at the upper level will function in a slower time scale compared to the lower layer. It is obvious that to be able to supervise the bottom layer successfully, the supervisor(s) should have an estimate of the global behavior of the subordinate complex system. From that perspective, off-line analysis information is very important. In general, two important features of the emergent behavior are *Characterization of Critical behavior (if any)* and *Identification of Scaling laws* as discussed earlier.

Based on the overall problem defined above, the objectives and contributions of this dissertation can be broadly divided into two parts that are delineated in the sequel. It is therefore presented in two parts as well. However, the dissertation does not explicitly address the issues of situation assessment, closing the control loop and human interfacing; they will be important future research directions.

3.1 Autonomous Perception: From Sensor Data to Knowledge

The sensor information obtained from the environment as well as about the physical system itself need to be analyzed for the purpose of identifying small changes for local adaptation or large changes for replanning. From the consideration of autonomy, it is important that the small sensing platforms are able to perform

in-situ signal processing and pattern recognition analysis. Also due to possible multi-modality of sensors, the relevant information need to be fused effectively in-situ to capture the emerging patterns of interest. Research is required to satisfy this need of handling large volumes of data, generated by onboard modern sensor suites, with sufficient speed and accuracy. Recent literature showed a promise of solving this problem via transformation of locally observed (and processed) sensor time series data to symbolic sequences over abstract alphabets and subsequently computing formal linguistic representations, known as probabilistic finite state automata (PFSA). As discussed in Section 1, a probabilistic finite state automaton (PFSA) is essentially a finite-state machine with defined event generation probabilities [29]. In a survey paper, Angluin and Smith [30] defined inductive inference as the process of hypothesizing a general rule from examples. A similar philosophy is adopted as a special case where the rule takes the form of a PFSA, and the symbol sequence is assumed to be drawn from a hypothetical stochastic regular language generated from the symbolized sensor time series data. The key advantage of using this symbolic approach accrues from structural simplicity of PFSA and its ability of statistical learning from large volumes of complex data. Conceptually, this is equivalent to learning the internal structure of a black box that is continuously emitting symbols. The resulting PFSA emerge as the most likely generators of the observed sequences [31, 17], and hence this concept can be treated as symbolic abstractions of the underlying causal patterns. The key idea is to compute small-order representations that effectively encode the underlying geometry of recurrence in sensed data. The loss of information due to symbolization can be minimized via proper choice of the partitioning function and it has been shown that this symbolic approach is superior to competing techniques for discovery of incipient anomaly patterns in electronic systems [32]. The appropriate branch of mathematics dealing with such formalisms is that of symbolic dynamics and shift spaces [33]. Efficacy of the this approach, which is called the symbolic dynamic filtering (SDF), stems from the theoretical fact that the states of such constructed PFSA are causal states capturing the causal semantics of the underlying dynamical system. A key assumption in PFSA construction is fast-time-scale statistical stationarity. Thus the stochastic process is assumed to be stationary for the period over which a sequence fragment is collected for such semantic compres-

sion. This assumption is not too restrictive for sufficiently high data acquisition rates and non-stationarity is captured by the evolution of the constructed PFSA over time. Thus each PFSA symbolically encodes the time-series collected over the fast time scale, and the transition probability matrix of the PFSA along with the state connectivity, capture the statistically stationary pattern of the process behavior within that short time interval. It is noted that there exist formalisms of learning probabilistic models that are more powerful than PFSA (e.g., belief (Bayesian) networks [34, 35] and stochastic context-free grammars [36]); however, learning such models proves to be ineffective in the context of the current problem due to the associated computational costs that make real-time execution difficult (if not impossible). In fact, experimental results in [37, 38, 39] show that it is 10 to 100 times faster to learn a pronunciation model for spoken words using a (certain kind of) PFSA than to learn a corresponding hidden Markov model (HMM), and yet the performance of the PFSA is actually slightly better. Also, it is possibly to come up with provably efficient algorithms to optimally learn this kind of PFSA, whereas optimally learning HMMs is provably hard the algorithms used in practice only find a local optimum.

This dissertation focuses on three crucial steps of the PFSA-based SDF approach. To improve the quality of feature extraction, various signal pre-processing techniques are investigated, such as Wavelet transform [40] and Hilbert transform [41]. A new generalization of classical Hilbert transform [42] is developed here for noise rejection and model order reduction. Although modeling of state machines from symbol sequences has been widely reported, similar efforts have not been expended to investigate data partitioning to optimally generate the symbol sequences. This dissertation addresses this issue and proposes a data partitioning procedure to extract low-dimensional features from time series while minimizing the loss of class separability information. Finally, a feature-level semantic information fusion technique is proposed for multi-sensor data interpretation. Therefore, the three main contributions of this part are:

1. *Generalized Hilbert transform for time-series data processing*
2. *Optimization of feature extraction via partitioning*
3. *Extraction of spatiotemporal correlation features*

Part- I (Chapters 2, 3 and 4) of this dissertation presents these contributions in detail along with validation examples and they are summarized as well in Chapter 7.

3.2 Distributed Decision & Control of Multi-agent Complex Systems

Due to the future requirements of large-scale cyber-physical systems, such as smart buildings, smart-grids, traffic systems and multi-asset intelligent surveillance and reconnaissance (ISR) missions, sensing, computation and control have to performed in a distributed fashion. The decentralized approach also brings about robustness and resilience to the system operations. However, with size of the systems and heterogeneity of components, come the problem of complexity. Therefore, a comprehensive analysis of average global behavior of a multi-agent complex system is necessary to identify the stability characteristics, performance metrics, overall system states and their controllability and observability characteristics. Due to complex interaction among components, it is often very difficult to get a straight forward model of such systems. Thus, different approaches are being investigated to model such systems; it will be discussed in Appendix-B. This dissertation adopts developments in statistical mechanics for a better understanding of such distributed systems, especially for characterization of critical behavior (if any) and identification of scaling laws, the two important features of emergent behaviors. In particular, the qualitative nature of phase transitions in communication network systems is characterized via defining network analogs of thermodynamic quantities (e.g., order parameter, temperature, and pressure). Phase transition phenomena are investigated for multiple intensive parameters. Network phase diagrams are constructed based on the network parameters. Such phase diagrams are useful for building control strategies for heterogeneous packet transmission in communication network systems. A survey of commonly used decentralized control algorithms is presented in Appendix-B. Artificial potential based algorithms are among the most popular and successful of such methods. However, there still exists a lot of unaddressed issues (e.g., time-varying potential fields, adaptation, scalability, issue of local minima, and saddle point). Furthermore, guarantee for stability and con-

vergence exists for a limited number of such control methods. Hence, investigation of decentralized control schemes is one of the major focuses of the research presented here. In particular, this dissertation investigates the problem of distributed decision propagation in mobile-agent networks. This essentially address the issue of information management in large scale networks with time-varying topology, that is crucial for problems such as information fusion, trust establishments in mobile ad hoc networks (MANETs) and threat monitoring by mobile sensor networks. There are two aspects of the problem: time varying network topology and distributed agent interaction policy. Two agent interaction policies (one linear and one nonlinear) are studied here. The linear strategy is essentially a generalized gossip algorithm under the language-measure-theoretic framework and the nonlinear strategy is a threshold based policy, known as binary decisions with externalities. Therefore, the two main contributions of this part are:

1. *Emergent behavior modeling in communication networks*
2. *Distributed decision propagation in mobile-agent networks*

Part-II (Chapters 5, and 6) of this dissertation presents these contributions in detail and they are summarized as well in Chapter 7.

4 Organization of the Dissertation

The dissertation is divided into two parts based on the research objectives described in the previous section. Part-I encompasses the contributions made in the area of information processing from sensor data. Brief descriptions of the chapters included in Part-I are provided below.

Part-I: Autonomous Perception: From Sensor Data to Knowledge

- Chapter 2 presents a brief overview of Symbolic Dynamic Filtering (SDF), the basic framework of information extraction used here. As the first step of feature extraction and pattern discovery, a novel generalization of Hilbert transform is proposed here as well.

- The second step in the SDF framework, namely partitioning is studied in Chapter 3. A partitioning optimization framework is formulated and validated in this chapter.

- Chapter 4 focuses on the issues of multi-sensor information fusion. In this context, a semantic method is proposed to extract relational sensor co-dependency patterns.

Part-II primarily deals with the modeling and control aspects of the problem in the setting of multi-agent systems.

Part-II: Distributed Decision & Control of Multi-agent Complex Systems

- Issues of modeling emergence are discussed in Chapter 5 in the setting of communication networks. Concepts of statistical mechanics are used for the analysis presented here.

- In Chapter 6, the distributed control aspect is dealt with. Linear and non-linear decision propagation algorithms are formulated here for mobile-agent networks.

The dissertation is summarized and concluded in Chapter 7 along with a few recommendations of future research directions that can possibly emerge from this research. Finally, there are three appendices; Appendix-A describes the NASA C-MAPSS test bed that is used for validation example in both Chapter 3 and Chapter 4; Appendix-B presents a brief survey to summarize the concepts of multi-agent systems; and Appendix-C provides a brief review of the Language-measure theory that serves as a theoretical background for part of the research presented in Chapter 6.

Part - I
Autonomous Perception:
From Sensor Data to Knowledge

Chapter 2

General Framework of Symbolic Dynamic Filtering

The recently developed statistical pattern recognition tool, called Symbolic Dynamic Filtering (SDF) is built upon the concepts from multiple disciplines including Statistical Mechanics [43, 44], Symbolic Dynamics [33], Statistical Pattern Recognition [45], and Information Theory [46]. It is shown to be a fast, efficient and computationally inexpensive pattern recognition tool. These qualities are particularly important requirements for autonomous cyber-physical system applications as discussed earlier in the introductory chapter. It has been shown by laboratory experimentation [47] that this method of pattern identification yields superior performance in terms of early detection of anomalies and robustness to measurement noise in comparison with other existing techniques such as Principal Component Analysis, (PCA) and Artificial Neural Networks (ANN). While the details are reported as pieces of information in previous publications [17, 43, 48], the essential concepts of space partitioning, symbol generation, and construction of a finite-state machine from the generated symbol sequence are succinctly explained in this chapter.

1 A Brief Overview

A basic consideration of Symbolic Dynamic Filtering technique involves the concept of two time-scales while analyzing time-series data from a given dynamical

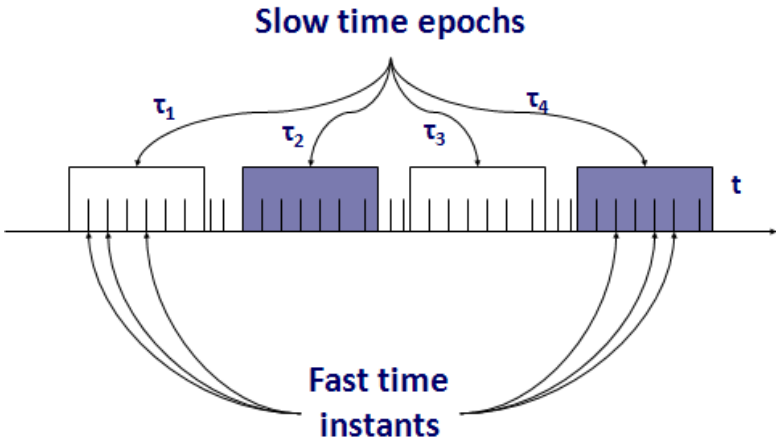


Figure 2.1. Pictorial view of the two time scales: (i) *Slow time scale* of system evolution and (ii) *Fast time scale* for data acquisition and signal conditioning

system. The *fast time scale* is related to the response time of the system dynamics. Over the span of a given time series data sequence, the dynamic behavior of the system is assumed to remain invariant, i.e., the process is quasi-stationary on the fast time scale. On the other hand, the *slow time scale* is related to the time span over which the critical parameters of the system may change and exhibit non-stationary dynamics. The concept of two time scales is illustrated in Fig. 2.1.

In the present context, the fast time scale is related to the time-scale of quasi-stationary behavior of the physical environment, as observed by an agent/sensor. Thus, with the help of SDF, the agent/sensor will be able to construct a concise description in situ of the partially observed environment. On a slower time scale, the environment might evolve. In that case, by pattern classification the agent will be able to distinguish between small and large changes in the environment over different slow time epochs. For small changes, the agent/sensor will try to adapt its behavior in a supervised learning procedure, where as it will inform to its supervisor about large changes. In both cases, the agent/sensor will use the compact probabilistic description of the environment provided by SDF. The method of constructing a compact probabilistic description of a system based on time-series observation from the system is briefly described in the sequel.

2 Symbolic Dynamics Encoding, and State Machine

This section briefly describes the concepts of *Symbolic Dynamics* for:

1. Encoding nonlinear system dynamics from observed time series data for generation of symbol sequences, and

2. Construction of probabilistic finite state automata ($PFSA$) from symbol sequences for generation of pattern vectors as representation of the environment's dynamical characteristics.

The continuously-varying finite-dimensional model of a dynamical system is usually formulated in the setting of an initial value problem as:

$$\frac{d\mathbf{x}(t)}{dt} = f(\mathbf{x}(t), \theta(t_s));\ \mathbf{x}(0) = \mathbf{x}_0, \tag{2.1}$$

where $t \in [0, \infty)$ denotes the (fast-scale) time; $\mathbf{x} \in \mathbb{R}^n$ is the state vector in the phase space; and $\theta \in \mathbb{R}^\ell$ is the (possibly anomalous) parameter vector varying in (slow-scale) time t_s. Let $\Omega \subset \mathbb{R}^n$ be a compact (i.e., closed and bounded) region, within which the trajectory of the dynamical system, governed by Eq. (2.1), is circumscribed. The region Ω is partitioned into a finite number of (mutually exclusive and exhaustive) cells, so as to obtain a coordinate grid. Let the cell, visited by the trajectory at a time instant, be denoted as a random variable taking a symbol value from the alphabet Σ. An orbit of the dynamical system is described by the time series data as: $\mathbb{O} \equiv \{\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \cdots\}$, where each $\mathbf{x}_i \in \Omega$, which passes through or touches one of the cells of the partition. Each initial state $\mathbf{x}_0 \in \Omega$ generates a sequence of symbols defined by a mapping from the phase space into the symbol space as:

$$\mathbf{x}_0 \rightarrow \sigma_{i_0}\sigma_{i_1}\sigma_{i_2}\ldots \tag{2.2}$$

where each $\sigma_{i_k},\ k = 0, 1, \cdots$ takes a symbol from the alphabet Σ. The mapping in Eq. (2.2) is called *Symbolic Dynamics* as it attributes a legal (i.e., physically admissible) symbol sequence to the system dynamics starting from an initial state. The partition is called a generating partition of the phase space Ω if every legal

(i.e., physically admissible) symbol sequence uniquely determines a specific initial condition $\mathbf{x}_0$. In other words, every (semi-infinite) symbol sequence uniquely identifies one continuous space orbit [49].

In physics and signal processing literature, similar procedure is known as coarse-graining or quantization. However, in both cases typically very fine and uniform quantization cells are considered to reduce loss of information. On the other hand, literature of dynamical systems in mathematics presents the concept of partitioning in the context of symbolic dynamics. Unlike in coarse-graining or quantization, partitioning typically involves nonuniform and much coarser representation of the phase space. This leads to a low order modeling of a complex system. Moreover, in case of a generating partition, there is no loss of information. In general though, symbolic dynamics is subjected to (possible) loss of information resulting from granular imprecision of partitioning boxes. However, the essential robust features (e.g., periodicity and chaotic behavior of an orbit) are expected to be preserved in the symbol sequences through an appropriate partitioning of the phase space [50].

Thus partitioning a finite region of the phase space leads to mapping from the partitioned space into the symbol alphabet. This represents a spatial and temporal discretization of the system dynamics defined by the trajectories. Although the theory of phase-space partitioning is well developed for one-dimensional mappings [49], very few results are known for two and higher dimensional systems. Furthermore, the state trajectory of the system variables may be unknown in case of systems for which a model as in Eq. (2.1) is not known or is difficult to obtain. As such, as an alternative, the time series data set of selected observable outputs can be used for partitioning and symbolic dynamic encoding. In general, the time series data can be generated from the available sensors and/or from analytically derived model variables. After partitioning, the symbol sequence is converted into a probabilistic finite-state machine, that is considered as the compressed abstract model of the system.

3 Phase Space Partitioning

As described earlier, a crucial step in symbolic time series analysis is partitioning of the phase space for symbol sequence generation [51]. Several partitioning

techniques have been reported in literature for symbol generation [52][53], primarily based on symbolic false nearest neighbors ($SFNN$). These techniques rely on partitioning the phase space and may become cumbersome and extremely computation-intensive if the dimension of the phase space is large. Moreover, if the time series data is noise-corrupted, then the symbolic false neighbors would rapidly grow in number and require a large symbol alphabet to capture the pertinent information on the system dynamics. Therefore, symbolic sequences as representations of the system dynamics should be generated by alternative methods because phase-space partitioning might prove to be a difficult task in the case of high dimensions and presence of noise. The wavelet or Hilbert transforms largely alleviate the difficulties of phase-space partitioning and are particularly effective with noisy data from high-dimensional dynamical systems [40, 41]. A further theoretical development has been made to generalize Hilbert transform [42] that might prove to be a better choice as a pre-processing technique.

In both of these approaches, time series data are first converted to wavelet domain or analytic signal domain, and the transformed space is then partitioned with alphabet size $|\Sigma|$ into segments. The choice of $|\Sigma|$ depends on specific experiments, noise level and also the available computation power. A large *alphabet* may be noise-sensitive while a small alphabet could miss the details of signal dynamics [40]. The partitioning is done such that the regions with more information are partitioned finer and those with sparse information are partitioned coarser. This is achieved by maximizing the Shannon entropy [46], which is defined as:

$$S = -\sum_{i=1}^{|\Sigma|} p_i \; log(p_i) \tag{2.3}$$

where p_i is the probability of a data point to be in the i^{th} partition segment. In this case, the size of the cells is smaller for regions with higher density of data points to ensures an unbiased partition such that each cell is allocated equal number of visits at the nominal condition. Uniform probability distribution, i.e., $p_i = \frac{1}{|\Sigma|}$ for $i = 1, 2, \ldots |\Sigma|$, is a consequence of maximum entropy partitioning [40]. In the illustrative example of Fig. 2.2, the partitioning contains 4 cells (i.e., line intervals in this case).

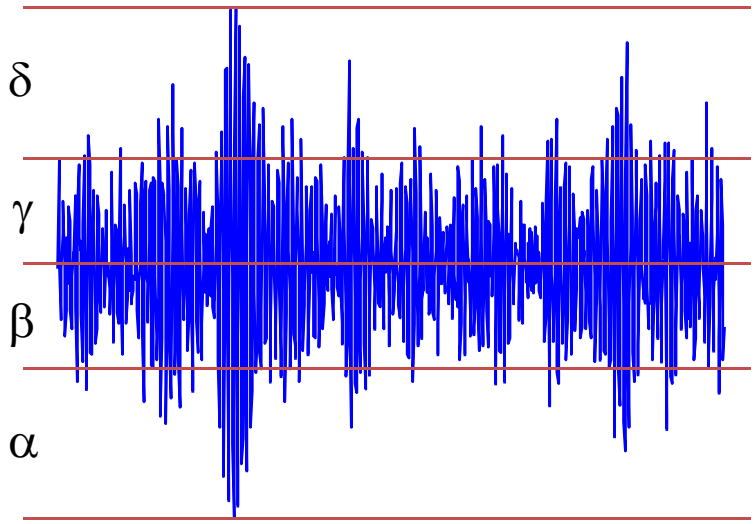


Figure 2.2. An Example of Partitioning

Once the partitioning is done with alphabet size $|\Sigma|$ at a reference slow time scale condition, it is kept constant for all slow time epochs, i.e., the structure of the partition is fixed at the reference condition. Therefore, the partitioning structure generated at the reference condition serve as the reference frame for computation of pattern vectors from time series data at subsequent slow time epochs.

4 Probabilistic Finite State Automata ($PFSA$) Construction

Once the symbol sequence is obtained, the next step is the construction of a Probabilistic Finite State Automaton ($PFSA$) (as shown in Fig. 2.3) and calculation of the respective state probability vector. The partitioning (see Fig. 2.2) is performed at the reference slow time epoch. A $PFSA$ is then constructed, where the states of the machine are defined corresponding to a given *alphabet* set Σ and window length D. This explicit state construction method helps defining abstract states for systems without considerable domain knowledge. The alphabet size $|\Sigma|$ is the total number of partition segments while the window length D is the length of

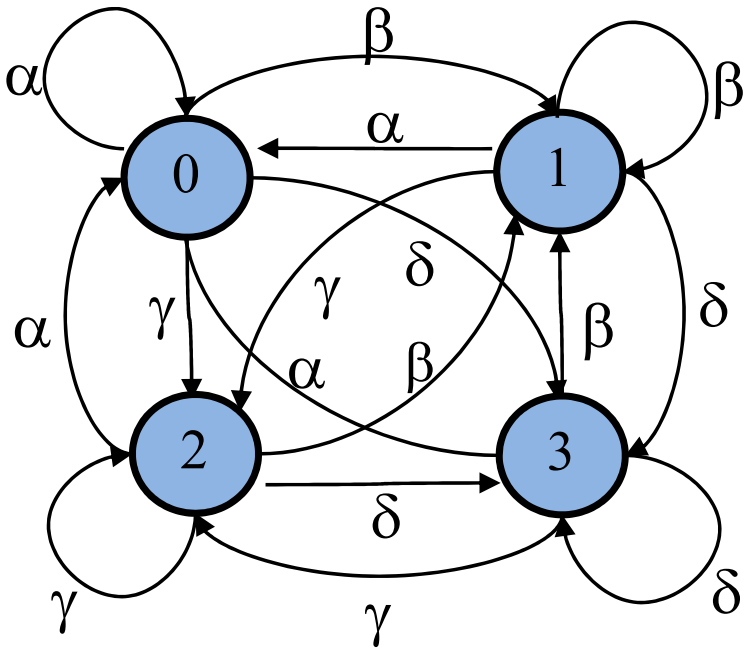


Figure 2.3. Example of Finite State Automaton

consecutive symbol words [17], which are chosen as all possible words of length D from the symbol sequence. Each state belongs to an equivalence class of symbol words of length D, which is characterized by a word of length D at the leading edge. Therefore, the number n of such equivalence classes (i.e., states) is less than or equal to the total permutations of the alphabet symbols within words of length D. That is, $n \leq |\Sigma|^D$; some of the states may be forbidden with zero probability of occurrence. For example, if $\Sigma = \{0, 1\}$, i.e., $|\Sigma| = 2$ and if $D = 2$, then the number of states is $n \leq |\Sigma|^D = 4$; and the possible states are $00, 01, 10$, and 11.

The choice of $|\Sigma|$ and D depends on specific applications and the noise level in the time series data as well as on the available computation power and memory availability. As stated earlier, a large *alphabet* may be noise-sensitive and a small alphabet could miss the details of signal dynamics. Similarly, while a larger value of D is more sensitive to signal distortion, it would create a much larger number

of states requiring more computation power and increased length of the data sets. For the analysis of data sets in this research, the window length is set to D=1; consequently, the set of states Q is equivalent to the symbol alphabet Σ. With the selection of the parameters D=1 and $|\Sigma|$=8, the $PFSA$ has only 8 states, which yields very fast computation and low memory requirements; hence, the algorithm is capable of early detection of anomalies and incipient faults. However, other applications, such as two-dimensional image processing, may require larger values of the parameter D and hence possibly larger number of states in the $PFSA$.

Using the symbol sequence generated from the time series data, the state machine is constructed on the principle of sliding block codes [33]. The window of length D on the symbol sequence $\ldots \sigma_{i_1}\ \sigma_{i_2} \ldots \sigma_{i_k} \ldots$ is shifted to the right by one symbol, such that it retains the last (D-1) symbols of the previous state and appends it with the new symbol σ_{i_ℓ} at the end. The symbolic permutation in the current window gives rise to a new state. The $PFSA$ constructed in this fashion is called the D-Markov machine [17], because of its Markov properties.

Definition 4.1. *A symbolic stationary process is called D-Markov if the probability of the next symbol depends only on the previous D symbols, i.e.,*

$$P\left(\sigma_{i_0}|\sigma_{i_{-1}}....\sigma_{i_{-D}}\sigma_{i_{-D-1}}....\right) = P\left(\sigma_{i_0}|\sigma_{i_{-1}}....\sigma_{i_{-D}}\right).$$

The finite state machine constructed above has D-Markov properties because the probability of occurrence of symbol σ_{i_ℓ} on a particular state depends only on the configuration of that state, i.e., the previous D symbols. Once the alphabet size $|\Sigma|$ and word length D are determined at the nominal condition (i.e., time epoch t_0), they are kept constant for all other slow time epochs. That is, the partitioning and the state machine structure generated at the reference condition serve as the reference frame for data analysis at subsequent slow time epochs.

The states of the machine are marked with the corresponding symbolic word permutation and the edges joining the states indicate the occurrence of a symbol σ_{i_ℓ}. The occurrence of a symbol at a state may keep the machine in the same state or move it to a new state.

Definition 4.2. *The probability of transitions from state q_j to state q_k belonging*

to the set Q of states under a transition $\delta : Q \times \Sigma \to Q$ is defined as

$$\pi_{jk} = P\left(\sigma \in \Sigma \mid \delta(q_j, \sigma) \to q_k\right); \sum_k \pi_{jk} = 1; \tag{2.4}$$

Thus, for a D-Markov machine, the irreducible stochastic matrix $\mathbf{\Pi} \equiv [\pi_{ij}]$ describes all transition probabilities between states such that it has at most $|\Sigma|^{D+1}$ nonzero entries. The left eigenvector $\mathbf{p}$ corresponding to the unit eigenvalue of $\mathbf{\Pi}$ is the state probability vector under the (fast time scale) stationary condition of the dynamical system [17].

On a given symbol sequence $....\sigma_{i_1}\sigma_{i_2}...\sigma_{i_l}....$ generated from the time series data collected at a slow time epoch, a window of length D is moved by keeping a count of occurrences of word sequences $\sigma_{i_1}\cdots\sigma_{i_D}\sigma_{i_{D+1}}$ and $\sigma_{i_1}\cdots\sigma_{i_D}$ which are respectively denoted by $N(\sigma_{i_1}\cdots\sigma_{i_D}\sigma_{i_{D+1}})$ and $N(\sigma_{i_1}\cdots\sigma_{i_D})$. Note that if $N(\sigma_{i_1}\cdots\sigma_{i_D}) = 0$, then the state $q \equiv \sigma_{i_1}\cdots\sigma_{i_D} \in Q$ has zero probability of occurrence. For $N(\sigma_{i_1}\cdots\sigma_{i_D}) \neq 0$, the transitions probabilities are then obtained by these frequency counts as follows:

$$\begin{aligned} \pi_{jk} \equiv P(q_k|q_j) = \frac{P(q_k, q_j)}{P(q_j)} &= \frac{P(\sigma_{i_1}\cdots\sigma_{i_D}\sigma)}{P(\sigma_{i_1}\cdots\sigma_{i_D})} \\ \Rightarrow \pi_{jk} &\approx \frac{N(\sigma_{i_1}\cdots\sigma_{i_D}\sigma)}{N(\sigma_{i_1}\cdots\sigma_{i_D})} \end{aligned} \tag{2.5}$$

where the corresponding states are denoted by $q_j \equiv \sigma_{i_1}\sigma_{i_2}\cdots\sigma_{i_D}$ and $q_k \equiv \sigma_{i_2}\cdots\sigma_{i_D}\sigma$.

The time series data at the reference condition, set as a benchmark, generates the *state transition matrix* $\mathbf{\Pi}$ that, in turn, is used to obtain the *state probability vector* $\mathbf{q}$ whose elements are the stationary probabilities of the state vector, where $\mathbf{q}$ is the left eigenvector of $\mathbf{\Pi}$ corresponding to the (unique) unit eigenvalue. The state probability vector $\mathbf{p}$ is obtained from time series data at a (possibly) faulty condition. The partitioning of time series data and the state machine structure should be the same in both cases but the respective state transition matrices could be different.

Pattern changes may take place in dynamical systems over slow time epochs due to various reasons. The pattern changes are quantified as deviations from the reference pattern (i.e., the probability distribution at the reference condition).

To identify variation of a single parameter in the dynamical system, the resulting anomalies (i.e., deviations of the evolving patterns from the reference pattern) are characterized by a scalar-valued function, called *anomaly measure* μ. The anomaly measures are obtained as:

$$\mu \equiv d\left(\mathbf{p}, \mathbf{q}\right) \tag{2.6}$$

where the $d(\bullet, \bullet)$ is an appropriately defined distance function. An extension of this method for identification of multiple parameters in dynamical systems has also been made in [54]. Instead of using the low dimensional state probability vector as a pattern, the entire Perron-Frobenius operator (state transition matrix) may be used as the pattern to reduce loss of information.

5 Stopping Rule for Determining Symbol Sequence Length

This section presents a stopping rule that is necessary to find a lower bound on the length of symbol sequence required for parameter identification of the stochastic matrix $\mathbf{\Pi}$. The stopping rule [55] is based on the properties of irreducible stochastic matrices [56]. The state transition matrix, constructed at the r^{th} iteration (i.e., from a symbol sequence of length r), is denoted as $\mathbf{\Pi}(r)$ that is an $n \times n$ irreducible stochastic matrix under stationary conditions. Similarly, the state probability vector $\mathbf{p}(r) \equiv [p_1(r)\ p_2(r)\ \cdots\ p_n(r)]$ is obtained as

$$p_i(r) = \frac{r_i}{\sum_{j=1}^{n} r_i} \tag{2.7}$$

where r_i is the number of symbols in the i^{th} state such that $\sum_{i=1}^{n} r_i = r$ for a symbol sequence of length r. The stopping rule makes use of the Perron-Frobenius Theorem [56] to establish a relation between the vector $\mathbf{p}(r)$ and the matrix $\mathbf{\Pi}(r)$. Since the matrix $\mathbf{\Pi}(r)$ is stochastic and irreducible, there exists a unique eigenvalue $\lambda = 1$ and the corresponding left eigenvector $\mathbf{p}(r)$ (normalized to unity in the sense of absolute sum). The left eigenvector $\mathbf{p}(r)$ represents the state probability vector, provided that the matrix parameters have converged after a sufficiently large number of iterations. That is, under the hypothetical arbitrarily long sequences,

the following condition is assumed to hold.

$$\mathbf{p}(r+1) = \mathbf{p}(r)\mathbf{\Pi}(r) \quad \Rightarrow \quad \mathbf{p}(r) = \mathbf{p}(r)\mathbf{\Pi}(r) \;\; \text{as } r \to \infty \tag{2.8}$$

Following Eq. (2.7), the absolute error between successive iterations is obtained such that

$$\| \left(\mathbf{p}(r) - \mathbf{p}(r+1)\right) \|_\infty = \| \, \mathbf{p}(r) \left(\mathbf{I} - \mathbf{\Pi}(r)\right) \|_\infty \leq \frac{1}{r} \tag{2.9}$$

where $\| \bullet \|_\infty$ is the max norm of the finite-dimensional vector $\bullet$.

To calculate the stopping point r_{stop}, a tolerance of η $(0 < \eta \ll 1)$ is specified for the relative error such that:

$$\frac{\| \left(\mathbf{p}(r) \; - \; \mathbf{p}(r+1)\right) \|_\infty}{\| \left(\mathbf{p}(r)\right) \|_\infty} \leq \eta \quad \forall \; r \geq r_{stop} \tag{2.10}$$

The objective is to obtain the least conservative estimate for r_{stop} such that the dominant elements of the probability vector have smaller relative errors than the remaining elements. Since the minimum possible value of $\| \left(\mathbf{p}(r)\right) \|_\infty$ for all r is $\frac{1}{n}$, where n is the dimension of $\mathbf{p}(r)$, the least of most conservative values of the stopping point is obtained from Eqs. (2.9) and (2.10) as:

$$r_{stop} \equiv int\left(\frac{n}{\eta}\right) \tag{2.11}$$

where $int(\bullet)$ is the integer part of the real number $\bullet$.

6 Specific Advantages

The major advantages of SDF for detecting both small and large changes in a dynamical system are listed below:

- Robustness to measurement noise and spurious signals [40]
- Adaptability to low-resolution sensing due to the coarse graining in space partitions [17]
- Capability for early detection of anomalies because of sensitivity to signal distortion [57]

- real-time execution on commercially available inexpensive platforms [58][57].

Various experimental and simulation based validation of this technique have been shown for different applications, including early detection of fatigue crack in metals [57], fault detection in aircraft gas turbine engines [48, 59], coal gasifiers [60], electric motors [61], parameter identification in active electronic circuits [32], robotic pattern identification [62] etc.

7 Data Pre-processing to Facilitate Partitioning

Recently, Subbu and Ray [41] have reported an application of Hilbert transform for symbolic time series analysis of dynamical systems where the space of analytic signals, derived from real-valued time-series data, is partitioned for symbol sequence generation. This method, called analytic signal space partitioning (*ASSP*), is comparable or superior to other partitioning techniques, such as symbolic false nearest neighbor partitioning (*SFNNP*) [63] and wavelet-space partitioning (*WSP*) [40, 64], in terms of performance, complexity and computation time. A major shortcoming of *SFNNP* is that the symbolic false neighbors rapidly grow in number for noisy data and may erroneously require a large symbol alphabet to capture pertinent information on the system dynamics. The wavelet transform largely alleviates these shortcomings and thus *WSP* is particulary effective for noisy data from high-dimensional dynamical systems. However, WSP has several other shortcomings such as identification of an appropriate basis function, selection of appropriate scales, and non-unique and lossy conversion of the two-dimensional scale-shift wavelet domain to a one-dimensional domain of scale-series sequences [40].

When applied to symbolic analysis in dynamical systems, *ASSP* is used to formulate a probabilistic finite-state model, called the D-Markov model [17], where the machine states are symbol blocks of depth D. For noisy systems, it is expected that modeling with a large D in the D-Markov machine would result in higher gain in information on the system dynamics. However, a large D increases the number of machine states, which in turn degrades computation efficiency (e.g., increased execution time and memory requirements) [47].

This section introduces a generalization of the classical Hilbert transform to modify *ASSP* for application to noisy systems. The objective here is to partition

the transformed signal space such that D-Markov machines can be constructed with a small D without significant loss of information for noisy signals. The key idea is to provide a mathematical structure of the generalized Hilbert transform such that the low-frequency region is more heavily weighted than that in the classical Hilbert transform. In this context, theoretical results are derived based on the concepts of information theory. These results are validated on time series data, generated from a laboratory apparatus of nonlinear electronic systems.

8 Generalization of Hilbert Transform

Hilbert transform and the associated concept of analytic signals, introduced by Gabor [65], have been widely adopted for time-frequency analysis in diverse applications of signal processing. Hilbert transform [66] of a real-valued signal $x(t)$ is defined as:

$$\widetilde{x}(t) \triangleq \mathcal{H}[x](t) = \frac{1}{\pi} \int_{\mathbb{R}} \frac{x(\tau)}{t-\tau} d\tau \tag{2.12}$$

That is, $\widetilde{x}(t)$ is the convolution of $x(t)$ with $\frac{1}{\pi t}$ over $\mathbb{R} \triangleq (-\infty, \infty)$, which is represented in the Fourier domain as:

$$\widehat{\widetilde{x}}(\omega) = -i \; sgn(\omega) \; \widehat{x}(\omega) \tag{2.13}$$

where $\widehat{x}(\omega) \triangleq \mathcal{F}[x](\omega)$ and $sgn(\omega) \triangleq \begin{cases} +1 & \text{if } \omega > 0 \\ -1 & \text{if } \omega < 0 \end{cases}$

Given the Hilbert transform of a real-valued signal $x(t)$, the complex-valued analytic signal [66] is defined as:

$$\mathcal{X}(t) \triangleq x(t) + i \; \widetilde{x}(t) \tag{2.14}$$

and the (real-valued) transfer function with input $\widehat{x}(\omega)$ and output $\widehat{\mathcal{X}}(\omega)$ is formulated as:

$$G(\omega) \triangleq \frac{\widehat{\mathcal{X}}(\omega)}{\widehat{x}(\omega)} = 1 + sgn(\omega) \tag{2.15}$$

Lohmann et al. [67] introduced the concept of a generalized Hilbert transform in the fractional Fourier space instead of the conventional Fourier space; a discrete version of this generalized Hilbert transform was developed later [68]. For geophysical applications, Luo et al. [69] proposed another type of generalized Hilbert transform that is essentially the windowed version of traditional Hilbert transform. However, the notion of generalization of Hilbert transform presented in the sequel is different from that in the previously published literature.

Let us define a generalized Hilbert transform as: $\mathcal{H}^\alpha$ of a real-valued signal $x(t)$ as the convolution:

$$\widetilde{x}^\alpha(t) \triangleq \mathcal{H}^\alpha[x](t) = x(t) * \left(\frac{sgn(t)}{\pi\ |t|^\alpha}\right) \quad \text{for } \alpha \in (0,1] \tag{2.16}$$

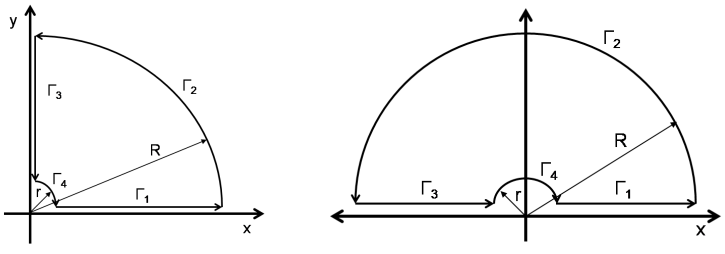
It is shown in the sequel that, as $\alpha \uparrow 1$ (i.e., the values of α form an increasing sequence of positive real numbers with the limit equal to 1), $\mathcal{H}^\alpha$ converges to $\mathcal{H}$, where $\mathcal{H}$ is the classical Hilbert transform defined in Eq. (2.12); that is, $\mathcal{H}^1 \equiv \mathcal{H}$.

Two lemmas are presented, which are necessary for derivation of the main results in the Fourier space.

Lemma 8.1.

$$\int_{-\infty}^{\infty} \frac{e^{-i\omega t}}{\pi|t|^\alpha} sgn(t)dt = -i\ sgn(\omega)\frac{2}{\pi}\frac{\Gamma(1-\alpha)}{|\omega|^{1-\alpha}} sin\left(\frac{\pi}{2}(1-\alpha)\right) \tag{2.17}$$

where $\alpha \in (0,1)$*; and* $\Gamma(1-\alpha) \triangleq \int_0^\infty \frac{e^{-y}}{y^\alpha} dy$.



(a) $0 < \alpha < 1$ (b) $\alpha = 1$

Figure 2.4. Contours of integration paths for generalized Hilbert transform

Proof. The lemma is proved by integration around a quarter circular contour as shown in Fig. 2.4(a). Following Cauchy residual theorem, if z is a complex number, then

$$\int_{\Gamma} \frac{e^{iz}}{z^{\alpha}} dz = 0 \tag{2.18}$$

where the closed contour Γ contains no residues and consists of the paths Γ_1, Γ_2, Γ_3, and Γ_4. As $R \to \infty$ and $r \to 0$, the integral around Γ_2 and the integral around Γ_4 goes to 0, respectively. Therefore,

$$\int_{\Gamma_1} \frac{e^{iz}}{z^{\alpha}} dz + \int_{\Gamma_3} \frac{e^{iz}}{z^{\alpha}} dz = 0 \tag{2.19}$$

$$\Rightarrow \int_{0}^{\infty} \frac{e^{ix}}{x^{\alpha}} dx + \int_{\infty}^{0} \frac{e^{-y}}{(iy)^{\alpha}} dy = 0 \tag{2.20}$$

The following two equations are derived from Eq. (2.20).

$$\int_{0}^{\infty} \frac{e^{ix}}{x^{\alpha}} dx = e^{i\frac{\pi}{2}(1-\alpha)} \Gamma(1-\alpha) \tag{2.21}$$

$$\int_{0}^{\infty} \frac{e^{-ix}}{x^{\alpha}} dx = e^{-i\frac{\pi}{2}(1-\alpha)} \Gamma(1-\alpha) \tag{2.22}$$

Now,

$$\int_{\infty}^{\infty} \frac{e^{-i\omega t}}{|t|^{\alpha}} sgn(t) dt = \int_{0}^{\infty} \frac{e^{-i\omega t}}{t^{\alpha}} dt - \int_{-\infty}^{0} \frac{e^{-i\omega t}}{|t|^{\alpha}} dt \tag{2.23}$$

For $\omega > 0$, it follows from Eq. (2.22) that

$$\int_{0}^{\infty} \frac{e^{-i\omega t}}{t^{\alpha}} dt = \frac{\Gamma(1-\alpha)}{|\omega|^{1-\alpha}} e^{-i\frac{\pi}{2}(1-\alpha)} \tag{2.24}$$

and using Eq. (2.21)

$$\int_{-\infty}^{0} \frac{e^{-i\omega t}}{|t|^{\alpha}} dt = \frac{\Gamma(1-\alpha)}{|\omega|^{1-\alpha}} e^{i\frac{\pi}{2}(1-\alpha)} \tag{2.25}$$

Similarly, for $\omega < 0$

$$\int_{0}^{\infty} \frac{e^{-i\omega t}}{t^{\alpha}} dt = \frac{\Gamma(1-\alpha)}{|\omega|^{1-\alpha}} e^{i\frac{\pi}{2}(1-\alpha)} \tag{2.26}$$

$$\int_{-\infty}^{0} \frac{e^{-i\omega t}}{|t|^{\alpha}} dt = \frac{\Gamma(1-\alpha)}{|\omega|^{1-\alpha}} e^{-i\frac{\pi}{2}(1-\alpha)} \tag{2.27}$$

It follows from Eqs. (2.24), (2.25), (2.26), and (2.27) that

$$\begin{aligned} &\int_{-\infty}^{\infty} \frac{e^{(-i\omega t)}}{|t|^{\alpha}} sgn(t) dt \\ &\quad = \begin{cases} \frac{\Gamma(1-\alpha)}{|\omega|^{1-\alpha}} [e^{-i\frac{\pi}{2}(1-\alpha)} - e^{i\frac{\pi}{2}(1-\alpha)}] & \text{if } \omega > 0 \\ \frac{\Gamma(1-\alpha)}{|\omega|^{1-\alpha}} [e^{i\frac{\pi}{2}(1-\alpha)} - e^{-i\frac{\pi}{2}(1-\alpha)}] & \text{if } \omega < 0 \end{cases} \end{aligned} \tag{2.28}$$

$$\begin{aligned} \Rightarrow &\int_{-\infty}^{\infty} \frac{e^{(-i\omega t)}}{\pi|t|^{\alpha}} sgn(t) dt \\ &\quad = -i\frac{2}{\pi} sgn(\omega) \frac{\Gamma(1-\alpha)}{|\omega|^{1-\alpha}} sin\big(\frac{\pi}{2}(1-\alpha)\big) \end{aligned} \tag{2.29}$$

□

Given $\Gamma(z) = \int_0^{\infty} t^{(z-1)} e^{-t} dt$ for $Re(z) > 0$, it follows that $0 < \Gamma(1-\alpha) < \infty$ for $\alpha \in (0,1)$.

Lemma 8.2. *As $\alpha \uparrow 1$, the integral $\int_{-\infty}^{\infty} \frac{e^{-i\omega t}}{\pi|t|^{\alpha}} sgn(t) dt \to -i\ sgn(\omega)$, i.e.,*

$$\lim_{\alpha \uparrow 1} \Gamma(1-\alpha)\ \Big(\frac{2}{\pi} sin\frac{\pi}{2}\ (1-\alpha)\Big) = 1. \tag{2.30}$$

Proof. Let $\theta \triangleq (1-\alpha)$ and $Z \triangleq \frac{2}{\pi} sin\big(\frac{\pi\theta}{2}\big)\Gamma(\theta)$. Then, for $\theta \in (0,1)$, it follows that

$$\begin{aligned} \lim_{\theta \to 0} Z &= \lim_{\theta \to 0} \frac{\Gamma(\theta) 2 sin\frac{\pi\theta}{2} cos\frac{\pi\theta}{2}}{\pi cos\frac{\pi\theta}{2}} \\ &= \frac{\lim_{\theta \to 0} \Gamma(\theta) sin(\pi\theta)}{\pi \lim_{\theta \to 0} cos\frac{\pi\theta}{2}} \\ &= \frac{\lim_{\theta \to 0} \frac{\pi}{\Gamma(1-\theta)}}{\pi} = 1 \end{aligned} \tag{2.31}$$

because $\forall z \in \mathbb{C}, \Gamma(z)\Gamma(1-z) = \frac{\pi}{sin(\pi z)}$ and $\Gamma(1) = 1$. □

It follows from the proofs of the two lemmas that the integration path changes

from a quarter circular contour to a half circular contour, as seen in Fig. 2.4(a) and Fig. 2.4(b), as $\alpha \uparrow 1$.

Taking Fourier transform of the convolution in Eq. (2.16) and an application of Lemma 8.1 yield:

$$\begin{aligned}\widehat{\widetilde{x}^{\alpha}}(\omega) &= \mathcal{F}\left(x(t) * \frac{sgn(t)}{\pi|t|^{\alpha}}\right) \\ &= \mathcal{F}(x(t)) \cdot \mathcal{F}\left(\frac{sgn(t)}{\pi|t|^{\alpha}}\right) \\ &= -i\ sgn(\omega)\ \frac{2}{\pi}\ \frac{\hat{x}(\omega)\Gamma(1-\alpha)}{|\omega|^{1-\alpha}}\ sin\left(\frac{\pi}{2}(1-\alpha)\right) \end{aligned} \tag{2.32}$$

Since $\Gamma(1-\alpha) < \infty$ for $\alpha \in (0,1)$, the generalized Hilbert transform $\widetilde{x}^{\alpha}(t)$ can be evaluated by taking the inverse Fourier transform of $\widehat{\widetilde{x}^{\alpha}}(\omega)$.

The above formulation shows that reduced α puts more weight on the low frequency part of the signal $x(t)$ and hence more effectively attenuates the high-frequency noise than the classical Hilbert transform. Following Lemma 8.2, as $\alpha \uparrow 1$, Fourier transform of the signal $\frac{sgn(t)}{\pi|t|^{\alpha}}$ converges to $-i\ sgn(\omega)$. This leads to the fact, that as $\alpha \uparrow 1$, $\mathcal{H}^{\alpha}$ converges to $\mathcal{H}$, where $\mathcal{H}$ is the classical Hilbert transform defined in Eq. (2.12).

Analogous to the analytic signal in Eq. (2.14), the (complex-valued) generalized analytic signal of the real-valued signal $x(t)$ is defined as:

$$\mathcal{X}^{\alpha}(t) \triangleq x(t) + i\ \widetilde{x}^{\alpha}(t) \tag{2.33}$$

and the (real-valued) transfer function with input $\widehat{x}(\omega)$ and output $\widehat{\mathcal{X}}^{\alpha}(\omega)$ is formulated as:

$$G^{\alpha}(\omega) \triangleq \frac{\widehat{\mathcal{X}}^{\alpha}(\omega)}{\widehat{x}(\omega)} = 1 + sgn(\omega)\ \left(\frac{\omega_0(\alpha)}{|\omega|}\right)^{(1-\alpha)} \tag{2.34}$$

where $\omega_0(\alpha) \triangleq \left(\frac{2}{\pi\ \Gamma(1-\alpha)} sin\left(\frac{\pi}{2}(1-\alpha)\right)\right)^{\frac{1}{1-\alpha}}$.

Remark 8.1. *For $\alpha = 1$, it follows from Eq. (2.16) that the real-valued signal $x(t)$*

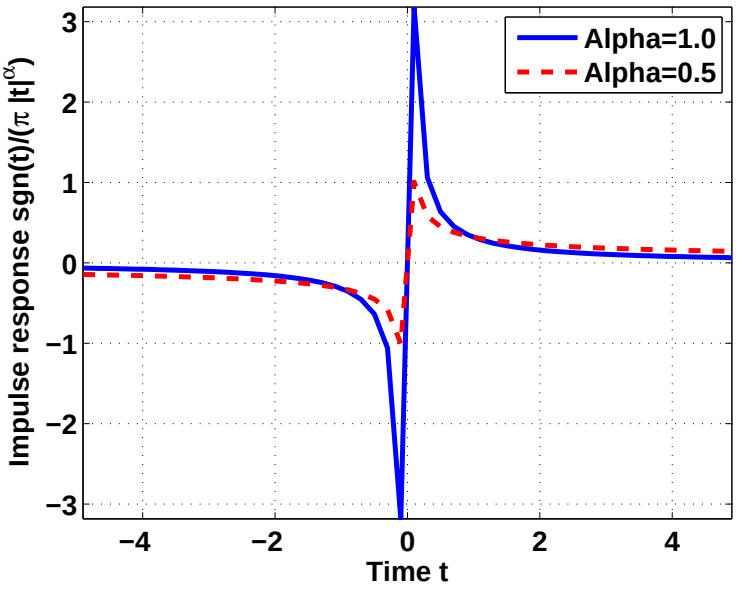


Figure 2.5. Impulse response $\left(\frac{sgn(t)}{\pi|t|^{\alpha}}\right)$ of the generalized Hilbert transform

is convoluted with $\frac{1}{\pi t}$. *The implication is that the effects of memory in the signal* $x(t)$ *reduce as fast as* $\frac{1}{\pi|t|}$. *As* α *is decreased, the tail of the impulse response of the generalized Hilbert transform* $\widetilde{x}^{\alpha}(t)$ *becomes increasingly fat as seen in Fig. 2.5. Hence, for* $0 < \alpha < 1$, *the generalized analytic signal* $\mathcal{X}^{\alpha}(t)$ *captures more (low-frequency) information from time series data than that for* $\alpha = 1$.

Remark 8.2. *Fourier transform of a real-valued signal does not contain any additional information beyond what is provided by the positive frequency components, because of the symmetry of its spectrum. Therefore, in the construction of an analytic signal in Eq. (2.14) and its transfer function in Eq. (2.15), Hilbert transform removes the negative frequency components while doubling the positive frequency components. For* $\alpha < 1$, *it follows from Fig. 2.6 that the negative frequency components of the transfer function* $G^{\alpha}(\omega)$ *of a generalized analytic signal are no longer zero. Therefore, the generalized analytic signal in Eq. (2.33) is not an analytic signal in the sense of Gabor [65] for* $\alpha < 1$. *However, the transfer functions of both analytic and generalized analytic signals are real-valued almost everywhere in the range* $\omega \in \mathbb{R}$. *The phase of the (real-valued) transfer function* $G^{\alpha}(\omega)$ *is either* 0 *or* $-\pi$ *as explained below.*

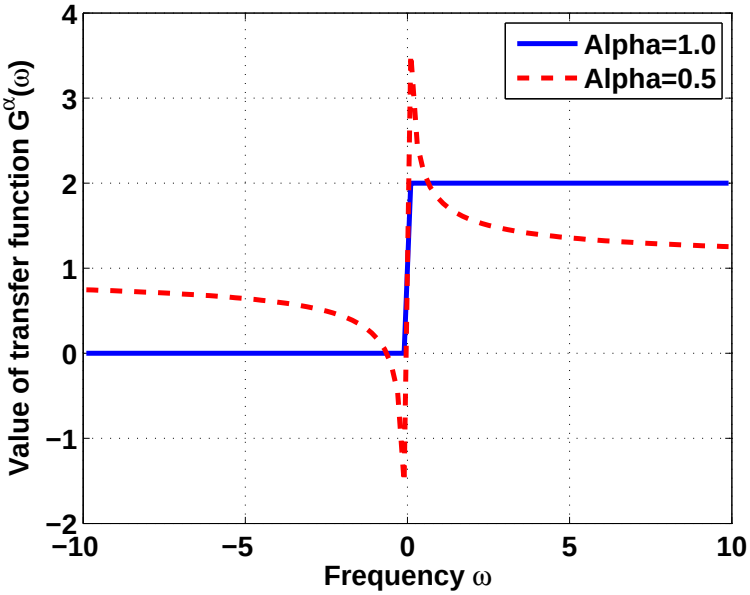


Figure 2.6. Transfer function $G^{\alpha}(\omega)$ of the generalized analytic signal

- *The phase of* $G(\omega)$ *(i.e.,* $G^{\alpha}(\omega)$ *for* $\alpha = 1$*) is* 0 *radians in the frequency range* $(0, \infty)$*. Its magnitude in the negative frequency range* $(-\infty, 0)$ *is identically equal to* 0*; therefore, the phase in this range is inconsequential.*

- *For* $0 < \alpha < 1$*, the phase of* $G^{\alpha}(\omega)$ *is* $-\pi$ *radians in the frequency range* $(-\omega_0(\alpha), 0)$*, where* ω_0 *is defined in Eq. (2.34), and is* 0 *radians in the frequency range* $\big(-\infty, -\omega_0(\alpha)\big) \bigcup (0, \infty)$*.*

9 Test Results and Validation

The concept of generalized Hilbert transform is tested and validated by symbolic analysis of time series data, generated from the same apparatus of nonlinear electronic systems reported in the earlier publication [41]. The symbol sequence, constructed from time series data, is passed through a fixed structure D-Markov machine [17] to compute the state-transition matrices, called Π-matrices, for two values of the depth parameter, $D = 1$ and $D = 2$. Performance of the two D-Markov representations for each partition, corresponding to different values of the parameter α, is compared in terms of the mutual information [46] and the associated

information gain. The procedure is delineated below.

9.1 Collection of Time Series Data

The nonlinear active electronic system in the test apparatus emulates the forced Duffing equation:

$$\frac{d^2x}{dt^2} + \beta\frac{dx}{dt} + x(t) + x^3(t) = Acos(\omega t) \tag{2.35}$$

Having the system parameters set to $\beta = 0.24$, $A = 22.0$, and $\omega = 5.0$, time series data of the variable $x(t)$ were collected from the electronic system apparatus. These data sets do not contain any substantial noise because the laboratory apparatus is carefully designed to shield spurious signals and noise. Therefore, to emulate the effects of noise in the time series data, additive first-order colored Gaussian noise was injected to the collected time series data to investigate the effects of signal-to-noise ratio (SNR). The profile of a typical signal, contaminated with 10 *db* additive Gaussian noise (i.e., $SNR = 10$), is shown in Fig. 2.7.

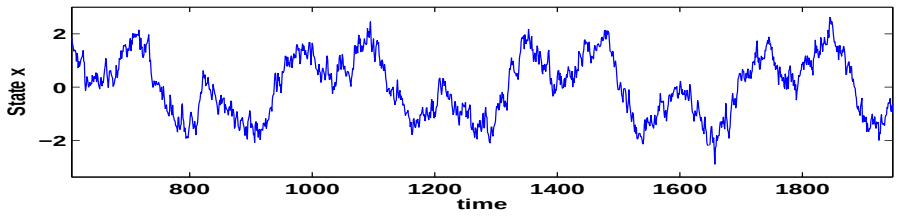


Figure 2.7. Signal contaminated with 10 *db* additive colored Gaussian noise

9.2 Construction of the Transformed Phase Space

Let the real-valued noisy time-series data $x(t)$ contain N data points. Upon generalized Hilbert transformation of this data sequence, a complex-valued generalized analytic signal $\mathcal{X}^{\alpha}(t)$ is constructed. Similar to the procedure described in [41], $\mathcal{X}^{\alpha}(t)$ is represented as a one-dimensional trajectory in the two-dimensional pseudo-phase space. Let Ω be a compact region in the pseudo-phase space, which encloses the trajectory of N such data points.

9.3 Partitioning and Symbol Generation

The next task is to partition Ω into finitely many mutually exclusive and exhaustive segments, where each segment is labelled with a symbol or letter. The partitioning is based on the magnitude and phase of the complex-valued signal $\mathcal{X}^{\alpha}(t)$ as well as the density of data points in these segments, following the procedure described in [41]. Each point in the partitioned data set is represented by a pair of symbols; one belonging to the alphabet Σ_R based on the magnitude (i.e., in the radial direction) and the other belonging to the alphabet Σ_A based on the phase (i.e., in the angular direction). In this way, the complex-valued signal $\mathcal{X}^{\alpha}(t)$ is partitioned into a symbol sequence by associating each pair of symbols to a single symbol belonging to an alphabet Σ that is defined as:

$$\Sigma \triangleq \left\{(\sigma_i, \sigma_j) \in \Sigma_R \times \Sigma_A\right\} \text{ and } |\Sigma| = |\Sigma_R||\Sigma_A| \tag{2.36}$$

The results presented in this chapter are generated with $\Sigma_R = 8$ in the radial direction and $\Sigma_A = 5$ in the angular direction, i.e., $|\Sigma| = 40$.

9.4 State Transition Matrices

The symbol sequence is now used to construct D-Markov machine models [17]. The assumption of statistical stationarity of the symbol sequence is implicit in the construction of Markov models. In this study, Markov chain models of depth $D = 1$ and $D = 2$ have been constructed.

Modeling of the symbolic process as a $(D = 1)$ Markov chain involves evaluation of the Π^1 matrix, where the ij^{th} matrix element π^1_{ij} is defined as the probability that $(n+1)^{th}$ state is i given that the n^{th} state was j, i.e.,

$$\pi^1_{ij} \triangleq P\left(q^1_{n+1} = i \mid q^1_n = j\right) \tag{2.37}$$

where q_k is the state at discrete time instant k. Evidently, the size of the Π matrix is $|\Sigma| \times |\Sigma|$, where $|\Sigma|$ is the number of symbols in the alphabet Σ.

Modeling the symbolic process as a $(D = 2)$ Markov chain involves evaluation

of a 3-dimensional matrix, where the ijk^{th} matrix element π^2_{ijk} is defined as:

$$\pi^2_{ijk} \triangleq P\left(q^2_{n+2} = i | q^2_{n+1} = j, q^2_n = k\right) \tag{2.38}$$

and size of the (sparse) Π^2 matrix is $|\Sigma| \times |\Sigma| \times |\Sigma|$.

Remark 9.1. *Elements of both* Π^1 *and* Π^2 *matrices are estimated by conditional frequency count and their convergence requires a symbol sequence of sufficient length. This aspect has been discussed in [55][47] and is referred to as the stopping rule that assigns a bound on the length of the symbol sequence for parameter identification of the stochastic matrices* Π^1 *and* Π^2*.*

9.5 Computation of Mutual Information

Effectiveness of generalized Hilbert transform for Markov model construction has been examined from an information theoretic perspective [46]. The rationale is that, in a noise-corrupted system, higher values of mutual information imply less uncertainties in the symbol sequence. The mutual information $\mathcal{I}$ is expressed in terms of entropy $\mathcal{S}$ for both $(D = 1)$ and $(D = 2)$ Markov chains in the following set of equations:

$$\mathcal{I}\left(q_{n+3}; q_{n+2}\right) \triangleq \mathcal{S}\left(q_{n+3}\right) - \mathcal{S}\left(q_{n+3} | q_{n+2}\right) \tag{2.39}$$

$$\mathcal{S}\left(q_{n+3}\right) \triangleq -\sum_{\ell=1}^{|\Sigma|} P\left(q_{n+3} = \ell\right) \quad \log_2 P\left(q_{n+3} = \ell\right) \tag{2.40}$$

Usage of maximum entropy partitioning [40] for symbol generation yields: $\mathcal{S}\left(q_{n+3}\right) = \log_2\left(|\Sigma|\right)$.

$$\mathcal{S}\left(q_{n+3} | q_{n+2}\right) \triangleq \sum_{\ell=1}^{|\Sigma|} P\left(q_{n+2} = \ell\right) \ \mathcal{S}\left(q_{n+3} | q_{n+2} = \ell\right) \tag{2.41}$$

where

$$\mathcal{S}\left(q_{n+3}|q_{n+2}=\ell\right)=-\sum_{j=1}^{|\Sigma|}P\left(q_{n+3}=j\mid q_{n+2}=\ell\right)\cdot \log_2 P\left(q_{n+3}=j|q_{n+2}=\ell\right) \tag{2.42}$$

$$\mathcal{I}\left(q_{n+3};q_{n+2},q_{n+1}\right)\triangleq\mathcal{S}\left(q_{n+3}\right)-\mathcal{S}\left(q_{n+3}|q_{n+2},q_{n+1}\right) \tag{2.43}$$

$$\begin{aligned}&\mathcal{S}\left(q_{n+3}|q_{n+2},q_{n+1}\right)\\&\quad\triangleq-\sum_{i=1}^{|\Sigma|}\sum_{j=1}^{|\Sigma|}P\left(q_{n+2}=i,q_{n+2}=j\right)\cdot\\&\quad\mathcal{S}\left(q_{n+3}|q_{n+2}=i,q_{n+1}=j\right)\end{aligned} \tag{2.44}$$

where

$$\begin{aligned}&\mathcal{S}\left(q_{n+3}|q_{n+2}=i,q_{n+1}=j\right)\\&\quad=-\sum_{\ell=1}^{|\Sigma|}P\left(q_{n+3}=\ell|q_{n+2}=i,q_{n+1}=j\right)\cdot\\&\quad\log_2 P\left(q_{n+3}=\ell|q_{n+2}=i,q_{n+j}=j\right)\end{aligned} \tag{2.45}$$

Based on Eq. (2.37) and Eqs. (2.39) to (2.42), the mutual information $\mathcal{I}\left(q_{n+3};q_{n+2}\right)$ is calculated from the Π^1 matrix. Similarly, based on Eq. (2.38) and Eqs. (2.43) to (2.45), $\mathcal{I}\left(q_{n+3};q_{n+2},q_{n+1}\right)$ is calculated from the Π^2 matrix. Then, information gain (abbreviated as $\mathcal{I}_{\mathcal{G}}$) with $D=2$ instead of $D=1$ in the Markov chain construction is defined as:

$$\mathcal{I}_{\mathcal{G}}\triangleq\mathcal{I}\left(q_{n+3};q_{n+2},q_{n+1}\right)-\mathcal{I}\left(q_{n+3};q_{n+2}\right) \tag{2.46}$$

9.6 Pertinent Results

This subsection presents test and validation of the concept of generalized Hilbert transform based on the time series data collected from the laboratory apparatus of nonlinear electronic system. The test results are interpreted in terms of mutual

information for ($D = 1$) and ($D = 2$) Markov chains for noise-contaminated data for different values of SNR and the parameter α (see Eq. (2.16)). The pertinent results on mutual information and information gain are presented in Fig. 2.8 and Fig. 2.9, respectively. Although results are shown only for $SNR = \infty, 10, 4$ and 0, several other experiments with intermediate values of SNR between ∞ and 0 were performed, which shows the same trend.

The information gain is always a positive quantity as seen in Eq. (2.46). In other words, there is always a gain in information upon increasing the depth of the Markov chain model. Pertinent inferences, drawn from these results, are presented below.

1. Mutual information increases with decrease in α irrespective of D and SNR as seen in Fig. 2.8.

2. Information gain $\mathcal{I}_{\mathcal{G}}$ (see Eq. (2.46) and Fig. 2.9) is minimal for $SNR \to \infty$ (i.e., for the signal with no noise injection). Therefore, D=1 Markov chain should be adequate with $ASSP$ using conventional Hilbert transform (see Eq. 2.12) for low-noise signals.

3. As SNR is decreased (i.e., percentage of additive noise is increased), information gain $\mathcal{I}_{\mathcal{G}}$ increases for all values of α in the range of 1.0 down to about 0.2. As α is decreased, information gain decreases as seen in Fig. 2.9. Therefore, even for a considerable amount of noise, a smaller value of α should be able to achieve noise attenuation and thus allow usage of $D = 1$ in D-Markov machines.

4. Results for a pathological case with $SNR \to 0$, (i.e., complete noise capture of the signal) in Fig. 2.8 and Fig. 2.9 show similar trends as above. The crossing of the information gain curves in Fig. 2.9 at low values of α (e.g., $\alpha \leq 0.2$) could possibly be attributed to the effects of coarse graining [49] due to symbol generation.

The rationale for the observed trends in Fig. 2.8 and Fig. 2.9 is reiterated as follows. An increase in depth D captures the effects of longer memory in the signal, and similarly reduced α puts more weight on the low-frequency components, which assists noise reduction.

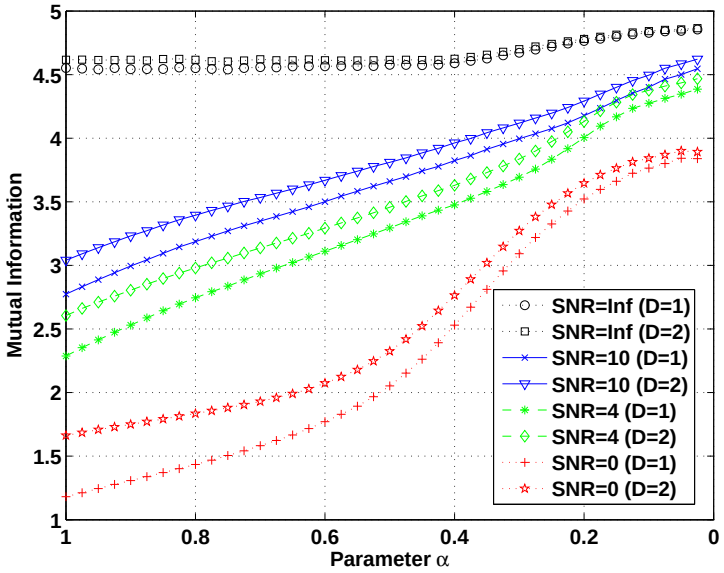


Figure 2.8. Profiles of mutual information

10 Summary, Conclusion and Future work

This study formulates a generalization of the classical Hilbert transform along with proofs of the pertinent lemmas that are required for the theoretical formulation. The proposed scheme of generalized Hilbert transform is shown to be potentially useful for symbolic time series analysis of noise-corrupted dynamical systems. The proposed concept of noise reduction via generalization of Hilbert transform is tested and validated on experimental data collected from a laboratory test apparatus.

The following conclusions are drawn from the validation results of the proposed generalization of Hilbert transform on a laboratory apparatus of nonlinear electronic systems as presented in the previous section.

- Generalized Hilbert transform with a smaller value of parameter α is capable of extracting more information from a data sequence irrespective of the depth of the D-Markov machine chosen for modeling.

- Information gain for a larger depth D reduces with smaller values of the parameter α.

- By selecting small values of the parameter α in the generalized Hilbert trans-

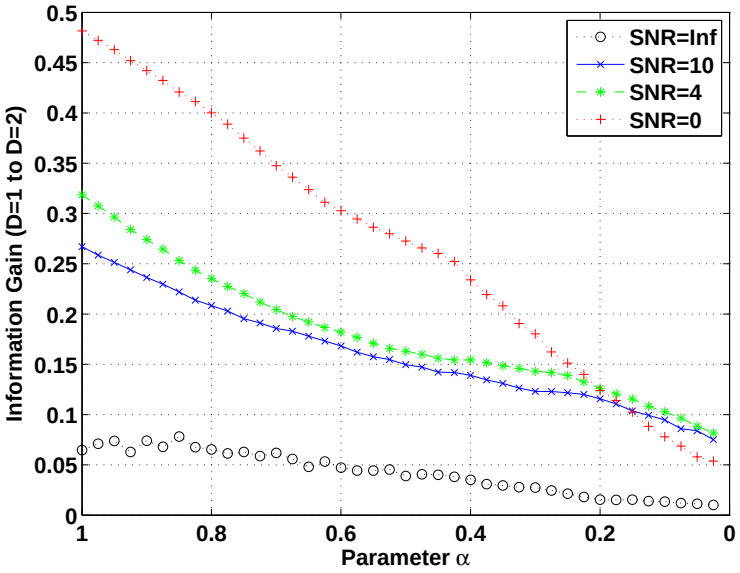


Figure 2.9. Profiles of information gain

form, it is possible to avoid using a computationally expensive larger depth D without loss of significant information.

The proposed method of noise attenuation via generalization of Hilbert transform is potentially useful for symbolic time series analysis of noise-corrupted dynamical systems. The future work should be directed toward advancement of the theory of partitioning as well as on application to different real-life uncertain systems. Examples include sensor networks that require on-board real-time analysis of noisy signals with very low computation capacity.

Chapter 3

Optimal Feature Extraction for Classification via Partitioning

This chapter deals with the Partitioning step of SDF in detail. It is evident from the discussions earlier that this first level of abstraction is extremely crucial for extracting features under the SDF framework. Although modeling of state machines from symbol sequences has been widely reported, similar efforts have not been expended to investigate partitioning of time series to optimally generate the symbol sequences. This chapter addresses this issue and proposes a data partitioning procedure to extract low-dimensional features from time series while optimizing the class separability. Three validation examples are presented here: (i) parameter identification in a Duffing system [70], (ii) data-driven fault detection in aircraft gas turbine engines [13], and (iii) classification of fatigue damage in mechanical structures, made of polycrystalline alloys [71]. In each case, the classification performance of the proposed data partitioning method is compared with those of two other classical data partitioning methods, namely uniform partitioning (UP) and maximum entropy partitioning (MEP).

1 Motivation

While the properties and variations of transformation from a symbol space to a pattern space have been thoroughly studied in the disciplines of mathematics, computer science and especially data mining, similar efforts have not been expended to

investigate partitioning of time series data to optimally generate symbol sequences for pattern classification. Ideally, the optimal partitioning for a dynamical system evolution is called the Generating Partition, which is defined as: A fixed and finite partitioning for a given map is called a generating partition, if an infinite symbol sequence uniquely determines the initial condition of the evolution [49]. In other words, the symbolic dynamics mapping is bijective, i.e., not only the initial condition determines the symbol sequence, but the reverse is true as well. In general, whether there exists a generating partition for a given map or not is a very tough question to answer. Apparently, it is possible to construct a tractable generating partition for a few simple maps. Symbolic false nearest neighbor partitioning (SFNNP) [63] optimizes a generating partition by avoiding topological degeneracy. However, a shortcoming of SFNNP is that it may become extremely computation intensive if the dimension of the phase space of the underlying dynamical system is large. Furthermore, if the time series data become noise-corrupted, the symbolic false neighbors rapidly grow in number and may erroneously require a large number of symbols to capture pertinent information on the system dynamics. Another possibly useful partitioning scheme is called the Markov partition that converts the (continuous) system evolution to a topological Markov chain. In realistic applications, the concepts of maximum entropy and uniform partitioning schemes are very popular. The concept of entropy is widely utilized in evaluating performance of partitioning techniques. In [72], Steuer *et al.* showed that homogeneous partition (*i.e.*, uniform partitioning) which ensures roughly equidistributed symbols may lead to spurious results for the estimated entropy and will not fully reveal the randomness of the sequence. They therefore suggested to maximize the entropy given a certain length of the alphabet(*i.e.*, maximum entropy partitioning). It was shown that maximum entropy partitioning gave a better tool to differentiate sequences than uniform partitioning. Li *et al.* [73] reported a similar algorithm using the base-scale entropy to detect dynamical complexity changes in time series analysis. This method is fast and robust, and able to effectively detect qualitative and quantitative dynamical changes. However, the fixed alphabet size may restrict its adaptation to changes in data sets, and more importantly, the choice of partition positions relies on iterative tests and may not be optimal. Chau [74] reported that the marginal maximum entropy criterion with recursive partitioning provides

asymptotically consistent probability density function estimates. The partitioning is called marginal due to the fact that partitioning is not performed in the full dimensionality, but rather marginally along each dimension, and thus is suboptimal. The previous chapter discussed the effects of data pre-processing via Wavelet or Hilbert transform for partitioning. Nevertheless, these partitioning techniques primarily attempt to provide an accurate symbolic representation of the underlying dynamical system under a given quasi-stationary condition, rather than trying to capture the data-evolution characteristics. In this study, the difficulties of the above-mentioned partitioning methods have been overcome with the objective of making the symbolic feature extraction, a robust and optimal time-series feature extraction tool for enhancement of pattern classification.

The key idea of the work reported here is to take advantage of non-stationary dynamics (in a slower scale) and optimize the partitioning process based on the statistical changes in the time series over a given set of training data belonging to different classes. Major contributions of the study are delineated below.

1. *Partitioning of time series for optimization of pattern classification; The resulting framework is general enough to incorporate additional objectives in the cost functional.*

2. *Construction of a cost function to incorporate trade-offs among sensitivity to changes in data characteristics, robustness to spurious disturbances, and quantization error by using fuzzy partitioning cell boundaries*

3. *Validation of the proposed concepts in different application areas for the purpose of parameter identification and fault detection*

The chapter is organized into seven sections including the present one. Section 2 presents the mathematical framework partitioning in the context of feature extraction and classification. The partitioning optimization scheme is elaborated in Section 3 along with its key features. Section 4 validates the proposed concepts on a simulation test bed of a second order non-autonomous forced Duffing equation [75]; the second validation example of data-driven fault detection in aircraft gas turbine engines under varying sensor noise characteristics is presented in Section 5. Finally, Section 6 presents the third validation example dealing with

fatigue damage classification. Section 8 summarizes the chapter and makes major conclusions along with recommendations for future research.

2 Partitioning: A Nonlinear Feature Extraction Technique

As discussed earlier, symbolic feature extraction from time series data is posed as a two-scale problem: *fast scale* where dynamic behavior of the system is assumed to be statistically stationary and *slow scale* where non-stationary evolution of the system dynamics may occur. A mathematical description of the partitioning process is given below.

Sensor time series, denoted as $\mathbf{q}$, is generated at a slow-scale epoch from a physical system or its dynamical model. A compact (i.e., closed and bounded) region $\Omega \in \mathbb{R}^n$, where $n \in \mathbb{N}$, within which the (quasi-stationary) time series is circumscribed, is identified. Let the space of time series data sets be represented as $\mathcal{Q} \subseteq \mathbb{R}^{n \times N}$, where the $N \in \mathbb{N}$ is sufficiently large for convergence of statistical properties within a specified threshold. (Note: n represents the dimensionality of the time-series and N is the number of data points in the time series. Encoding of Ω is accomplished by introducing a partition $\mathbb{B} \equiv \{B_0, ..., B_{(m-1)}\}$ consisting of m mutually exclusive (i.e., $B_j \cap B_k = \emptyset \ \forall j \neq k$), and exhaustive (i.e., $\cup_{j=0}^{m-1} B_j = \Omega$) cells. Let each cell be labeled by symbols $s_j \in \Sigma$, where $\Sigma = \{s_0, ..., s_{m-1}\}$ is called the alphabet. This process of coarse graining can be executed by uniform, maximum entropy, or any other scheme of partitioning. Then, the time series data points in $\{\mathbf{q}\}$, which visit the cell B_j are denoted as $s_j \ \forall j = 0, 1, ..., m-1$. This step enables transformation of the time series data $\{\mathbf{q}\}$ to a symbol sequence $\{\mathbf{s}\}$. In the SDF framework, the symbol sequence $\{\mathbf{s}\}$ is compressed by a probabilistic finite state automaton ($PFSA$), where $j, k \in \{1, 2, ..., r\}$ are the states of the $PFSA$ with the $(r \times r)$ state transition matrix $\Pi = [\pi_{jk}]$ that is obtained at slow-scale epochs. (Note: Π is a stochastic matrix, i.e., the transition probability $\pi_{jk} \geq 0$ and $\sum_k \pi_{jk} = 1$). To compress the information further, the state probability vector $\mathbf{p} = [p_1 \ \cdots \ p_r]$ that is the left eigenvector corresponding to the (unique) unity eigenvalue of the irreducible stochastic matrix Π is calculated. The vector $\mathbf{p}$ is the extracted feature vector and is a low-dimensional compression of the long

time series data representing the dynamical system at the slow-scale epoch. The following subsection describes the classification framework using low dimensional features extracted from time series data by classical partitioning schemes.

2.1 Classification Using Low-dimensional Feature Vectors

For classification using SDF, the reference time series, belonging to a class denoted as Cl_1, is symbolized by one of the standard partitioning schemes (e.g., Uniform Partitioning (UP) or Maximum Entropy partitioning (MEP)) [17, 40, 41]. Then, using the steps described earlier, a low-dimensional feature vector $\mathbf{p}^{Cl_1}$ is constructed for the reference slow-scale epoch. Similarly, from a time series belonging to a different class denoted as Cl_2, a feature vector $\mathbf{p}^{Cl_2}$ is constructed using the same partitioning as in Cl_1. The next step is to classify the data in the constructed low-dimensional feature space. In this respect, there are many options for selecting classifiers that could either be parametric or non-parametric. Among the parametric classifiers, one of the commonly used techniques relies on the second-order statistics in the feature space, where the mean feature is calculated for every class along with the variance of the feature space distribution in each class of the training set. Then, a test feature vector is classified by using the Mahalnobis distance [76] or the Bhattacharya distance [77] of the test vector from the mean feature vector of each class. However, these methods are not efficient for non-Gaussian distributions, where the feature space distributions may not be adequately described by the second order statistics. Consequently, a non-parametric classifier (e.g., k-NN classifier [78]) is potentially a better choice. In this study, the k-NN classifier has been used because the Gaussian probability distribution cannot be assured in the feature space. However, in general, any other suitable classifier, such as the Support Vector Machines (SVM) or the Gaussian Mixture Models (GMM) may also be used [78]. To classify the test data set, the time series sets are converted into feature vectors using the same partitioning that has been used to generate the training features. Then, using the labeled training features, the test features are classified by a k-NN classifier with suitable specifications (neighborhood size and distance metric).

3 Optimization of Partitioning

Many optimization criteria have been reported in literature for feature extraction in multi-class classification problems. However, none can be more fundamental than minimization of classification error. Although, there have been attempts to approximate the classification error as the Bayes error using pairwise weighting functions for multi-class problems [79], the Bayes error itself cannot be expressed analytically except a few special cases. On the other hand, even if the Fisher criteria is not directly equivalent to classification error, it is widely used as a feature extraction optimization criteria in literature. This criteria involves maximization of the inter-class variance and minimization of the intra-class variance for the training data set. For a K-class ($K > 2$) classification problem, a general K-class Fisher criterion may be used; alternatively, a sum of $K(K-1)/2$ number of 2-class Fisher criteria may be used as well. This criterion is very useful for binary classification problems, especially when the samples are distributed in a Gaussian manner in the feature space. For multi-class problems [80], a criterion has been proposed based on the minimum classification error (MCE) that is calculated by the misclassification rate over the training samples. Formally, these two (fundamentally different) optimization criteria are known as the (i) Filter method and (ii) Wrapper method. While a filter method uses information content feedback (e.g., Fisher criterion, and statistical dependence) as optimization criteria for feature extraction, a wrapper method includes the classifier inside the optimization loop and attempts to maximize the predictive accuracy (e.g., classification rate) using statistical re-sampling or cross-validation [78]. In this methodology, the wrapper method is adopted (i.e., via minimization of the classification error on the training set) for optimization, primarily because of the non-binary nature of the problem at hand and the possible non-Gaussian distribution of training samples in the feature space.

In this context, ideally one should jointly minimize every element of the confusion matrix [81]. However, in that case, the dimension of the objective space may rapidly increase with an increase in the number of classes. To circumvent this situation in the present work, two costs are defined on the confusion matrix by using another weighting matrix, elements of which denote the relative penalty values for different confusions in the classification process. Formally, let there be

$Cl_1, \cdots, Cl_n$ classes of labeled time-series data in the training set. A partitioning $\mathbb{B}$ is employed to extract features from each sample and a k-NN classifier $\mathbb{K}$ is used as a classifier. The confusion matrix $\mathbf{C}$ is obtained upon completion of the classification process, where the ij^{th} element c_{ij} of $\mathbf{C}$ denotes the frequency of data from class Cl_i being classified as data from Cl_j. Let $\mathbf{W}$ be the weighting matrix, where the ij^{th} element w_{ij} of $\mathbf{W}$ denotes the penalty incurred for classifying data from Cl_i as data from class Cl_j. (Note: Since there is no penalty for correct classification, the diagonal elements of $\mathbf{W}$ are identically equal to 0, i.e., $w_{ii} = 0 \; \forall \; i$.) With these specifications, two costs, $CostE$ and $CostW$, that are to be minimized are defined as follows.

The cost $CostE$ due to expected classification error is defined as:

$$CostE = \frac{1}{N_s} \left(\sum_i \sum_j w_{ij} c_{ij} \right) \tag{3.1}$$

where N_s is the total number of training samples including all classes. The above equation represents the total penalty for misclassification across all classes. Thus $CostE$ is related to the expected classification error. The weights w_{ij} are selected based on the prior knowledge about the data and the user's requirements.

It is implicitly assumed in many supervised learning algorithms that the training data set is a statistically similar representation of the whole data set. However, this assumption may not be accurate in practice. A solution to this problem is to choose a feature extractor that minimizes the worst-case classification error [82]. In this setting, that cost $CostW$ due to worst-case classification error is defined as:

$$CostW = \max_i \left(\frac{1}{N_i} \sum_j w_{ij} c_{ij} \right) \tag{3.2}$$

where N_i is the number of training samples in the class Cl_i. (Note: In the present formulation, the objective space is two-dimensional for a multi-class classification problem and the dimension is not a function of the number of classes.)

Sensitivity and Robustness

Cost minimization requires a choice of partitioning that could be sensitive to class separation in the data space. However, the enhanced sensitivity of optimal parti-

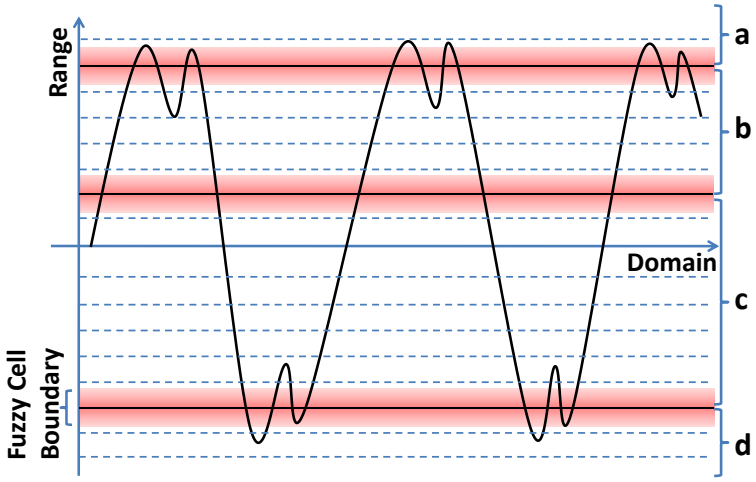


Figure 3.1. Fuzzy Cell Boundaries to obtain $CostE_{robust}$ and $CostW_{robust}$

tioning may cause large classification errors due to noise and spurious disturbances in the data and statistical variations among training and testing data sets. Hence, it is important to invoke robustness properties into the optimal partitioning. In this study, robustness has been incorporated by modifying the costs $CostE$ and $CostW$, introduced in the previous subsection.

For one-dimensional time series data, a partitioning consisting of $|\Sigma|$ cells is represented by $(|\Sigma| - 1)$ points that serve as cell boundaries. In the sequel, a Σ-cell partitioning $\mathbb{B}$ is expressed as $\Lambda_{|\Sigma|} \triangleq \{\lambda_1, \lambda_2, \cdots, \lambda_{|\Sigma|-1}\}$, where λ_i denotes a partitioning boundary. Thus, a $Cost$ is dependent on the specific partitioning $\Lambda_{|\Sigma|}$ and is denoted by $Cost(\Lambda_{|\Sigma|})$. The key idea of a robust cost for a partitioning $\Lambda_{|\Sigma|}$ is that it is expected to remain invariant under small perturbations of the partitioning points, $\lambda_1, \lambda_2, \cdots, \lambda_{|\Sigma|-1}$ (i.e., fuzzy cell boundaries). To define a robust cost, a distribution of partitioning $f_{\Lambda_{|\Sigma|}}(\cdot)$ is considered, where a sample of the distribution is denoted as $\tilde{\Lambda}_{|\Sigma|} = \{\tilde{\lambda}_1, \tilde{\lambda}_2, \cdots, \tilde{\lambda}_{|\Sigma|-1}\}$ and $\tilde{\lambda}_i$'s are chosen from independent Gaussian distributions with mean λ_i and a uniform standard deviation σ_λ; the choice of σ_λ is discussed later in Subsection 3.1.

The resulting cost distribution is denoted as $f_{Cost(\Lambda_{|\Sigma|})}(\cdot)$, and $Cost_{robust}$ is defined as the cost value below which 95% of the population of distribution $f_{Cost(\Lambda_{|\Sigma|})}(\cdot)$ remains and is denoted as:

$$Cost_{robust} = P_{95}[f_{Cost(\Lambda_{|\Sigma|})}(\cdot)] \tag{3.3}$$

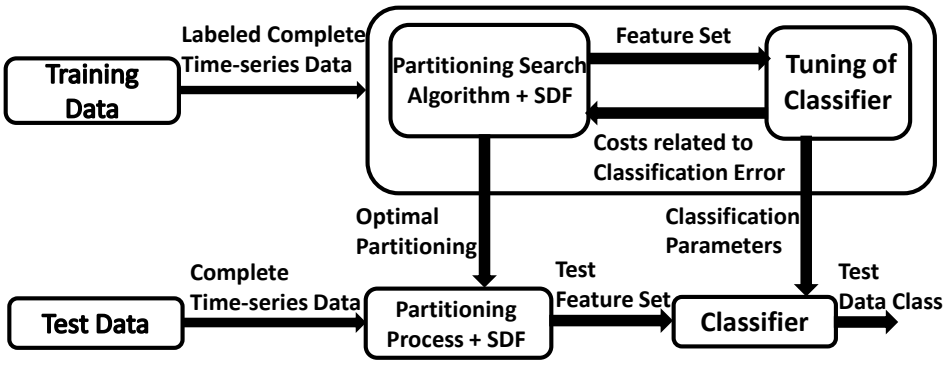


Figure 3.2. General Framework for Optimization of Feature Extraction

For the analysis to be presented in Subsection 3.1, it is assumed that sufficient samples are generated to obtain a good estimate of $Cost_{robust}$ as explained in Fig. 3.1, where the firm lines denote the boundaries that divide the range space into four cells, namely a, b, c and d. However, in general, the cell boundaries are fuzzy in nature, which leads to a probabilistic $Cost_{robust}$ instead of a deterministic cost based on the firm partitioning boundaries. Thus, a robust cost due to expected classification error, $CostE_{robust}$, and a robust cost due to worst-case classification error, $CostW_{robust}$, are defined for a given partitioning $\Lambda_{|\Sigma|}$.

Finally, a multi-objective overall cost $CostO$ is defined as a linear convex combination of $CostE_{robust}$ and $CostW_{robust}$ in terms a scalar parameter $\alpha \in [0, 1]$.

$$CostO \triangleq \alpha CostE_{robust} + (1 - \alpha) CostW_{robust} \tag{3.4}$$

Ideally, the optimization procedure involves construction of the Pareto front [6] by minimizing $CostO$ for different values of α that can be freely chosen as the operating point. However, for simplicity α is taken to be 0.5 in this work, i.e., equal weight for the costs $CostE_{robust}$ and $CostW_{robust}$. Thus, the optimal Σ-cell partitioning $\mathbb{B}^*$ is the solution to the following optimization problem:

$$\mathbb{B}^* = \arg\min_{\mathbb{B}} CostO(\mathbb{B}) \tag{3.5}$$

Figure 3.2 depicts a general outline of the classification process. Labeled time series data from the training set are partitioned. The low-dimensional feature vectors that are generated by symbolization and $PFSA$ construction are fed to the

classifier. After classification, the training error costs, defined above, are computed and fed back to the feature extraction block. In the classification aspect, the classifier may be tuned to the obtain better classification rates. For example, for k-NN classifiers [78], the choice of neighborhood size or the distance metric can be tuned. Similarly, for support vector machines [78], an appropriate hyperplane should be selected to achieve good classification. The key idea is to update the partitioning to reduce the cost based on the feedback. The iteration is continued until the set of optimal partitioning in a multi-objective scenario and the correspondingly tuned classifier are obtained. Generally, choice of the optimal partitioning is made based on the choice of operating point α by the user. After the choice is made, the optimal partitioning and the tuned classifier are used to classify the test data set. Although this is the general framework that is being proposed for the optimization methodology, tuning of the classifier is not addressed in this chapter, because the main focus here is to choose the optimal partitioning to minimize the cost related to classification errors.

3.1 Optimization Procedure

A sequential search-based technique has been adopted in this study for optimization of the partitioning, which was previously proposed in [83]. As the continuity of the partitioning function with respect to the range space of classification error-related costs may not exist or at least are not adequately analyzed, gradient-based optimization methods are not explored here. To construct the search space, a suitably fine grid size depending on the data characteristics needs to be assumed. Each of the grid boundaries denotes a possible position of a partitioning cell boundary, as illustrated in Fig. 3.1. Here, the dotted lines denote the possible positions of a partitioning cell boundary and as discussed before, for a chosen partitioning (denoted by firm lines), the partitioning boundaries are perturbed to obtain a $Cost_{robust}$.

Let the data space region Ω be divided into G grid cells for search, i.e., there are $(G-1)$ grid boundaries excluding the boundaries. Thus, there are $|\Sigma|-1$ partitioning boundaries to choose among $(G-1)$ possibilities, i.e., the number of elements (i.e., $(|\Sigma|-1)$-dimensional partitioning vectors) in the space $\mathcal{P}$ of all possible partitioning is: ${}^{(G-1)}C_{(|\Sigma|-1)}$. It is clear from this analysis that the

partitioning space $\mathcal{P}$ may become significantly large with an increase in values of G and $|\Sigma|$ (e.g., for $G >> |\Sigma|$, computational complexity increases approximately by a factor of $G/|\Sigma|$ with increase in the value of $|\Sigma|$ by one). Furthermore, for each element of $\mathcal{P}$, a sufficiently large number of perturbed samples need to be collected in order to obtain the $Cost_{robust}$. Therefore, usage of a direct search approach becomes infeasible for evaluation of all possible partitioning. Hence, a sub-optimal solution is developed in this chapter to reduce the computational complexity of the optimization problem.

The objective space consists of the scalar-valued cost $CostO$, while decisions are made in the space $\mathcal{P}$ of all possible partitionings. The overall cost is dependent on a specific partitioning Λ and is denoted by $CostO(\Lambda)$. This sub-optimal partitioning scheme involves sequential estimation of the elements of the partitioning Λ.

The partitioning process is initiated by searching the optimal cell boundary to divide the data set into two cells, i.e., $\Lambda_2 = \{\lambda_1\}$, where λ_1 is evaluated as

$$\lambda_1^* = \arg\min_{\lambda_1} CostO(\Lambda_2) \tag{3.6}$$

Now, the two-cell optimal partitioning is given by $\Lambda_2^* = \{\lambda_1^*\}$.

Note that to obtain $CostO$ (i.e., both $CostE_{robust}$ and $CostW_{robust}$) for a given partitioning, a suitable σ_λ needs to be chosen. Let the gap between two search grid boundaries (i.e., two consecutive dotted lines in Fig. 3.1) be l_λ, and σ_λ is chosen as $l_\lambda/3$ in this study. The choice of such a standard deviation of the Gaussian perturbation is made for approximately complete coverage of the search space. Note that there could be an overlap of perturbation regions between two consecutive search grid boundaries, which leads to a smoother (that is essentially robust) cost variation across the domain space. This issue will be further discussed in Subsection 7.2. In addition, depending on the gap l_λ and data characteristics, a suitable sample size is chosen to approximate the cost distribution under the fuzzy cell boundary condition.

The next step is to partition the data into three cells as Λ_3) by dividing either of the two existing cells of Λ_2^* with the placement of a new partition boundary at

λ_2, where λ_2 is evaluated as

$$\lambda_2^* = \arg\min_{\lambda_2} CostO(\Lambda_3) \tag{3.7}$$

where $\Lambda_3 = \{\lambda_1^*, \lambda_2\}$. The optimal 3-cell partitioning is obtained as $\Lambda_3^* = \{\lambda_1^*, \lambda_2^*\}$. In this (local) optimization procedure, the cell that provides the largest decrement in $CostO$ upon further segmentation ends up being partitioned. Iteratively, this procedure is extended to obtain the $|\Sigma|$ cell partitioning as follows.

$$\lambda_{|\Sigma|-1}^* = \arg\min_{\lambda_{|\Sigma|-1}} CostO(\Lambda_{|\Sigma|}) \tag{3.8}$$

where $\Lambda_{|\Sigma|} = \Lambda_{|\Sigma|-1}^* \cup \{\lambda_{|\Sigma|-1}\}$ and the optimal $|\Sigma|$ cell partitioning is given by $\Lambda_{|\Sigma|}^* = \Lambda_{|\Sigma|-1}^* \cup \{\lambda_{|\Sigma|-1}^*\}$

In this optimization procedure, the cost function decreases monotonically with every additional sequential operation, under the assumption of correct estimation of $CostE_{robust}$, $CostW_{robust}$ and hence, $CostO$ under the fuzzy cell boundary condition. Formally, $CostO(\Lambda_{|\Sigma|-1}^*) \geq CostO(\Lambda_{|\Sigma|}^*)$. as explained below.

Let $\Lambda_{|\Sigma|-1}^*$ be the ($|\Sigma|$-1)-cell partitioning that minimizes $CostO$, based on the algorithm, $\Lambda_{|\Sigma|} = \Lambda_{|\Sigma|-1}^* \cup \{\lambda_{|\Sigma|-1}\}$). If $\lambda_{|\Sigma|-1}$ is chosen such that it already belongs to $\Lambda_{|\Sigma|-1}^*$, then there would be no change in the partitioning structure, i.e.,

$$CostO(\Lambda_{|\Sigma|}) = CostO(\Lambda_{|\Sigma|-1}^*) \quad \text{for} \quad \lambda_{|\Sigma|-1} \in \Lambda_{|\Sigma|-1}^* \tag{3.9}$$

If $\lambda_{|\Sigma|-1} \in \Lambda_{|\Sigma|-1}^*$, then the partitioning $\Lambda_{|\Sigma|}$ is essentially treated as a $(|\Sigma|-1)$-cell partitioning for the purpose of cost calculation. By definition,

$$CostO(\Lambda_{|\Sigma|}^*) \leq CostO(\Lambda_{|\Sigma|}) \quad \forall \Lambda_{|\Sigma|} \tag{3.10}$$

Then, it follows that,

$$\min(CostO(\Lambda_{|\Sigma|-1})) \geq \min(CostO(\Lambda_{|\Sigma|})) \tag{3.11}$$

Or,

$$CostO(\Lambda_{|\Sigma|-1}^*) \geq CostO(\Lambda_{|\Sigma|}^*) \tag{3.12}$$

The monotonicity in the cost function allows formulation of a rule for termination of the sequential optimization algorithm. The process of creating additional partitioning cells is stopped if the cost decrease falls below a specified positive scalar threshold η_{stop} and the stopping rule is as follows.

$\Lambda^*_{|\Sigma|-1}$ is the optimal partitioning (and $|\Sigma|-1$ is the optimal Alphabet size) if

$$CostO(\Lambda^*_{|\Sigma|-1}) - CostO(\Lambda^*_{|\Sigma|}) \leq \eta_{stop}. \tag{3.13}$$

In contrast to the direct search of the entire space of partitioning, the computational complexity of this approach increases linearly with $|\Sigma|$. This approach also allows the user to have finer grid size for the partitioning search.

4 Validation Example 1: Parameter Identification in Nonlinear Dynamical Systems

This section describes the first example and the associated results to validate the merits of the proposed technique. The problem of parameter identification in the nonlinear Duffing system that is posed as a multi-class classification problem in Subsection 4.1 and Subsection 4.2 presents the classification results along with relevant discussions.

4.1 Problem Description

The exogenously excited Duffing system [75] is nonlinear with chaotic properties and its governing equation is:

$$\frac{d^2y(t)}{dt^2} + \beta\frac{dy}{dt} + \alpha_1 y(t) + y^3(t) = A\cos(\Omega t) \tag{3.14}$$

where the amplitude $A = 22.0$, excitation frequency $\Omega = 5.0$, and reference values of the remaining parameters, to be identified, are: $\alpha_1 = 1.0$ and $\beta = 0.1$. It is known that this system goes through a bifurcation at different combinations of α_1 and β, which can be identified by standard feature extraction procedures [54]. The problem at hand is to accurately identify the ranges of the parameters α_1

and β when the system has not undergone any bifurcation. In this example, multiple classes are defined based on the combination of approximate ranges of the parameters α_1 and β as described below.

Parameter	**Values of α_1**
Range 1	0.800 to 0.934
Range 2	0.934 to 1.067
Range 3	1.067 to 1.200

Parameter	**Values of β**
Range 1	0.100 to 0.147
Range 2	0.147 to 0.194
Range 3	0.194 to 0.240

In this study, classes are defined as Cartesian products of the ranges of α_1 and β. Thus, there are 9 (i.e., 3×3) classes of data, where a class is uniquely defined by a range of α_1 and a range of β. Two hundred simulation runs of the Duffing system have been conducted for each class to generate data set for analysis among which 100 samples are chosen as the training set and the remaining 100 samples are kept as testing set. Parameters α_1 and β are chosen randomly from independent Uniform distributions for both parameters within the prescribed ranges given in above table. Figure 3.3 plots the samples generated using the above logic in the two dimensional parameter space. Different classes of samples are shown in different colors and as well as marked with the class numbers in the figure. For each sample point in the parameter space, time series has been collected for *State* y, the length of the simulation time window being 80 seconds sampled at 100 Hz, which generates $8,000$ data points. Figure 3.4 exhibits typical phase plots of the Duffing system from each of the nine classes. The following section presents the classification performance of the optimal partitioning along with a comparison with that of the classical partitioning schemes.

4.2 Results and Discussion

A weighting matrix $\mathbf{W}$ needs to be defined to calculate the classification error related costs as discussed in Section 3. In the present case, $\mathbf{W}$ is defined according to the adjacency properties of classes in the parameter space. That means $w_{ii} = 0,\ \forall i \in \{1, 2, \cdots, 9\}$, i.e. there is no penalty when Cl_i is classified as Cl_i and in general $w_{ij} = |R_{\alpha_1}(i) - R_{\alpha_1}(j)| + |R_{\beta}(i) - R_{\beta}(j)|,\ \forall i \in \{1, 2, \cdots, 9\}$, where $R_{\gamma}(k)$ denotes the range number for parameter γ (in this case, α_1 or β) in class k. In

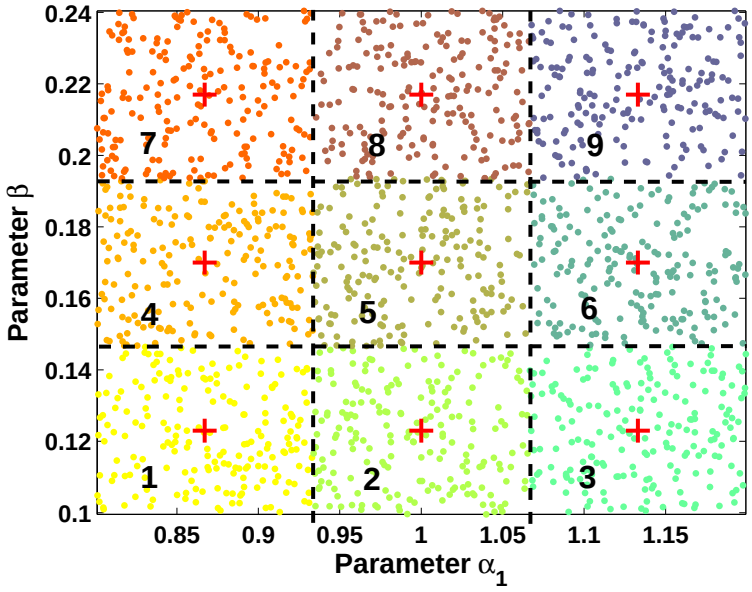


Figure 3.3. Parameter Space with Class Labels

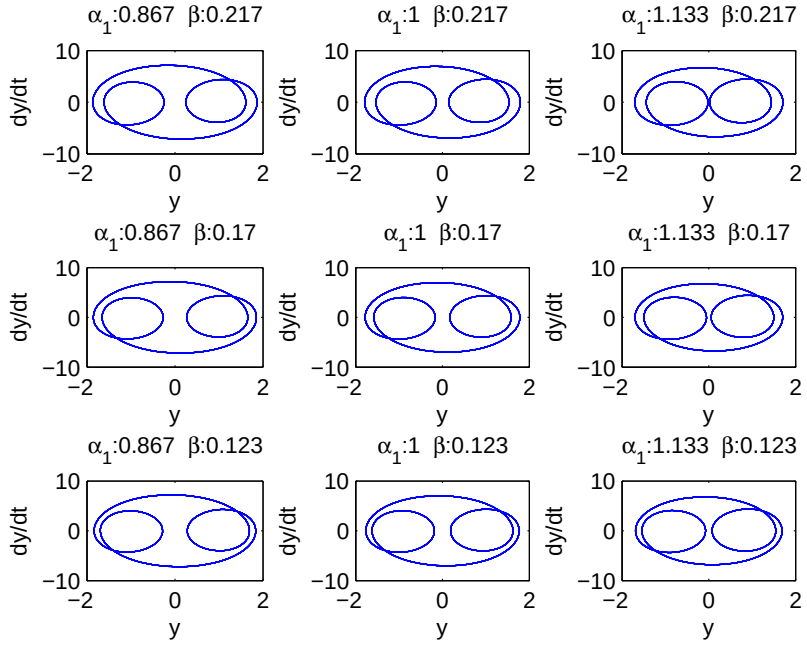


Figure 3.4. Representative Phase Space Plots for Different Classes

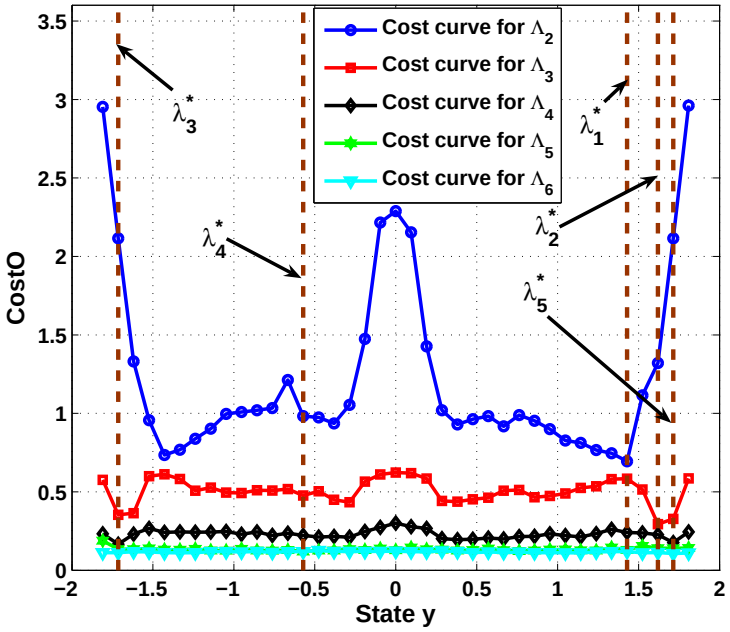


Figure 3.5. Cost curves and Optimal Partitioning boundaries for different Alphabet sizes obtained during the Sequential Optimization procedure

this context, $\mathbf{W}$ is written as:

$$\mathbf{W} = \begin{pmatrix} 0 & 1 & 2 & 1 & 2 & 3 & 2 & 3 & 4 \\ 1 & 0 & 1 & 2 & 1 & 2 & 3 & 2 & 3 \\ 2 & 1 & 0 & 3 & 2 & 1 & 4 & 3 & 2 \\ 1 & 2 & 3 & 0 & 1 & 2 & 1 & 2 & 3 \\ 2 & 1 & 2 & 1 & 0 & 1 & 2 & 1 & 2 \\ 3 & 2 & 1 & 2 & 1 & 0 & 3 & 2 & 1 \\ 2 & 3 & 4 & 1 & 2 & 3 & 0 & 1 & 2 \\ 3 & 2 & 3 & 2 & 1 & 2 & 1 & 0 & 1 \\ 4 & 3 & 2 & 3 & 2 & 1 & 2 & 1 & 0 \end{pmatrix}$$

The data space region Ω is divided into 40 grid cells, i.e., 39 grid boundaries excluding the boundaries of Ω. The sequential partitioning optimization procedure described in Subsection 3.1 is then employed to identify the optimal partitioning. Figure 3.5 depicts the optimization process for obtaining the optimal partitioning, where λ_1^* is evaluated by minimizing $CostO(\Lambda_2)$. Both the cost curve and its

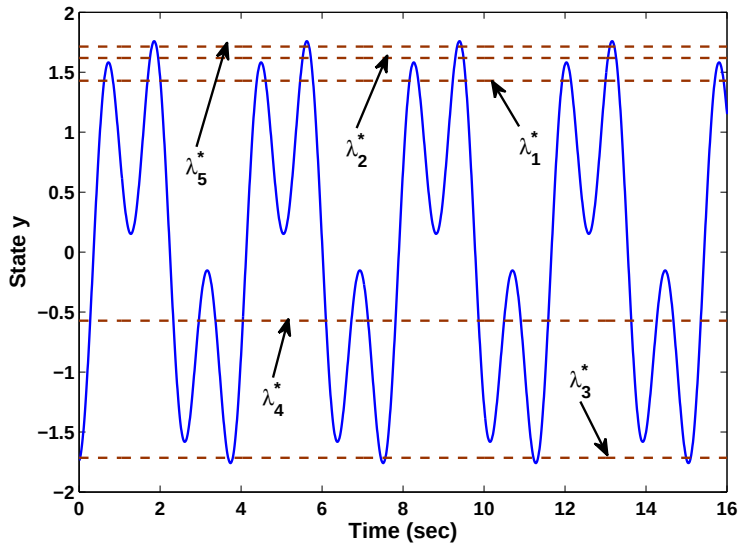


Figure 3.6. Optimal Partitioning marked on the data space with a representative time-series from Class 5

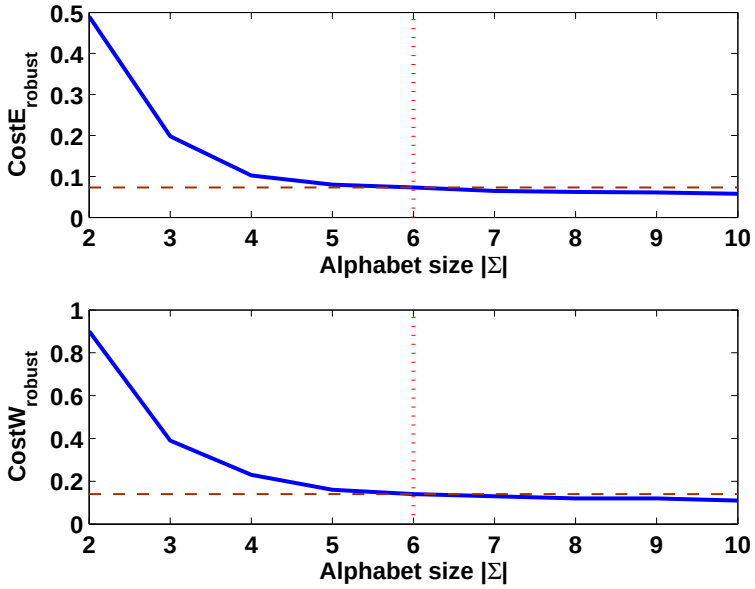


Figure 3.7. Decrease in $CostE$ and $CostW$ with increase in Alphabet size, for Optimal Partitioning $|\Sigma|$ is chosen to be 6

corresponding optimal value λ_1^* are shown in Fig. 3.5. Similarly, the second optimal partitioning boundary λ_2^* is obtained by minimizing the cost function $CostO(\Lambda_3) \triangleq CostO(\{\lambda_1^*, \lambda_2\})$. As described in Subsection 3.1, λ_1^* is kept fixed while λ_2 is optimized. This suboptimal process is recursively continued until the threshold $\eta_{stop} = 0.01$ is reached, which leads to the creation of 6 cells (i.e., 5 partitions) denoted by $\Lambda_6^* = \{\lambda_1^*, \cdots, \lambda_5^*\}$ as shown in Fig. 3.5. For SDF analysis, the depth for constructing $PFSA$ states is taken to be, $D = 1$ and features are classified by a k-NN classifier (with $k = 5$) using the Euclidean distance metric. Also, for estimation of $CostE_{robust}$ and $CostW_{robust}$, σ_λ is taken to be $l_\lambda = 0.0333$ and 50 perturbed samples are taken for each partitioning elements in the search space. Choice of such σ_λ leads to smooth cost curves across the *State* y values (domain space) as seen in Fig. 3.5.

Figure 3.6 plots the optimal partitioning Λ_6^* on a representative time-series from the reference class 5. Finally, the decrease in $CostE_{robust}$ and $CostW_{robust}$ with the increase in alphabet size is shown in Fig. 3.7. The optimal alphabet size and corresponding cost values are marked on each plate. The confusion matrix obtained by using the optimal partitioning (OptP) on the test data set is given below.

$$\mathbf{C}_{test}^{OptP} = \left(\begin{array}{c|ccccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ \hline 1 & 98 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 3 & 92 & 5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 4 & 96 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 6 & 0 & 0 & 88 & 5 & 0 & 1 & 0 & 0 \\ 5 & 0 & 0 & 0 & 3 & 84 & 12 & 0 & 1 & 0 \\ 6 & 0 & 1 & 1 & 0 & 1 & 94 & 0 & 0 & 3 \\ 7 & 0 & 0 & 0 & 4 & 0 & 0 & 92 & 4 & 0 \\ 8 & 0 & 0 & 0 & 1 & 4 & 0 & 8 & 83 & 4 \\ 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 13 & 87 \end{array}\right)$$

It is observed that the class separability is retained in an efficient way by the nonlinear feature extraction (partitioning) process even after compressing a time series data (with 8,000 data points) into a 6-dimensional feature (state probability)

vector. The confusion matrices for Uniform and Maximum Entropy partitioning on the test data set are also provided below for comparison.

$$\mathbf{C}_{test}^{UP} = \left(\begin{array}{c|ccccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ \hline 1 & 84 & 10 & 5 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 7 & 87 & 4 & 1 & 1 & 0 & 0 & 0 & 0 \\ 3 & 14 & 3 & 77 & 1 & 5 & 0 & 0 & 0 & 0 \\ 4 & 10 & 1 & 2 & 76 & 11 & 0 & 0 & 0 & 0 \\ 5 & 0 & 1 & 0 & 16 & 79 & 3 & 0 & 1 & 0 \\ 6 & 0 & 0 & 3 & 3 & 3 & 84 & 0 & 2 & 5 \\ 7 & 0 & 0 & 0 & 2 & 2 & 0 & 88 & 8 & 0 \\ 8 & 0 & 0 & 0 & 0 & 3 & 0 & 13 & 76 & 8 \\ 9 & 0 & 0 & 0 & 0 & 0 & 1 & 3 & 11 & 85 \end{array}\right)$$

$$\mathbf{C}_{test}^{MEP} = \left(\begin{array}{c|ccccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ \hline 1 & 83 & 12 & 4 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 13 & 82 & 3 & 2 & 0 & 0 & 0 & 0 & 0 \\ 3 & 2 & 5 & 87 & 1 & 5 & 0 & 0 & 0 & 0 \\ 4 & 1 & 1 & 4 & 85 & 3 & 6 & 0 & 0 & 0 \\ 5 & 0 & 2 & 0 & 9 & 84 & 5 & 0 & 0 & 0 \\ 6 & 0 & 0 & 0 & 0 & 10 & 81 & 0 & 4 & 5 \\ 7 & 0 & 0 & 0 & 2 & 0 & 1 & 86 & 11 & 0 \\ 8 & 0 & 0 & 0 & 0 & 0 & 8 & 11 & 74 & 7 \\ 9 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 10 & 87 \end{array}\right)$$

Table 3.8 presents the comparison of the classification error related costs for OptP, UP and MEP on the test data set.

Table 3.1. Comparison of Classification Performances of Partitioning Schemes on Test-data set (100×9 samples)

Partitioning	$CostE$	$CostW$
OptP	0.0978	0.1800
UP	0.2289	0.4400
MEP	0.2200	0.3400

The observations made from these results indicate that the classification per-

formance may be improved compared to that of the classical partitioning schemes by optimizing the partitioning process over a representative training set for the particular problem at hand. However, it should be noted that for some problems, the classical partitioning schemes may perform as well as the optimal one. Therefore, the optimization procedure may also be used to evaluate the capability of any partitioning scheme towards achieving a better classification rate. The evaluation can be performed by using a part of the labeled training data set as the validation set. Finally, although the construction of the cost functions theoretically allow problems with large number of classes, in practice it should be understood that its upper limit will be constrained by the alphabet size used for partitioning which is also the dimension of the feature space. Also note that the model complexity of a probabilistic finite state automaton ($PSFA$), as obtained from time series data, is related to the number of states (hence, to the number of partitions) in the $PSFA$. Therefore, the choice of η_{stop} is critical in our approach during the process of partitioning optimization to alleviates the issue of over-training.

5 Validation Example 2: Data-driven Fault Diagnosis in Aircraft Gas Turbine Engines

Component-level fault diagnosis in an aircraft gas turbine engine involves identification of the fault type, and location & quantification of the fault level. In the C-MAPSS test-bed (detailed in Appendix-A) setting, the physical fault scenarios (e.g., fouling, increased tip clearance, and seal wear) are assumed to manifest themselves in reducing the efficiency and degrade the flow properties of the associated rotating component(s) of the engine. Although the present problem formulation can be easily extended for simultaneous faults in multiple components (see [84] for details), the present case studies deal with a single component faults and the task is to detect a fault and identify its level under varying sensor noise condition. A major challenge in any sensor-data-driven detection tool is to identify the actual failure in the system in the presence of sensor degradation (e.g., drift and noise) without succumbing to a large number of false alarms or missed detections. The situation becomes even more critical if the control system uses observations

from the degraded sensors as feed-back signals and generates the control inputs accordingly. Traditionally, redundant sensors along with methods based on analytic redundancy are used for sensor fault identification. A different approach to this problem is adopted in this chapter to utilize the partitioning optimization technique. Different class labels are assigned to data sets that are generated from different engine health conditions. The same class labels are assigned to data from engines with similar health conditions and the associated sensor data are subjected to different noise variance at respective sensor degradation levels. To this end, the data-driven fault detection problem is posed as a multi-class pattern classification problem, where the tasks of class assignment and feature extraction via partitioning are optimized in a supervised manner to enhance the performance of component-level fault diagnosis under varying sensor noise condition [13].

5.1 Case Study 1: HPC Fault Diagnosis

The two health parameters that define the health status of HPC, are the efficiency and flow health parameters (ψ_{HPC}, ζ_{HPC}). However, it is observed that Ps_{30} sensor observation does not capture any signature of change in the flow health parameter ζ_{HPC} and since, only sensor Ps_{30} is used for fault diagnosis in this example, three fault levels are considered only based on the efficiency health parameter ψ_{HPC}. Table 3.2 shows the approximate ranges of efficiency health parameters under different fault levels. Here, the low fault level indicates very minimal loss in efficiency and hence also includes the absolute nominal health condition ($\psi_{HPC} = 1$).

Table 3.2. Fault Levels in HPC

Fault Level	**Efficiency Range (ψ_{HPC})**
Low Fault	1.0000 to 0.9834
Medium Fault	0.9834 to 0.9668
High Fault	0.9668 to 0.9500

Similarly, depending on the noise standard deviation in the Ps_{30} sensor, three sensor noise levels are considered for the study. Table 3.3 shows the approximate ranges of sensor noise standard deviation as percent of operating point trim values considered under different levels [85][86]. Note that although the noise is additive in nature, its standard deviation values depend on the operating point trim values

in a multiplicative sense.

Table 3.3. Variable Noise Levels in Ps_{30}

Noise Level	Standard Deviation Range (%)
Level 1	0.35 to 0.45
Level 2	0.45 to 0.55
Level 3	0.55 to 0.65

Thus, in the above context, there are (3×3)=9 classes of data sets that need to be obtained to define a class by a HPC fault level and a noise level of the Ps_{30} sensor. Seventy simulation runs of the engine system were performed for each class to generate data sets for analysis, among which 50 samples are chosen as the training set and the rest of the samples are kept as the testing set. HPC efficiency and Ps_{30} standard deviation parameters are chosen randomly from independent Gaussian distributions for both parameters, such that approximately 95% of the parameter values are within the prescribed ranges given in Tables 3.2 and 3.3. In other words, the mean of the Gaussian distribution used for a particular parameter level is taken as the central value of the parameter range in that level and the standard deviation is taken such that the boundary values of the parameter range are 2σ distance away from the central value. For example, for the class with low fault in HPC and level 1 noise in Ps_{30}, the HPC efficiency values were chosen from an independent Gaussian distribution with mean as 0.9917 and standard deviation as 4.15×10^{-3}, whereas the standard deviation percentage values were chosen from another independent Gaussian distribution with mean as 0.40 and standard deviation as 0.025. Figure 4.5 plots the samples generated using the above logic in the two dimensional parametric space. Different classes of samples are shown in different colors in the figure. The axis for sensor noise standard deviation ($\sigma_{Ps_{30}}$) represents actual standard deviation values (not as percent of the operating point trim values of Ps_{30} reading).

For each data sample, a time series was collected for Ps_{30} sensor under persistent excitation of TRA inputs that have truncated triangular profiles with the mean value of 40°, fluctuations within ±8° and frequency of 0.056 Hz as shown in Fig. 3.9. The ambient conditions are chosen to be at the sea level when the engine is on the ground (i.e. altitude $a = 0.0$, Mach number $M = 0.0$) for fault monitoring and maintenance by the engineering personnel. For each experiment,

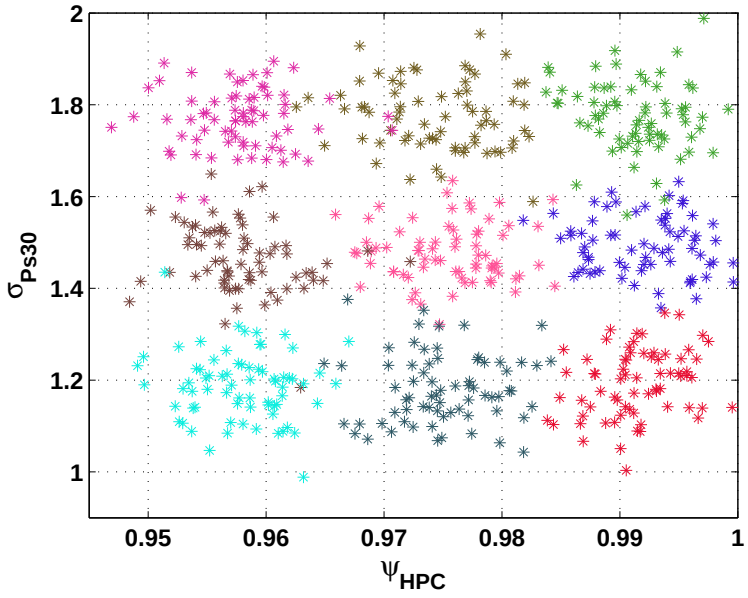


Figure 3.8. Original class labels for data collection

the engine simulation is conducted at a frequency of 66.67 Hz (i.e., inter-sample time of $15ms$) and the length of the simulation time window is 150 seconds, which generate $10,000$ data points. Figure 4.6 shows representative examples of Ps_{30} time series data from each of the nine classes.

As stated earlier, the objective of this study is to build a data-driven diagnostic algorithm that is robust to varying sensor noise level, provided that the sensor noise is within an allowable range. From this perspective, the definition of a data class is changed and made only dependent on the HPC efficiency parameters. Thus, the 9 original classes are reduced to 3 classes as shown in Fig. 3.11. This is the final class assignment for the data set, where each class has (50×3)=150 training samples and (20×3)=60 testing samples.

Thus, in the above context, the problem of component level fault detection in presence of varying sensor noise condition in aircraft gas turbine engines is formulated as a multi-class classification problem (in the present scenario, number of classes is 3).

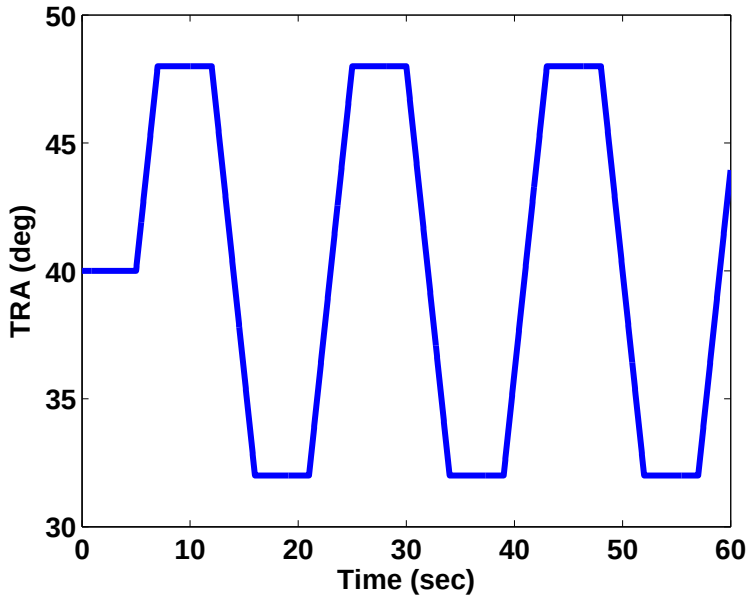


Figure 3.9. Profile of throttle resolving angle (TRA)

5.1.1 Results and Discussion

At the beginning of the optimization procedure, a weighting matrix $\mathbf{W}$ needs to be defined to calculate the costs $Cost_E$ and $Cost_W$ from the confusion matrix for the training data set. In this case study, $\mathbf{W}$ is defined according to the adjacency properties of classes in the parameter space, i.e., $w_{ii} = 0\ \forall i \in \{1, 2, 3\}$, i.e. there is no penalty for correct classification. The weights are selected as: $w_{ij} = |i-j|,\ \forall i \in \{1, 2, 3\}$, i.e., given that a data sample originally from Cl_i is classified as a member of Cl_j, the penalty incurred by the classification process increases with increase in the separation of Cl_i and Cl_j in the parameter space. Then, it follows that:

$$\mathbf{W} = \begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 1 \\ 2 & 1 & 0 \end{pmatrix}$$

The data space region Ω is divided into 100 grid cells, i.e., 99 grid boundaries excluding the boundaries of Ω. The sequential partitioning optimization procedure is then employed to identify the optimal partitioning. The threshold value η_{stop} for stopping the algorithm is chosen to be 0.001 and the optimal alphabet size is found to be, $|\Sigma| = 4$. For SDF analysis, the depth for constructing $PFSA$ sates is taken

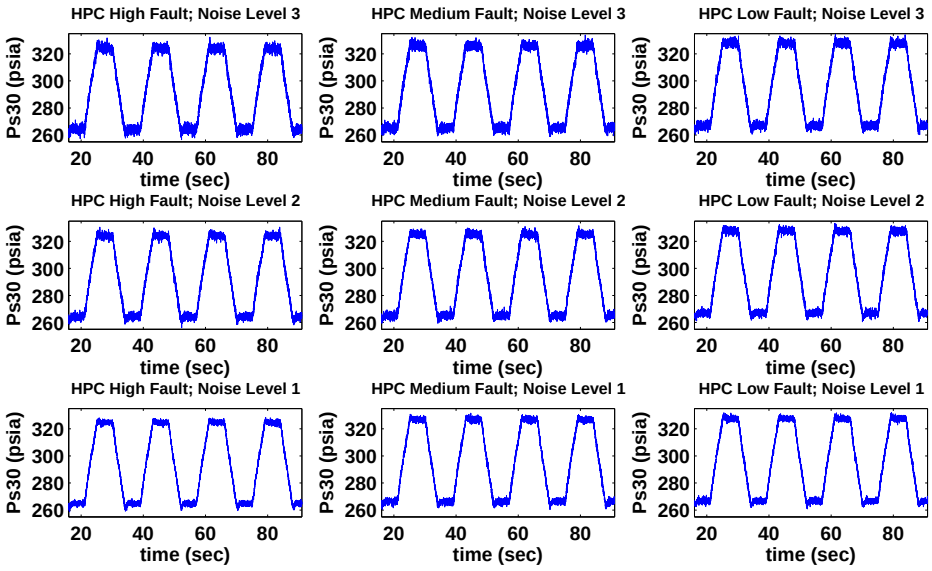


Figure 3.10. Representative time series data for HPC fault and Ps_{30} degradation conditions

to be, $D = 1$ and features are classified by a k-NN classifier (with $k = 5$) using the Euclidean distance metric. Figure 3.12 shows locations of the training features in the three-dimensional plot using first three linearly independent elements of the feature vectors obtained by using the chosen optimal partitioning, OptP. Note, only $(|\Sigma| - 1)$ out of its $|\Sigma|$ elements of a feature vector are linearly independent, because a training feature vector, $\mathbf{p}$ is also a probability vector, i.e., the sum of its elements is constrained to be equal to 1. The class separability is retained by the feature extraction (partitioning) process even after compressing a time series (with 10,000 data points) into 3 numbers.

For comparison purpose, classical partitioning schemes, such as, Uniform Partitioning (UP) and Maximum Entropy Partitioning (MEP) are also used with the same alphabet size, $\Sigma = 4$. Figures 3.13 and 3.14 show the location of each training time series in the three dimensional (using first three linearly independent elements of the feature vectors) feature space plot using Uniform partitioning and Maximum Entropy partitioning, respectively.

Finally, the confusion matrices for the Optimal, Uniform and Maximum Entropy Partitioning on the test data set are given by $\mathbf{C}_{test}^{OptP}$, $\mathbf{C}_{test}^{UP}$ and $\mathbf{C}_{test}^{MEP}$ re-

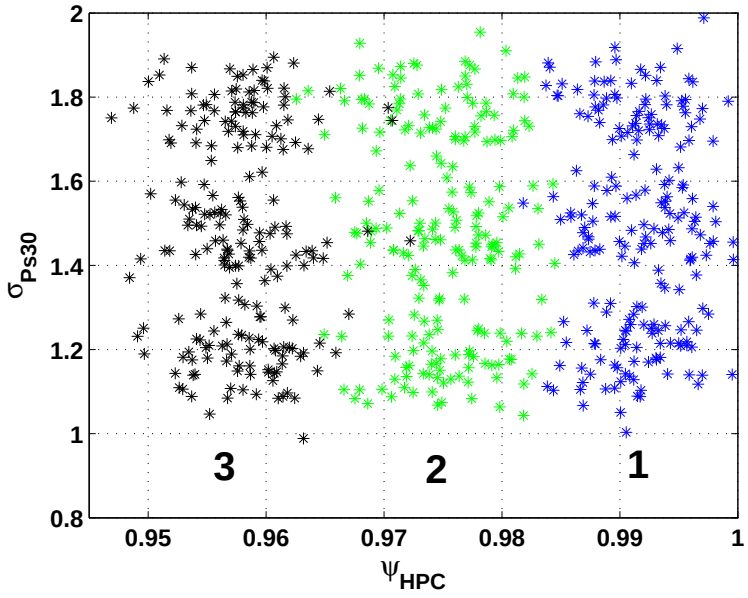


Figure 3.11. Revised class assignment for fault detection

spectively.

$$\mathbf{C}_{test}^{OptP} = \left(\begin{array}{c|ccc} & 1 & 2 & 3 \\ \hline 1 & 58 & 2 & 0 \\ 2 & 1 & 57 & 2 \\ 3 & 0 & 2 & 58 \end{array}\right)$$

$$\mathbf{C}_{test}^{UP} = \left(\begin{array}{c|ccc} & 1 & 2 & 3 \\ \hline 1 & 58 & 2 & 0 \\ 2 & 4 & 54 & 2 \\ 3 & 0 & 1 & 59 \end{array}\right)$$

$$\mathbf{C}_{test}^{MEP} = \left(\begin{array}{c|ccc} & 1 & 2 & 3 \\ \hline 1 & 57 & 3 & 0 \\ 2 & 7 & 51 & 2 \\ 3 & 0 & 3 & 57 \end{array}\right)$$

Table 3.8 compares the values of $Cost_E$ and $Cost_W$ for OptP, UP and MEP on the test set. It is interesting to notice that although the optimization is performed based on only minimization of $Cost_E$, both $Cost_E$ and $Cost_W$ have reduced compared to the classical partitioning schemes. This fact can be attributed to the

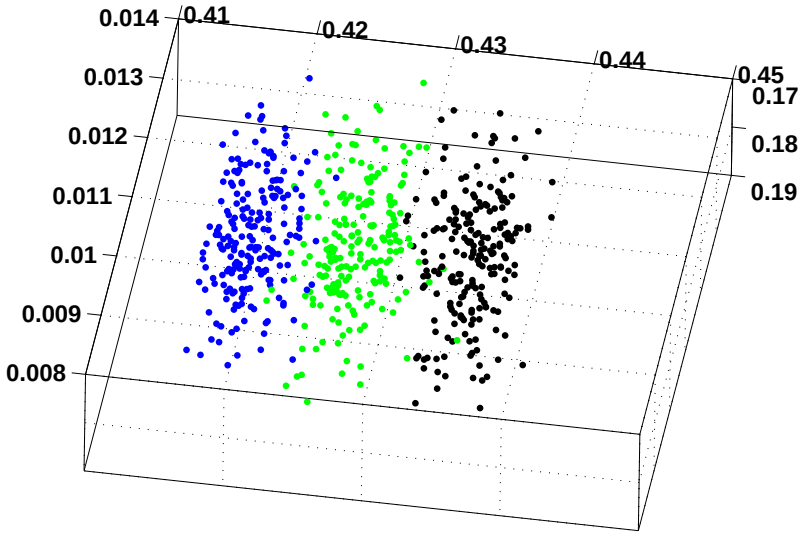


Figure 3.12. Feature space of the training set using optimal partitioning

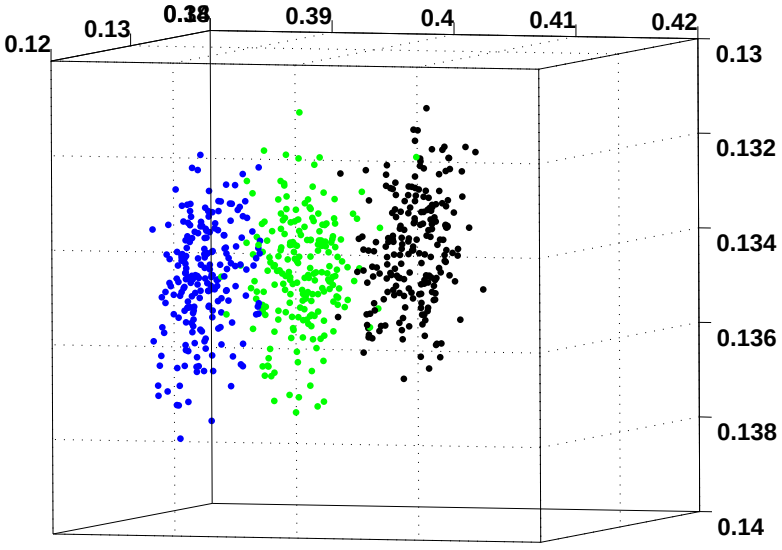


Figure 3.13. Feature space of training set – uniform partitioning (UP)

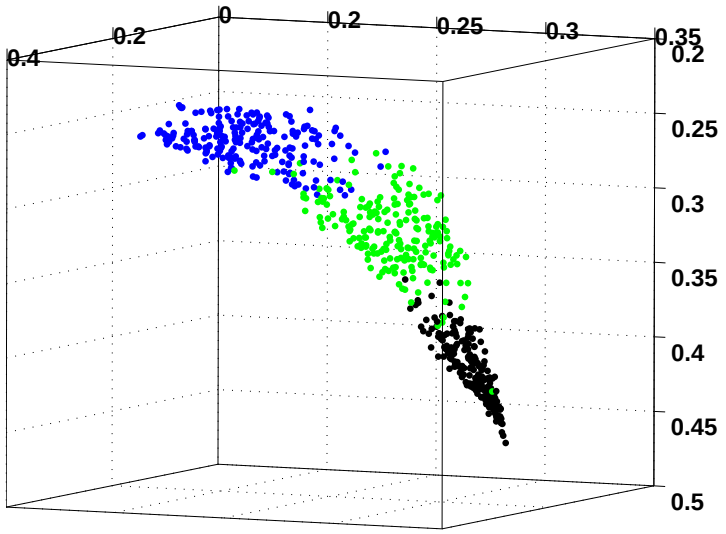


Figure 3.14. Feature space of training set – maximum entropy partitioning (MEP)

nature of the current data set. Thus, HPC fault levels can be efficiently determined

Table 3.4. Comparison of Classification Performances of Different Partitioning Schemes on Test Data Set (60×3 samples)

Partitioning	$Cost_E$	$Cost_W$
OptP	0.0389	0.0500
UP	0.0500	0.1000
MEP	0.0833	0.1500

under varying noise levels using the SDF methodology and optimization of partitioning may potentially improve its performance over other classical partitioning techniques.

5.2 Case Study 2: HPT Fault Diagnosis

This subsection presents a case study of fault detection in the high pressure turbine (HPT) component only by monitoring temperature sensor T_{48}, that is located at HPT exit. A representative example of T_{48} time series data is shown in Fig. 3.15. It has been observed in the previous case study for HPC, that sensor Ps_{30} observation only captures signature of change in efficiency health parameter (ψ_{HPC}) and does

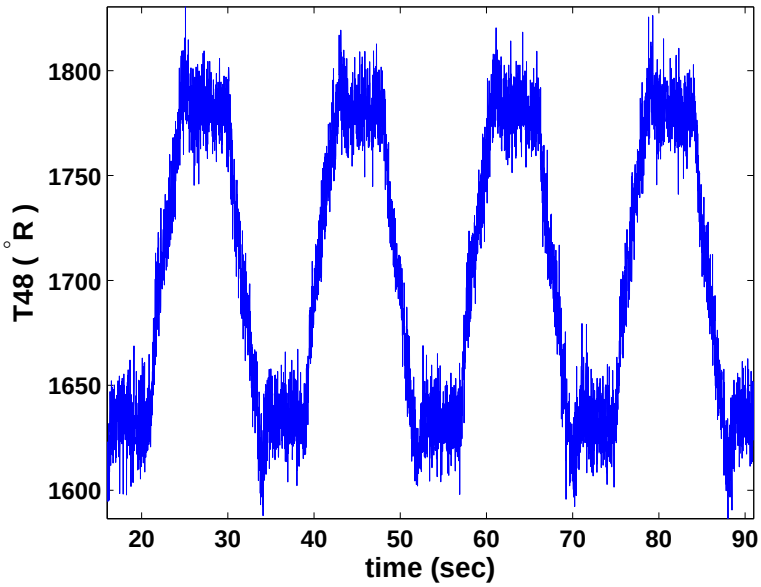


Figure 3.15. Representative Sensor T_{48} Observation

not capture signature of change in flow health parameters (ζ_{HPC}). In contrast, T_{48} observation captures signature of change in both efficiency health parameters (ψ_{HPT}) and in flow health parameter (ζ_{HPT}). However, ζ_{HPT} signature is much weaker compared to that of ψ_{HPT}. Therefore, this case study will investigate classification of both the fault modes.

Table 3.5 shows the approximate ranges of efficiency and flow health parameters of different classes considered for the study. Similarly, different (additive) noise

Table 3.5. Fault Classes in HPT

Efficiency Class Label	**Class Label**	**Efficiency Range (ψ_{HPT})**	**Flow Range (ζ_{HPT})**
Efficiency Class 1	Class 1	1.0000 to 0.9834	1.0000 to 0.9834
	Class 2	1.0000 to 0.9834	0.9834 to 0.9668
	Class 3	1.0000 to 0.9834	0.9668 to 0.9500
Efficiency Class 2	Class 4	0.9834 to 0.9668	1.0000 to 0.9834
	Class 5	0.9834 to 0.9668	0.9834 to 0.9668
	Class 6	0.9834 to 0.9668	0.9668 to 0.9500
Efficiency Class 3	Class 7	0.9668 to 0.9500	1.0000 to 0.9834
	Class 8	0.9668 to 0.9500	0.9834 to 0.9668
	Class 9	0.9668 to 0.9500	0.9668 to 0.9500

levels (i.e., approximate ranges of sensor noise standard deviation as percent of operating point trim values [85] [86]) in T_{48} observation are shown in Table 3.6. Thus, in this context, there are $(9 \times 3) = 27$ classes of data sets that need to

Table 3.6. Variable Noise Levels in T_{48}

Noise Level	Standard Deviation Range (%)
Level 1	0.45 to 0.55
Level 2	0.55 to 0.65
Level 3	0.65 to 0.75

be obtained to define a class by a HPT fault level and a noise level of the T_{48} sensor. Seventy simulation runs of the engine system were performed for each class to generate data sets for analysis, among which 50 samples are chosen as the training set and the rest of the samples are kept as the testing set. The samples are generated in the same way as in the Case Study 1. As different sensor noise level classes are merged into one, there are only 9 classes for classification purpose with 150 training samples and 60 testing samples in each class.

5.2.1 Results and Discussion

The first step for classification is to define the penalty weighting matrix $\mathbf{W}$, that involves prior knowledge about the data. As shown in Table 3.5, for each efficiency class, there are three flow classes. However, it is observed that both efficiency and flow degradation signatures are of similar nature in time domain, with efficiency degradation signatures being quantitatively stronger than the flow degradation signatures. Therefore, classification among flow classes under one efficiency level is potentially a very hard problem under the current high noise environment. Thus, the primary goal of the classification process is to classify different flow classes under different efficiency classes. The penalty weighting matrix is defined accordingly. The 3 diagonal blocks represent penalty values among different flow classes in one efficiency class which are defined similarly as in the previous case study. The off-diagonal blocks represent the penalty values for different flow classes under different efficiency classes. The symmetric weighting matrix for classification

process is given below.

$$\mathbf{W} = \begin{pmatrix} 0 & 1 & 2 & 2 & 4 & 6 & 3 & 6 & 9 \\ 1 & 0 & 1 & 4 & 2 & 4 & 6 & 3 & 6 \\ 2 & 1 & 0 & 6 & 4 & 2 & 9 & 6 & 3 \\ 2 & 4 & 6 & 0 & 1 & 2 & 2 & 4 & 6 \\ 4 & 2 & 4 & 1 & 0 & 1 & 4 & 2 & 4 \\ 6 & 4 & 2 & 2 & 1 & 0 & 6 & 4 & 2 \\ 3 & 6 & 9 & 2 & 4 & 6 & 0 & 1 & 2 \\ 6 & 3 & 6 & 4 & 2 & 4 & 1 & 0 & 1 \\ 9 & 6 & 3 & 6 & 4 & 2 & 2 & 1 & 0 \end{pmatrix}$$

The sequential optimization process has been carried out for this problem and it has been found that the optimal alphabet size is 7, with the algorithm stopping threshold value, $\eta_{stop} = 0.005$. The confusion matrices for Optimal, Uniform and Maximum Entropy Partitioning on the test data set are given by $\mathbf{C}_{test}^{OptP}$ $\mathbf{C}_{test}^{UP}$ and $\mathbf{C}_{test}^{MEP}$ respectively. Table 4.4 shows the comparison of classification performances using different partitioning processes. However, the cost values in this table should not be compared with those in Tab. 3.8 as the range of cost values is a function of the weighting matrix used for the optimization process.

$$\mathbf{C}_{test}^{OPT} = \left(\begin{array}{c|ccccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ \hline 1 & 35 & 21 & 3 & 0 & 1 & 0 & 0 & 0 & 0 \\ 2 & 11 & 24 & 24 & 0 & 0 & 1 & 0 & 0 & 0 \\ 3 & 3 & 22 & 35 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 41 & 17 & 2 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 19 & 22 & 19 & 0 & 0 & 0 \\ 6 & 1 & 0 & 0 & 7 & 21 & 31 & 0 & 0 & 0 \\ 7 & 0 & 0 & 0 & 0 & 0 & 0 & 39 & 17 & 4 \\ 8 & 0 & 0 & 0 & 0 & 0 & 0 & 16 & 25 & 19 \\ 9 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 21 & 35 \end{array}\right)$$

$$\mathbf{C}_{test}^{UP} = \left(\begin{array}{c|ccccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ \hline 1 & 32 & 22 & 5 & 0 & 0 & 1 & 0 & 0 & 0 \\ 2 & 15 & 22 & 22 & 0 & 0 & 1 & 0 & 0 & 0 \\ 3 & 3 & 17 & 40 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 37 & 16 & 7 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 21 & 15 & 24 & 0 & 0 & 0 \\ 6 & 1 & 0 & 0 & 4 & 18 & 37 & 0 & 0 & 0 \\ 7 & 0 & 0 & 0 & 0 & 0 & 0 & 30 & 17 & 13 \\ 8 & 0 & 0 & 0 & 0 & 0 & 0 & 21 & 21 & 18 \\ 9 & 0 & 0 & 0 & 1 & 0 & 0 & 12 & 27 & 20 \end{array}\right)$$

$$\mathbf{C}_{test}^{MEP} = \left(\begin{array}{c|ccccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ \hline 1 & 37 & 18 & 4 & 0 & 0 & 1 & 0 & 0 & 0 \\ 2 & 16 & 24 & 19 & 0 & 0 & 1 & 0 & 0 & 0 \\ 3 & 5 & 21 & 34 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 30 & 23 & 7 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 17 & 23 & 20 & 0 & 0 & 0 \\ 6 & 1 & 0 & 0 & 7 & 19 & 33 & 0 & 0 & 0 \\ 7 & 0 & 0 & 0 & 0 & 0 & 0 & 30 & 18 & 12 \\ 8 & 0 & 0 & 0 & 0 & 0 & 0 & 23 & 21 & 16 \\ 9 & 0 & 0 & 0 & 1 & 0 & 0 & 12 & 18 & 29 \end{array}\right)$$

Table 3.7. Comparison of Classification Performances of Different Partitioning Schemes on Test Data Set (60×9 samples)

Partitioning	$Cost_E$	$Cost_W$
OptP	0.5315	0.6833
UP	0.6444	0.9500
MEP	0.6370	0.8000

It is observed from these results that samples of different flow classes under different efficiency classes are efficiently classified. But as expected, the classification performance degrades for different flow classes under one efficiency level. However, the optimization process improves the performance of SDF compared to the classical partitioning methods. With a close inspection of the confusion

matrix corresponding to the optimal partitioning, it can be observed that the confusion between low-level faults and high-level fault classes for HPT flow under same efficiency level are significantly reduced by the partitioning optimization.

6 Validation Example 3: Fatigue Damage Classification

Fatigue is broadly classified into two phases; namely crack initiation and crack propagation. As damage mechanism of these two phases are significantly different and similar detection methods may not work effectively to classify different damage levels in these two phases. Damage evolution in crack initiation phase is much slower, resulting in smaller change in ultrasonic signals as compared to that in the crack propagation phase. So to get maximum information from the ultrasonic data sequences, different partitionings are required in these two phases. Damage classification schemes for these phases are defined in the sequel.

6.1 Crack initiation

Since crack initiation predominantly forms a significant portion of the total life [87], especially in the high cycle fatigue, the quantification of fatigue damage during crack initiation is of paramount importance for safety, reliability, and maintenance of mechanical and aerospace structures. As this phase is caused by multiple small cracks, dislocations and other defects, identifying any direct physical parameter to define damage in this regime is difficult. In the present classification method, different classes are defined based on the fraction of crack initiation life. This is defined as the number of load cycles that a specimen sustains before appearance of the crack at surface. In this work, crack initiation is divided into three classes :

Class	Used crack initiation life fraction
1	0 to 0.5
2	0 .5 to 0.8
3	0.8 to 1.0

Identifying the class of damage is very critical for maintenance as well as for control operation. If the component is in first class of damage then it means that

more than half (50%) of life is still remaining. Therefore, it is safe to continue the operation. While the third class suggests that more than 80% of life is consumed and hence, crack may appear soon. In this situation, alarm should be sent to the controller in order to take appropriate action for safe operation. This can also be a basis for the maintenance schedule.

6.2 Crack Propagation

The phase transition from crack initiation to crack propagation occurs when several small microcracks coalesce together to develop a single large crack that propagates under an oscillating load. Several crack propagation models have been developed based on the inherent stochastic nature of the fatigue damage evolution for prediction of the remaining useful life. Due to the stochastic nature of material properties and operating conditions, any physical model requires the knowledge of some parameter associated with the particular specimen or component. These parameters are random and can not be known beforehand. Also, crack propagation rate is a function of crack length; after a certain length, crack propagates unstably leading to catastrophic failure. So appropriate action must be taken before a crack attains a critical size. Therefore, in this classification problem, damage is classified on basis of crack length. The crack propagation stage is divided into four classes:

Class	Crack Length
1	0.5 to 1.75 mm
2	1.75 to 3.5 mm
3	3.5 to 5.5 mm
4	more than 5.5 mm

In the present work, a large number of ultrasonic data have been collected for different crack lengths on different specimens to demonstrate the efficiency of the present classification scheme in the presence of material properties stochastic nature. The following subsection presents the details of the experimental apparatus used for damage detection using ultrasonic sensors. It also describes the test specimens and the test procedure.

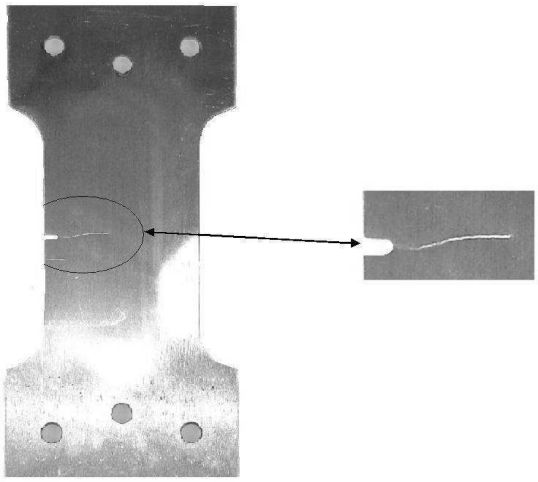


Figure 3.16. Cracked specimen with a side notch

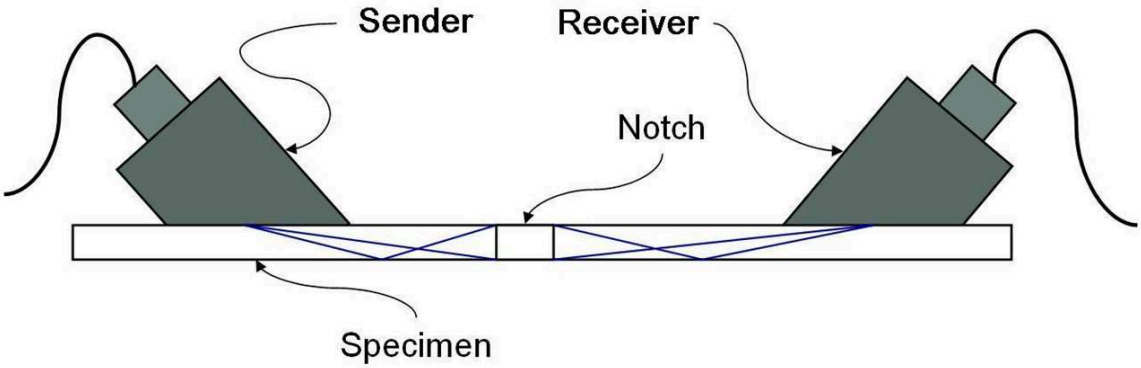


Figure 3.17. Ultrasonic flaw detection scheme

6.3 Experimental Apparatus and Test Procedure

The experimental apparatus is designed to study the fatigue damage growth in mechanical structures. The apparatus consists of an MTS 831.10 Elastomer Test System that is integrated with Olympus BX Series microscope with a long working-distance objective. A camera, mounted on the microscope, takes images with a resolution of 2 micron per pixel at a distance of 20mm. The ultrasonic sensing device is triggered at a frequency of 5 MHz at each peak of the fluctuating load. Various components of the apparatus communicate over a TCP/IP network and post sensor, microscope and fatigue test data on the network in real time. This data can used by analysis algorithms for anomaly detection and health monitoring of the specimens in real-time.

Figure 3.16 shows a side-notched 7075-T6 aluminum alloy specimen used in the fatigue damage test apparatus. Each specimen is 3 mm thick and 50 mm wide, and has a slot of 1.58 $mm \times 4.57$ mm on one edge. The notch is made to increase the stress concentration factor that localizes crack initiation and propagation under the fluctuating load. Fatigue tests were conducted at a constant amplitude sinusoidal load for low-cycle fatigue, where the maximum and minimum loads were kept constant at 87MPa and 4.85MPa, respectively. For low cycle fatigue studied in this chapter, the stress amplitude at the crack tip is sufficiently high to observe the elasto-plastic behavior in the specimens under cyclic loading. A significant amount of internal damage caused by multiple small cracks, dislocations and microstructural defects alters the ultrasonic impedance, which results in signal distortion and attenuation at the receiver end.

The optical images were collected automatically at every 200 cycles by the optical microscope which is always focussed in the crack tip. As soon as crack is visible by the microscope, crack length is noted down after every 200 cycles. Ultrasonic waves with a frequency of 5 MHz were triggered at each peak of the sinusoidal load to generate data points in each cycle. Since the ultrasonic frequency is much higher than the load frequency, data acquisition was done for a very short interval in the time scale of load cycling. Therefore, it can be implied that ultrasonic data were collected at the peak of each sinusoidal load cycle, where the stress is maximum and the crack is open causing maximum attenuation of the ultrasonic waves. The slow time epochs for data analysis were chosen to be 1000 load cycles (i.e., $\sim$80 sec) apart. To generate training and test data sample multiple experiments are conducted on different specimen. For each specimen all ultrasonic signals are labeled with crack length.

6.4 Results and Discussion

This section presents the classification results for the two stages of fatigue damage evolution : (i) crack initiation and (ii) crack propagation. The damage mechanisms are entirely different in these two regimes and this is evident from the big change in the ultrasonic signal responses in these two phases. For classification, at first time series data are processed via wavelet transform at a particular scale, time shift

and basis function selected appropriately. Each transformed signal is normalized with respect to the maximum amplitude of the transformed signal obtained at the beginning of experiment, when there is no damage in the specimen. The normalization is done to remove the effect of variability in placement of ultrasonic sensors during different experiments.

6.5 Crack Initiation

The fatigue crack initiation phase is divided into three classes based on the

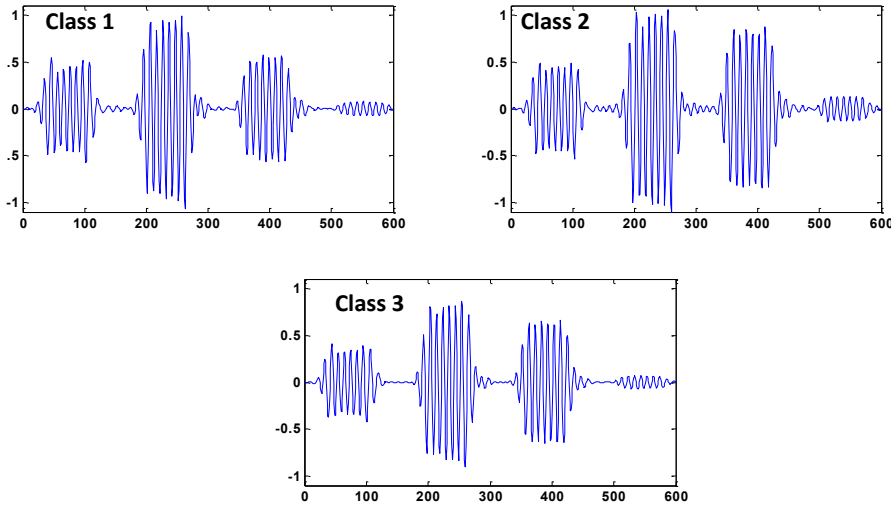


Figure 3.18. Representative signal from different classes in crack initiation phase

fraction of crack initiation life. The three classes represent different life fraction as discussed earlier. Fig. 3.18 shows the representative test data sample for each class. For partitioning the data, a weighting matrix **W** needs to be defined to calculate the costs $Cost_E$ and $Cost_W$ from the confusion matrix for the training data set. The weights w_{ij} are defined as penalty for classifying a data sample originally from Cl_i is classified as a member of Cl_j. The weighting matrix is chosen such that it follows two rules in the case of fatigue damage classification: (i) $w_{ii} = 0 \ \forall i \in \{1, 2, 3\}$, i.e. there is no penalty for correct classification and (ii) $w_{ij} > w_{ji}$; if $i > j$, i.e., penalty for classifying a higher level damage as a lower

level damage is much more than the other way round. Here, W is chosen as:

$$\mathbf{W} = \begin{pmatrix} 0 & 1 & 2 \\ 3 & 0 & 1 \\ 6 & 3 & 0 \end{pmatrix}$$

The data space region Ω is divided into 100 grid cells, i.e., 99 grid boundaries excluding the boundaries of Ω. The sequential partitioning optimization procedure is then employed to identify the optimal partitioning. The threshold value η_{stop} for stopping the algorithm is chosen to be 0.001 and the optimal alphabet size is found to be, $|\Sigma| = 7$. For SDF analysis, the depth for constructing $PFSA$ sates is taken to be, $D = 1$ and features are classified by a k-NN classifier (with $k = 5$) using the Euclidean distance metric.

For comparison purpose, classical partitioning schemes, such as, Uniform Partitioning (UP) and Maximum Entropy Partitioning (MEP) are also used with the same alphabet size, $\Sigma = 7$. Finally, the confusion matrices for the Optimal, Uniform and Maximum Entropy Partitioning on the test data set are given by $\mathbf{C}_{test}^{OptP}$, $\mathbf{C}_{test}^{UP}$ and $\mathbf{C}_{test}^{MEP}$ respectively.

$$\mathbf{C}_{test}^{OptP} = \left(\begin{array}{c|ccc} & 1 & 2 & 3 \\ \hline 1 & 95 & 5 & 0 \\ 2 & 4 & 91 & 5 \\ 3 & 0 & 9 & 91 \end{array}\right)$$

$$\mathbf{C}_{test}^{UP} = \left(\begin{array}{c|ccc} & 1 & 2 & 3 \\ \hline 1 & 92 & 8 & 0 \\ 2 & 3 & 87 & 10 \\ 3 & 0 & 10 & 90 \end{array}\right)$$

$$\mathbf{C}_{test}^{MEP} = \left(\begin{array}{c|ccc} & 1 & 2 & 3 \\ \hline 1 & 95 & 5 & 0 \\ 2 & 3 & 90 & 7 \\ 3 & 0 & 11 & 89 \end{array}\right)$$

Table 3.8 compares the values of $Cost_E$ and $Cost_W$ for OptP, UP and MEP on

the test set. It is interesting to notice that both $Cost_E$ and $Cost_W$ have reduced compared to the classical partitioning schemes. Also observing the confusion matrices, it is evident that confusion between different classes are the least for optimal partitioning as compared to other two partitioning methods.

Table 3.8. Comparison of Classification Performances of Different Partitioning Schemes on Test Data Set

Partitioning	$Cost_E$	$Cost_W$
OptP	0.1222	0.1866
UP	0.1489	0.2000
MEP	0.1357	0.2267

6.6 Crack Propagation

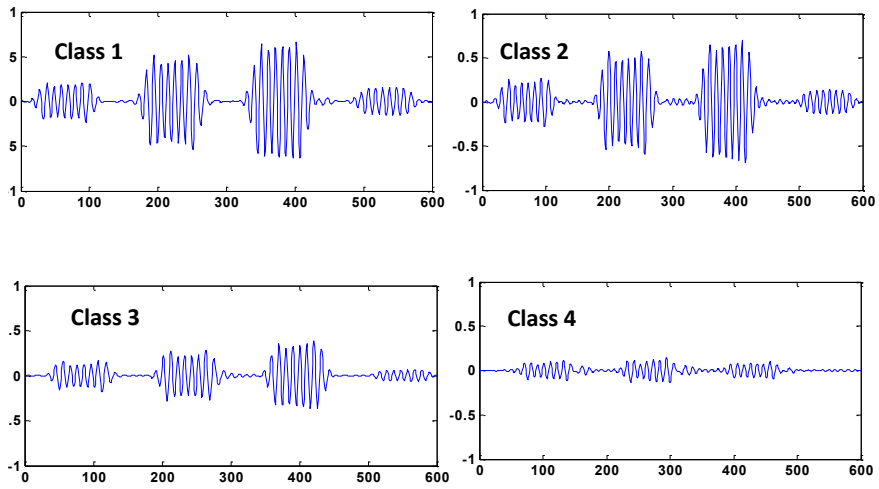


Figure 3.19. Representative signal from different classes in crack propagation phase

As discussed earlier, damage is defined by crack length during the crack propagation phase. Training and test data sets are classified based on the crack length as all the ultrasonic signals are labeled with the corresponding crack lengths. This regime has been partitioned into four classes as defined in section 6.2. Fig. 3.19 shows the representative test data sample for each class. The weighting matrix W

is defined as per rules described in section 6.5. Here, W matrix is chosen as:

$$\mathbf{W} = \begin{pmatrix} 0 & 1 & 2 & 3 \\ 3 & 0 & 1 & 2 \\ 6 & 3 & 0 & 1 \\ 9 & 6 & 3 & 0 \end{pmatrix}$$

Sequential optimization of partitioning has been carried on the training data set to find the optimal partitioning. The optimal alphabet size is 6 with stopping threshold value $\eta_{stop} = 0.005$. The confusion matrices for Optimal, Uniform and Maximum Entropy Partitioning on the test data set are given by $\mathbf{C}_{test}^{OptP}$ $\mathbf{C}_{test}^{UP}$ and $\mathbf{C}_{test}^{MEP}$ respectively. Table 4.4 shows the comparison of classification performances using different partitioning processes.

Table 3.9. Comparison of Classification Performances of Different Partitioning Schemes on Test Data Set

Partitioning	$Cost_E$	$Cost_W$
OptP	0.255	0.5
UP	0.40333	0.68
MEP	0.31333	0.52

The confusion matrices obtained from the three different partitioning schemes are:

$$\mathbf{C}_{test}^{OptP} = \left(\begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 93 & 7 & 0 & 0 \\ 2 & 5 & 89 & 6 & 0 \\ 3 & 0 & 4 & 92 & 4 \\ 4 & 0 & 3 & 7 & 90 \end{array}\right)$$

$$\mathbf{C}_{test}^{UP} = \left(\begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 94 & 5 & 1 & 0 \\ 2 & 15 & 74 & 11 & 0 \\ 3 & 0 & 9 & 85 & 6 \\ 4 & 1 & 5 & 11 & 83 \end{array}\right)$$

$$\mathbf{C}_{test}^{MEP} = \left(\begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 93 & 7 & 0 & 0 \\ 2 & 12 & 82 & 6 & 0 \\ 3 & 0 & 7 & 89 & 4 \\ 4 & 1 & 3 & 7 & 89 \end{array}\right)$$

It is observed that both costs $Cost_E$ and $Cost_W$ have reduced for optimal partitioning as compared to maximum entropy and uniform partitioning. Also, from confusion matrices it is evident that optimal partitioning has improved the classification results. By close observation of the confusion matrices, it is seen that the chances of higher level damage being classified as a lower level damage is reduced by the current choice of weighting matrix.

Thus, the proposed method of fatigue damage classification has been validated to yield low classification error rates in presence of sensor noise and variability in the material properties that have a significant bearing on the ultrasonic impedance. The advantages of the proposed fatigue detection scheme include real-time implementation where the classification information can be readily used to make maintenance & control decisions, which is critical for the safe operation of machines or structural components.

7 Summary, Conclusions and Future work

This chapter presents a nonlinear symbolic technique for feature extraction from time-series of observed sensor data. The feature extraction algorithm is formulated to enhance the classification rate, where the time series data is optimally partitioned in the symbolic dynamic filtering (SDF) framework. A trade-off between sensitivity and robustness of classification is achieved through the construction of the cost functionals. It is demonstrated that the classification rate can be improved beyond what is achieved using the conventional partitioning techniques (e,g., maximum entropy partitioning and uniform partitioning). The optimization procedure can be also used to evaluate the capability of other partitioning schemes towards achieving a particular objective.

However, it should be noted, that the success of the partitioning optimization

process relies on the very nature of the slowly varying non-stationary characteristics of the sensor observation due to an evolving fault in the underlying dynamical system and the degradation of the sensor itself. For example, in the engine fault detection example, if the evolution characteristics of the plant and the sensor itself have very similar signature, then these classes of faults could be almost indistinguishable in the data space, potentially making the detection tool ineffective in the presence of sensor degradation. However, such an ill-posed problem could be made a well-posed one if the data set is appropriately transformed into another domain. Here is a simple pathological example to explain this remark. Suppose, there is a bias fault in a particular sensor observation and, at the same time due to an actual system failure, the data show a mean shift and a change in frequency. Now, if the extent of the bias in the sensor and mean-shift due to plant fault are of similar values, then it will be extremely difficult to distinguish them in the time domain using SDF. However, since there would be a change in the data frequency spectrum due to the plant fault, these two types of data evolutions could be distinguished if the data set is analyzed in the frequency domain. Thus, identifying suitable data pre-processing methods from the training data set is an important aspect, that should be investigated in future. Apart from this issue, few other research directions are:

- Tuning the classifier within the optimization loop as described in the general framework in Fig. 3.2
- Development of an adaptive feature extraction framework for optimal partitioning under different environmental and signal conditions

Chapter 4

Hierarchical Semantic Sensor Fusion

Data-driven analysis of a complex system requires interpretation of multi-sensor information to assure enhanced performance; the objective here is to develop a feature-level semantic sensor information fusion technique. The hierarchical approach, adopted here, involves construction of composite patterns consisting of: (i) atomic patterns extracted from single sensor data, and (ii) relational patterns that represent cross-dependence among different sets of sensor data. The underlying theories are presented along with necessary assumptions and the proposed method is validated on the NASA C-MAPSS simulation model of aircraft gas turbine engines for fault diagnosis [88]. The two experimental case studies involve both single and multiple fault scenarios.

1 Motivation

Patterns generated from the time series of a single sensor may not carry sufficient information to characterize a complex system. Moreover, different characteristics may generate similar signatures in the observation from a single sensor. Hence, a data-driven modeling tool should have the capability to characterize, quantify and interpret multiple sensor outputs. However, sensor information fusion for a complex system is extremely challenging. First of all, a complex system typically has sensors with a large variation in modality (e.g., pressure, temperature, speed, and acceleration). Hence, a data-level information fusion is extremely difficult due to the inherent scaling problem of sensor data of different modality. On the other

hand, decision level fusion generally requires an in depth understanding of the physical system and its signatures in different sensor observations. In literature, Dempster-Shafer evidence theory for fusion has been applied for engine fault diagnosis [89]; similarly, the concept of multi-sensor based Bayesian Belief Networks has also been used for fault diagnosis in turbofan engines [90]. Both techniques belong to the class of decision fusion at an upper level. One of the main aspects of SDF-based method is semantic representation of sensor data, irrespective of modality and other sensor specific characteristics. This would facilitates feature-level fusion of non-homogeneous sensors. They are derived based on a *semantic framework* for pattern extraction and classification [88]. To handle a large volume of data in real time, a hierarchical framework for information fusion is proposed that progressively leads from machine representations of observed data to fault classification. One of novelties of the proposed approach is identifying the cross-dependence among different sensor observations to reduce loss of information.

The chapter is organized in five sections including the present one. Section 2 explains the Hierarchical framework of multi-sensor data interpretation and fusion in detail. Case studies validating the proposed method on the C-MAPSS test bed are discussed in Section 3. Finally, the chapter is summarized and concluded in Section 5 with recommendations of future work.

2 Semantic Framework for Multi-sensor Data Interpretation and Fusion

A hierarchical (three-layered) semantic framework is proposed in this work for the purpose of multi-sensor data interpretation and fusion. The basic structure of this architecture is inspired by the information fusion model proposed by the Data Fusion Information Group (DGIF) [91]. The lowest level of this hierarchy deals with signal conditioning, transformation and finally feature extraction for unimodal sensor data streams. In the present framework, patterns discovered from individual sensors are called atomic patterns and SDF is used to extract them.

Let $\mathbb{L} = \{\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_N\}$ be the universal set of *atomic* patterns. The *atomic* pattern library $\mathbb{L}$ is set of modal footprints identified from individual sensing

modalities for various fault classes. Given the atomic pattern library, a popular framework for addressing information fusion is what is called the *set-theoretic* approach. In this framework, higher level patterns or contexts are modeled as subsets of $\mathbb{L}$. Thus a composite pattern, resulting from fusion of atomic patterns, is a collection of elements from $\mathbb{L}$ and the composite pattern library $\mathbb{L}^* \subset 2^{\mathbb{L}}$. The disadvantage of this approach is that it considers only modal footprints for constructing composite patterns as a *bag of atomic patterns*; relational dependencies, if any, between patterns are disregarded. However, the relational dependencies should not be ignored for many problems in practice, e.g., in the present validation problem of fault diagnosis of simultaneously degrading electro-mechanically connected aircraft engine components. Therefore, a hierarchical semantic framework for multi-sensor data interpretation and fusion is proposed that involves a common approach to information fusion going from one level to another and to include relational dependencies for composite pattern representation. Thus, the middle layer deals with the relational dependencies among atomic patterns, where relationships are modeled as the cross-dependencies among sensor data streams from different sensors. The cross-dependencies are discovered via relational PFSA that essentially capture the dynamics of state transition in one symbol sequence (obtained from one sensor) corresponding to a symbol appearance in the second symbol sequence (obtained from another sensor). Loose time-synchronization between sensor observations will be enough for this purpose. Symbol-level cross-dependencies between modalities are exploited to reduce information loss.

Finally, the top layer consists of higher level composite patterns that will be represented as digraphs where the atomic patterns (AP) are modeled as nodes and dependencies between nodes are modeled as relational patterns (RP). A formal definition is as follows:

Definition 2.1 (Composite pattern representation)**.** *Let* $\mathbb{L} = \{\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_N\}$ *be the atomic pattern library. Let* $\mathbb{L}^* \subset 2^{\mathbb{L}}$ *be the set of allowable primitives for a class. Then a composite pattern library* $\mathbb{H}^r = \{\mathcal{H}_1^r, \mathcal{H}_2^r, \ldots, \mathcal{H}_M^r\}$ *where a composite pattern* $\mathcal{H}_i^r$ *is digraph* $\mathcal{H}_i^r = (\mathcal{L}_{V_i}, \mathcal{E}_{V_i})$*;* $\mathcal{L}_{V_i} \subset \mathbb{L}$ *with the index set* $V_i \subset \{1, 2, \ldots, N\}$ *and* $\mathcal{E}_i = \{\mathcal{R}_{jk} | j, k \in V_i \times V_i\}$ *is a set of relational PFSAs. The digraph representation is illustrated in Figure 4.1.*

The relational probabilistic finite state automata (PFSA) are discovered using

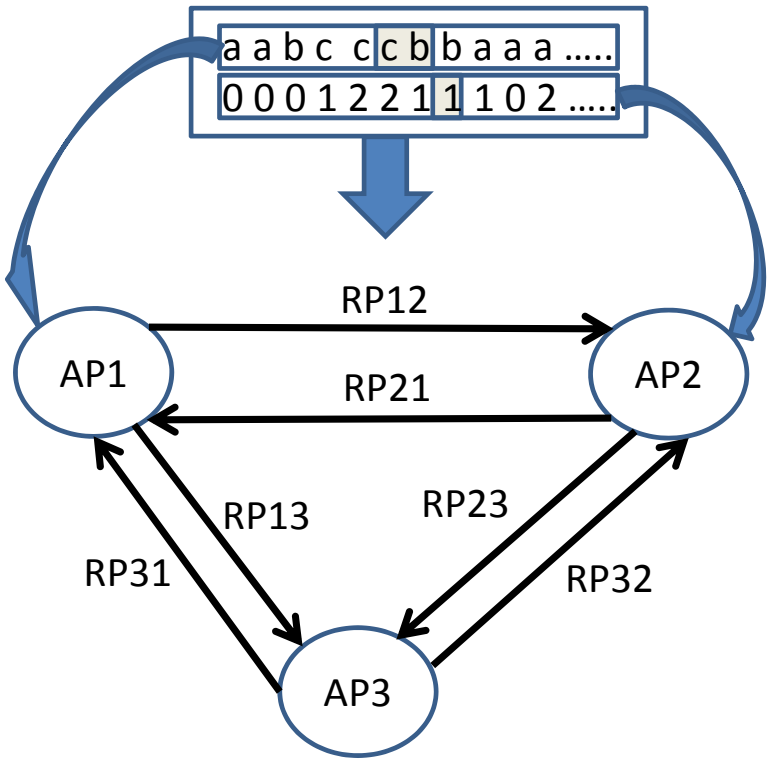


Figure 4.1. Composite Pattern Digraph

xD-Markov machine construction to determine cross-dependence; the algorithm is described in Section 2.1.

2.1 Construction of Relational PFSA: xD-Markov machine

In the SDF framework, a probabilistic finite state automata ($PFSA$), is constructed and a symbol sequence $\{\mathbf{s}\}$ generated via partitioning is run through the $PFSA$. As described earlier, the $PFSA$ considered in this framework is known as D-Markov machine [17]. A formal definition is as follows:

Definition 2.2 (D-Markov)**.** *A D-Markov machine with depth D is defined as a 4-tuple $\mathcal{M} \triangleq (\mathcal{Q}, \Sigma, \delta, \tilde{\Pi})$ such that:*

- $\Sigma = \{\sigma_0, ..., \sigma_{|\Sigma|-1}\}$ *is the alphabet set of symbol sequence* $\{\boldsymbol{s}\}$
- $\mathcal{Q} = \{q_1, q_2, \ldots, q_{|\Sigma|^D}\}$ *is the state set corresponding to symbol sequence* $\{\boldsymbol{s}\}$*. States represent all possible words of length D, using the symbol alphabet*

- $\delta : \mathcal{Q} \times \Sigma \rightarrow \mathcal{Q}$ *is the state transition mapping that maps the transition in a symbol sequence from one state to another upon arrival of a symbol*
- $\tilde{\Pi}$ *is the symbol generation matrix of size* $|\mathcal{Q}| \times |\Sigma|$*; an element* $\tilde{\Pi}_{ij}$ *denotes the probability of finding* j^{th} *symbol from* i^{th} *state in the symbol sequence* $\{\boldsymbol{s}\}$

To reduce dimensionality, the corresponding stationary state probability vector can be considered as the extracted feature vector. However, $\tilde{\Pi}$ can also be reshaped into a vector of length $|\mathcal{Q}| \times |\Sigma|$ and is treated as the extracted feature vector that is still a low-dimensional representation of the dynamical system at the slow-scale epoch. Let this feature vector obtained from a single symbol sequence (sensor data) be called an Atomic Pattern (AP).

Similarly, xD-Markov machines can be constructed from two symbol sequences $\{\mathbf{s}_1\}$ and $\{\mathbf{s}_2\}$ obtained from two different sensors (possibly of different modalities) to capture the symbol level cross-dependence. A formal definition is as follows:

Definition 2.3 (xD-Markov)**.** *Let* $\mathcal{M}_1$ *and* $\mathcal{M}_2$ *be the PFSAs corresponding to symbol streams* $\{\boldsymbol{s}_1\}$ *and* $\{\boldsymbol{s}_2\}$ *respectively. Then a xD-Markov machine is defined as a 5-tuple* $\mathcal{M}_{1\rightarrow 2} \triangleq (\mathcal{Q}_1, \Sigma_1, \Sigma_2, \delta_1, \tilde{\Pi}_{12})$ *such that:*

- $\Sigma_1 = \{\sigma_0, ..., \sigma_{|\Sigma_1|-1}\}$ *is the alphabet set of symbol sequence* $\{\boldsymbol{s}_1\}$
- $\mathcal{Q}_1 = \{q_1, q_2, \ldots, q_{|\Sigma|_1^{D_1}}\}$ *is the state set corresponding to symbol sequence* $\{\boldsymbol{s}_1\}$*, where* D_1 *is the depth for* $\{\boldsymbol{s}_1\}$
- $\Sigma_2 = \{\sigma_0, ..., \sigma_{|\Sigma_2|-1}\}$ *is the alphabet set of symbol sequence* $\{\boldsymbol{s}_2\}$
- $\delta_1 : \mathcal{Q}_1 \times \Sigma_1 \rightarrow \mathcal{Q}_1$ *is the state transition mapping that maps the transition in symbol sequence* $\{\boldsymbol{s}_1\}$ *from one state to another upon arrival of a symbol in* $\{\boldsymbol{s}_1\}$
- $\tilde{\Pi}_{12}$ *is the symbol generation matrix of size* $|\mathcal{Q}_1| \times |\Sigma_2|$*; the* ij^{th} *element of* $\tilde{\Pi}_{12}$ *denotes the probability of finding* j^{th} *symbol in* $\{\boldsymbol{s}_2\}$ *while making a transition from* i^{th} *state in the symbol sequence* $\{\boldsymbol{s}_1\}$

In practice, $\tilde{\Pi}_{12}$ is reshaped into a vector of length $|\mathcal{Q}_1| \times |\Sigma_2|$ and is treated as the extracted feature vector that is a low-dimensional representation of the

relational dependence between $\{\mathbf{s}_1\}$ and $\{\mathbf{s}_2\}$. This feature vector is called a Relational Pattern (RP). Figure 4.1 schematically describes the basic concept of the xD-Markov machine. Note, a Relational Pattern between two symbol sequences is not necessarily symmetric; therefore, RPs need to be identified for both directions. Also, when both symbol sequences are same, the relational patterns are essentially the atomic pattern corresponding to the symbol sequence; i.e., xD-Markov machine reduces to a simple D-Markov machine.

The set-theoretic approach falls at one end of the spectrum of information fusion; here all relationships are excluded and any fusion is solely done in the decision-theoretic sense where the presence (or absence) of one or more footprints can be used to estimate the probability of the fault class under consideration. The other end of the spectrum is to fuse data at the lowest level and construct machines (PFSAs) working in the product space of all sensors. This approach would be able extract modal dependencies before they are lost when constructing separate machines for individual sensor or modalities. But working in the product space has the danger of state space explosion especially when the sensors and sensing modalities can be numerous, as in a case of a complex system equipped with high number of sensors. The proposed approach is a trade-off between the two ends of the spectrum and attempts to include relational dependencies between sensing modalities, while keeping it tractable for a practical application. A hierarchical approach ensures that composite patterns are identified only when it's constituting units at the lower level have been observed. In the current framework we have considered relations taken only two at a time, but we propose to explore relations between higher order cliques as future work.

3 Gas Turbine Engine Fault Diagnosis via Semantic Sensor Fusion

In this research, the dynamic gas-path model, mentioned earlier (and detailed in Appendix-A) has been used to serve as the (simulated) plant that generates sets of (simulated) multi-sensor time series data. The two experimental case studies involve both single and multiple fault scenarios. Furthermore, since aircraft engines

undergo natural degradation during the course of their normal operation, the issue of distinguishing between faults and natural degradation is also addressed.

3.1 Case-study I: Degradation vs. Faults in a Single Component

In the context of gas path health monitoring of aircraft engines, two types of changes in engine performance can be considered: (i) Natural deterioration, i.e., gradual degradation of engine components due to wear and usage and (ii) Rapid faults, i.e., abrupt deterioration of turbo machineries. The fault detection method described in this chapter identifies the health status of an engine component at a particular slow-scale epoch. Now, in a real life scenario, faults need to be distinguished from the usual gradual degradation of a component. However, to achieve this goal, the health status must be monitored over several slow-time epochs. Given a temporal profile of the health status identified by the current method, faults can be distinguished from the usual degradation. Typically, the engine health deteriorates at a slow rate for usual degradation while a fault is the cause of an abrupt change in health status or degradation at a comparatively faster rate. However, gas path health management systems need to be designed to perform both functions: 1) estimation and trend monitoring of all engine health parameters over the lifetime of the engine; and 2) detection of rapid/abrupt performance changes and isolate the root cause. In this section, representative experimentations are performed in order to validate the capability of the current methodology of distinguishing normal degradation from faults for a given usage information. Moreover, the performance of the entire (typical) sensor suite will be examined over the spectrum of faults in the engine components and actuators. Such investigation will potentially lead to construction of an optimal sensor suite and a hierarchical procedure of fault detection and isolation.

In the case study, the engines under consideration are assumed to have undergone 1000 flight cycles. Therefore, components of a nominal engine of this type may degrade to some extent (and not remain ideal). A stochastic damage model has been developed and incorporated in the new C-MAPSS *Transient Test-case Generator* [92] (developed by NASA), based on the experimental data for trending

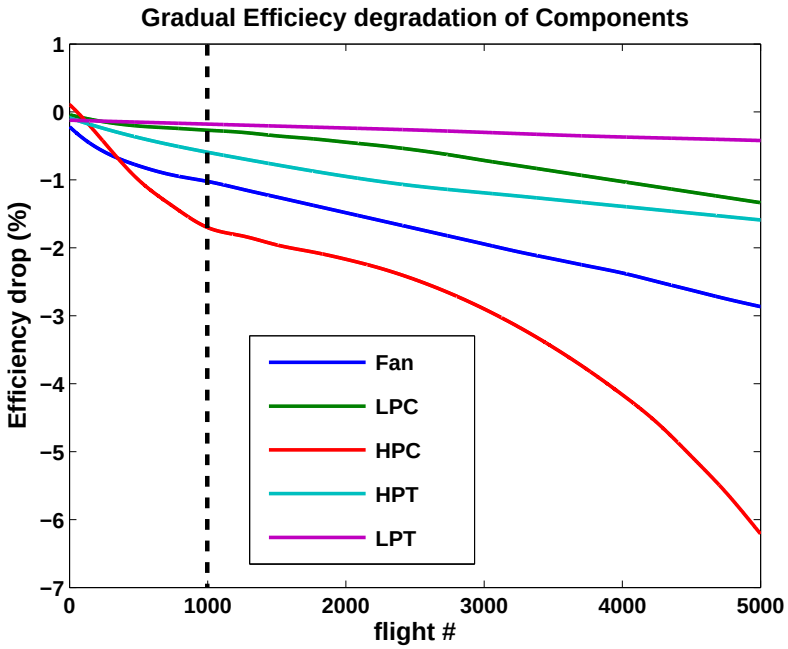


Figure 4.2. Natural Efficiency Degradation Profiles for Rotating Components

the natural deterioration of the engine components. Figure 4.2 is a typical outcome of the model that shows the natural degradation profiles of efficiency health parameters for the five rotating components of the engine (similar profiles can be generated for the flow health parameters). An engine after 1000 cycles of flight is considered nominal if the changes in health parameters are within the expected worst-case degradation limits (after 1000 cycles) calculated by the model. Hence, for the nominal data samples of engine operation, health parameters of the rotating components are considered within the prescribed ranges in a uniformly random fashion. Three actuators, Variable Stator Vane (VSV), Variable Bleed Valve (VBV) and Fuel Flow Rate (Wf), are assumed to be in ideal health conditions for a nominally operating engine after 1000 cycles.

The method of injecting rapid/abrupt faults is summarized below (Note, this fault injection method has been used in the transient test-case generator code, it is explained in this chapter for the sake of completeness). For all five rotating components, faults exhibit random magnitudes (F_m), and a random health parameter ratio (HPR). While F_m and HPR directly determines the change in efficiency health parameter $\psi(C)$ of a component C, the change in flow health parameter

ζ_C is determined by HPR for a given change in ψ_C. Formally, the following two relations are used.

$$\delta_{\psi_C} = -\frac{F_m}{\sqrt{1+HPR^2}} \tag{4.1}$$

$$\delta_{\zeta_C} = \delta_{\psi_C} \cdot HPR \tag{4.2}$$

where, δ_{ψ_C} and δ_{ζ_C} denote the changes in ψ_C and ζ_C respectively.

In the case study, fault magnitude (F_m) follows a random uniform distribution ranging from 1 to 7. Health parameter ratios (HPR) for Fan, LPC, and HPC are uniformly distributed between 1.0 and 2.0, whereas HPR s for HPT and LPT are uniformly distributed between 0.5 to 1.0. The changes in health parameters occur from certain base values of ψ_C and ζ_C. For the present example, base values are taken as the health parameters after maximum possible natural degradation after 1000 flight cycles. Injecting actuator faults is rather straightforward; only faults due to scale shift are considered in this example. The ranges for random uniformly distributed scale shift magnitudes ($S_m(A)$ for actuator A) are: (i) 1% to 7% for $S_m(VSV)$, (ii) 1% to 19% for $S_m(VBV)$ and (iii) 1% to 7% for $S_m(Wf)$.

There are 9 health condition classes considered for this study. Apart from the nominal class, 8 fault classes signify (single) faults in 5 rotating components and 3 actuators. Hundred samples are generated (using the logic described above) for each of these 9 classes, among which 50 are taken as training samples and 50 are taken as test samples. The experiments are performed with the same TRA, Mach Number and Altitude inputs as used in the previous chapter. The entire commercially available sensor suite of 6 sensors (N_c, N_f, P_{50}, Ps_{30}, T_{24} and T_{48}) is used for fault detection.

3.2 Validation Results for Case-study I

Pertinent fault detection results for the problem described above is presented in this subsection. For data partitioning, maximum entropy partitioning is used with alphabet size, $|\Sigma| = 5$ for all six sensors (although alphabet size does not need to be same for different sensors). The depth for constructing $PFSA$ states is taken to be, $D = 1$ for both atomic and relational pattern construction and features are classified by a k-NN classifier (with $k = 5$) using the Euclidean distance metric.

Table 4.1 provides the classification errors corresponding to all atomic and relational patterns. The cross-dependence direction is from Sensor 1 to Sensor 2 in the table. Hence, the diagonal elements represent the classification error percentages corresponding to the atomic patterns, where as the off-diagonal elements represent the classification error percentages corresponding to the relational patterns.

Table 4.1. Comparison of Classification Error Percentages using Atomic and Relational Patterns on Test Data Set (50×9 samples); Cross-dependence direction: $Sensor1 \rightarrow Sensor2$

		Sensor 2					
		Nc	Nf	P_{50}	Ps_{30}	T_{24}	T_{48}
Sensor 1	Nc	26.22	17.78	16.89	7.11	2.89	5.78
	Nf	19.33	43.56	49.78	12.44	21.11	14.44
	P_{50}	12.00	50.22	52.00	34.22	28.44	29.78
	Ps_{30}	7.56	10.89	32.22	23.78	16.00	15.33
	T_{24}	4.22	17.78	29.78	14.22	43.78	26.44
	T_{48}	8.00	15.78	33.78	5.11	18.67	34.44

It is observed from the above table that the relational patterns are able to extract useful information from the perspective of fault diagnosis, which is physically meaningful due to the strong electro-mechanical interconnections among the rotating components and the actuators. Therefore, ignoring these cross-dependencies should affect the fault detection results. Finally all the patterns are concatenated to construct the overall composite pattern. The classification error on the test data set using the composite pattern is found to be 2% and the corresponding confusion matrix is given below. In a confusion matrix C, element C_{ij} denotes the frequency of classifying test sample from class i as a sample from class j. This exercise may be enough while treating the engine as a black-box and using all sensors blindly for fault detection. However, deeper questions regarding relationships among fault locations and sensor locations or the optimal sensor-suite still remain. These two issues are briefly discussed in the sequel.

$$\mathbf{C}_{test}^{\text{All}} = \left(\begin{array}{c|ccccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ \hline 1 & 50 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 48 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 3 & 0 & 0 & 50 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 50 & 0 & 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 50 & 0 & 0 & 0 & 0 \\ 6 & 0 & 0 & 0 & 0 & 0 & 50 & 0 & 0 & 0 \\ 7 & 0 & 0 & 0 & 0 & 0 & 0 & 50 & 0 & 0 \\ 8 & 0 & 0 & 0 & 0 & 0 & 0 & 6 & 44 & 0 \\ 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 49 \end{array}\right)$$

Building a Hierarchical Decision Engine

From the physical understanding of the engine model, it is clear that there may be variation in sensitivity of different sensors to different fault conditions (locations). For example, detection accuracy (false alarm) for faults in components mounted on the core shaft may be higher (lower) while using the Nc sensor. However, from the previous observations, it is evident that just using Nc may not be enough. This issue will be more clear with the following two concrete examples. Figures 4.3 and 4.4 are two bar charts plotting the detection accuracies (in percentage) for HPT and VBV fault conditions respectively while using different atomic and relational patterns. In this case, (as it is seen in the figure) there are 36 different patterns (6 atomic and 30 relational) available for 6 sensors.

Figure 4.3 shows that the relational patterns involving Nc and T_{24} perform very well in detecting HPT faults. This can be explained from the physical understanding of the engine lay out. First of all, Nc monitors the speed of the core shaft on which HPT is mounted. On the other hand, T_{24} monitors the temperature at HPC inlet and HPC is mechanically connected to HPT as it is mounted on the core shaft as well. So naturally, relational patterns involving these two sensors should perform well in detecting HPT faults. In the second example, as shown by Figure 4.4, relational patterns involving Nf and T_{48} perform very well in detecting Variable Bleed Valve (VBV) faults. VBV is used to control the pressure of the gas flowing from LPC to HPC. It is a modulating valve and actuator assembly

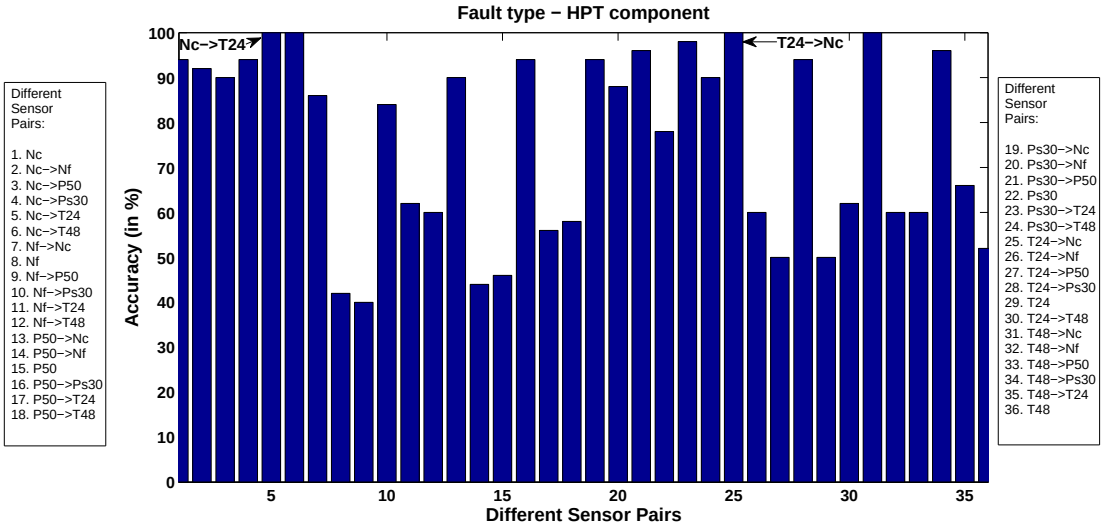


Figure 4.3. HPT Fault Detection Accuracy for Different Atomic and Relational Patterns

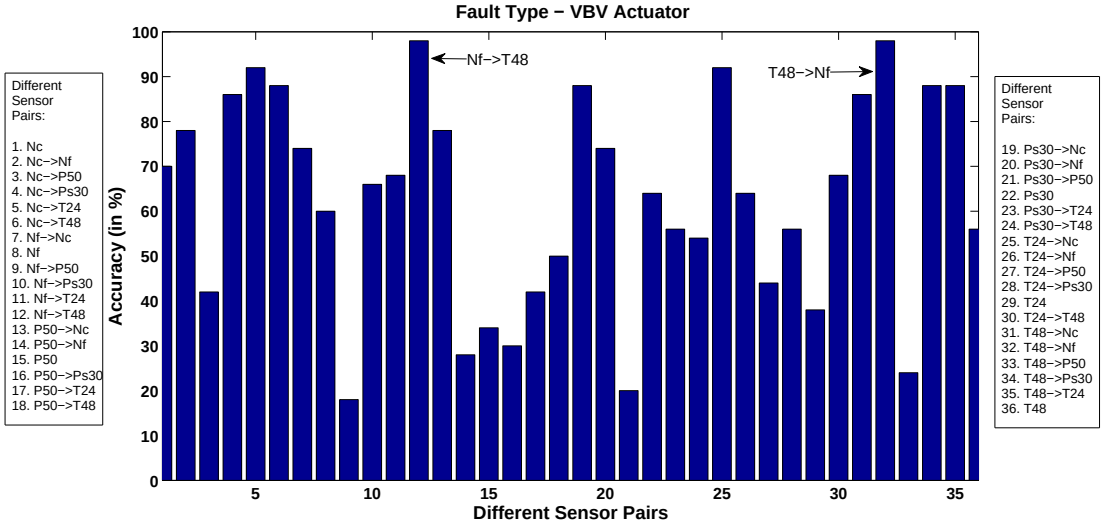


Figure 4.4. VBV Fault Detection Accuracy for Different Atomic and Relational Patterns

that bleeds off excess air to the atmosphere as necessary to prevent turbine surge. Therefore, it is natural that Nf, that monitors speed of fan shaft (LPC is mounted on the fan shaft) and T_{48}, that monitors HPT exit temperature (VBV essentially prevents turbine surge) perform very well in detecting VBV faults.

The above discussion leads to the idea of building a hierarchical decision engine.

In this framework, there will be a hierarchical strategy of isolating the fault location by going through decisions made by sensor sets that are much smaller in size compared to the full sensor suite. For example, (referring to the two examples above) if patterns from Nc and T_{24} determine that a fault has occurred in VBV, decision confidence can be enhanced based on the decision made by patterns from Nf and T_{48}. And if the decision regarding the fault location is supported by the corresponding most efficient sensor group, then the user does not need to go for investigating remaining patterns. Thus, this procedure can handle data from large sensor suites. However, another interesting question remains: Is there an optimal sensor suite (probably smaller than the full sensor suite) that performs well enough in every fault conditions? This issue is discussed in the sequel.

Optimal Sensor Suite

Building a hierarchical decision engine may be very useful when one has considerable physical understanding of the system and the interconnection characteristics among its sub-systems. This approach can also handle the issue of scalability. However, given a large volume of training data, it is possible to identify an optimal sensor suite that performs well enough for all fault conditions. Although, this may require handling large dimensional composite patterns all the time, it does not require much insight regarding the physics of the system. Here is an example optimization procedure for the current problem.

Let the detection accuracy (in fraction, i.e., varies from 0 to 1) of pattern P for a fault class i be denoted as $D_i(P)$ and the minimum allowable accuracy be denoted as D_T. With these notations, a pattern will be called an optimal pattern P^*, if it satisfies the following condition:

$$\min_i D_i(P) > D_T \tag{4.3}$$

This is essentially placing a bound on the worst performance of a pattern. Therefore, the set of optimal patterns $\{P^*\}$ will increase with decrease in the value of D_T. An alternate cost function may be the average performance of a pattern. In the present study, considering $D_T = 0.85$, one can obtain that three relational patterns are optimal, namely Nc to T_{24}, T_{24} to Nc and Nc to Ps_{30}. The following confusion matrix is obtained by using the three sensors involved in these relational

patterns. Note that to maintain the structure of a composite pattern, all three (3) atomic and six (6) relational patterns that are generated by these three sensors have been used.

$$\mathbf{C}_{test}^{\text{Opt}} = \left(\begin{array}{c|ccccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ \hline 1 & 50 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 48 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 3 & 0 & 0 & 50 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 50 & 0 & 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 50 & 0 & 0 & 0 & 0 \\ 6 & 0 & 0 & 0 & 0 & 0 & 50 & 0 & 0 & 0 \\ 7 & 0 & 0 & 0 & 0 & 0 & 0 & 49 & 1 & 0 \\ 8 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 48 & 0 \\ 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 49 \end{array}\right)$$

The overall classification error is reduced from 2% to 1.33%, which is the consequence of leaving out quite a few non-optimal patterns that may be useful for the detecting some of the faults. This is somewhat counterintuitive from an information theoretic point of view as more patterns should provide more information. However, in the present classification setting, non-optimal patterns may incorporate more ambiguities in decision making.

3.3 Case-study II: Simultaneously Occurring Multiple Faults

Component-level fault diagnosis in an aircraft gas turbine engine involves identification of the fault type, and location & quantification of the fault level. In the C-MAPSS test-bed setting, the physical fault scenarios (e.g., fouling, increased tip clearance, and seal wear) are assumed to manifest themselves in affecting the efficiency and flow of the associated engine component(s). In the present case study, a simultaneous fault scenario has been considered involving two major rotating components of the engine, namely, HPC and HPT. Choice of these components has important significance from the perspective of diagnosis of simultaneously occurring faults. As it can be seen in Figure A.1, both HPC and HPT are mounted on the core shaft of the engine. Hence, they have strong mechanical interconnec-

tion between them. Besides, they also have electrical interconnection via control loop. Such strong electro-mechanical interconnection increases difficulty of the fault diagnosis problem. On the other hand, information from an HPC sensor will possibly have strong cross-dependency with an HPT sensor and it may not be reasonable to ignore that for the purpose of fault diagnosis. In the present study, three heterogeneous, non-collocated and commonly used sensors are selected that are placed in the HPC-HPT subsystem as seen in Figure A.2. The three sensors are as follows:

Table 4.2. Sensors for Fault Diagnosis in the HPC-HPT subsystem

Sensors	**Description**	**Noise Std. (%)**
Ps_{30}	HPC exit static pressure	0.50
T_{48}	HPT exit temperature	0.75
N_c	Core spool speed	0.25

Sensor noise standard deviation values are provided as percent of operating point trim values [85].

Remark 3.1. *The three chosen sensors are of different modalities (pressure, temperature, speed) and while the pressure sensor is located at HPC exit, the temperature sensor is located at HPT exit and the speed sensor is measuring the rotational speed of the core shaft on which both HPC and HPT are mounted. The challenge here is to identify the relational dependencies among these sensor data to enhance the fault diagnosis performance.*

Diagnosis involves both fault localization and fault level identification. The health parameters that defines the health status of HPC and HPT, are the efficiency and flow health parameters (ψ_{HPC}, ζ_{HPC} and ψ_{HPT}, ζ_{HPT}). Three fault levels are considered similarly for both HPC and HPT. Table 4.3 shows the approximate ranges of efficiency health parameters under different fault levels. Corresponding flow health parameters are chosen using the same logic used in Section 3.1. Here, the low fault level indicates very minimal loss in efficiency and flow and hence also includes the absolute nominal health condition ($\psi_{HPC} = \zeta_{HPC} = 1$ or $\psi_{HPT} = \zeta_{HPT} = 1$). In the context of the case-study I, the low fault level can be considered as the nominal class as well (i.e., low level faults do not raise any alarm).

In this study, classes are defined as Cartesian products of the ranges of HPC and HPT health parameters. Thus, there are 9 (i.e., 3×3) classes of data that can

Table 4.3. Fault Levels in HPC/HPT

Fault Level	Efficiency Range
Low Fault	1.0000 to 0.9867
Medium Fault	0.9867 to 0.9733
High Fault	0.9733 to 0.9600

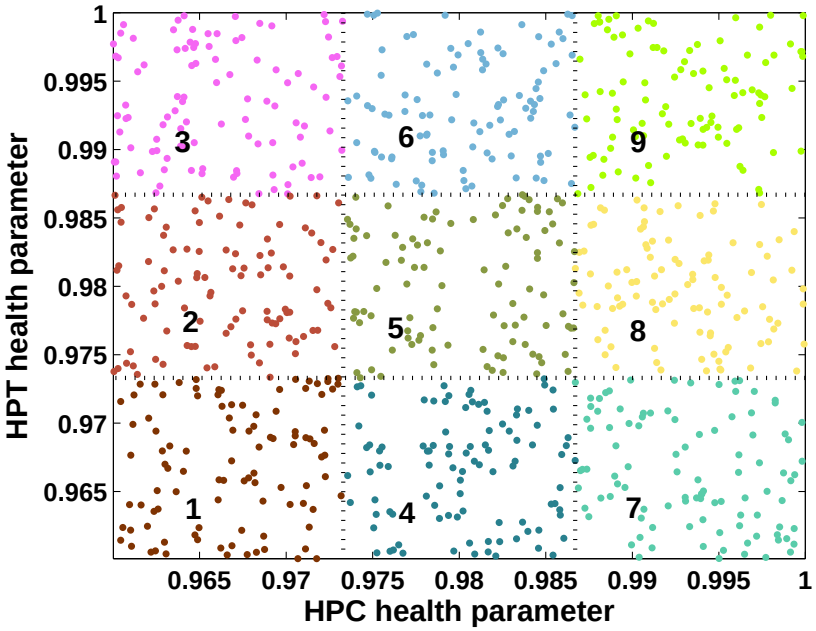
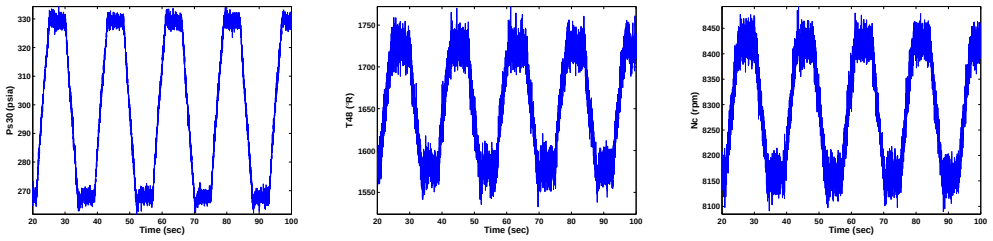


Figure 4.5. Fault classes for data collection and classification

be obtained when a class is uniquely defined by an HPC fault level (a range of HPC health parameters) and an HPT fault level (a range of HPT health parameters). Hundred simulation runs of the engine system have been conducted for each class to generate data set for analysis among which 50 samples are chosen as the training set and the remaining 50 samples are kept as testing set. HPC/HPT health parameters are chosen randomly from independent uniform distributions for health parameters within the prescribed ranges given in above table. Figure 4.5 plots the samples generated using the above logic in the two dimensional parameter space. Different classes of samples are shown in different colors and as well as marked with the class numbers in the figure.

For each data sample, a time series was collected for all three sensors (given in Table A.1) under persistent excitation of TRA. The experiments are performed



(a) Ps_{30} sensor observation (b) T_{48} sensor observation (c) Nc sensor observation

Figure 4.6. Representative time series observations from different Sensors

with the same TRA, Mach Number and Altitude inputs as used in the previous chapter. For each experiment, the engine simulation is conducted at a frequency of 66.67 Hz (i.e., inter-sample time of $15ms$) and the length of the simulation time window is 150 seconds, which generate $10,000$ data points. Figure 4.6 shows representative examples of time series data from each of the three sensors. Thus, in the above context, the problem of multi-component fault diagnosis in aircraft gas turbine engines is formulated as a multi-class classification problem (in the present scenario, number of classes is 9).

3.4 Validation Results for Case-study II

This section presents pertinent fault diagnosis results for the Case-study II. The signal processing specifications remains same as in Case Study I. For a particular health parametric condition, three atomic patterns are generated from three sensor observations and six relational patterns are generated by extracting pairwise directed cross-dependencies. Finally all the patterns are concatenated to construct the overall composite pattern. The classification error on the test data set using the composite pattern is found to be 11.56%. Similar to Table 4.1, Table 4.4 provides the classification errors corresponding to all atomic and relational patterns.

It is observed from the above table that the relational patterns are able to extract useful information from the perspective of fault diagnosis. The problem in the present study has been posed in such a way that the sensor information from different sensors actually have cross-dependencies due to strong electro-mechanical interconnections between HPC and HPT. Therefore, ignoring these cross-dependencies should affect the fault diagnosis results. This result once again confirms the con-

Table 4.4. Comparison of Classification Error Percentages using Atomic and Relational Patterns on Test Data Set (50×9 samples); Cross-dependence direction: $Sensor1 \rightarrow Sensor2$

		Sensor 2		
		Ps_{30}	T_{48}	Nc
Sensor 1	Ps_{30}	56.44	15.11	41.56
	T_{48}	12.89	56.89	16.00
	Nc	25.11	11.78	52.22

jecture and shows that the xD-Markov machine construction can extract those cross-dependencies. It should be noted that the fault diagnosis algorithm is completely data-driven and has no model information. Therefore, the result is significantly encouraging. Representative classification confusion matrices corresponding to atomic patterns from Sensors T_{48}, Nc and their relational patterns of both directions are provided below.

A close observation reveals similarity of fault signatures on single sensor data for two completely different fault conditions, e.g., fault signatures of Class 4 (HPC medium fault, HPT low fault) has been confused with Class 1 (HPC low fault, HPT low fault) for both sensors T_{48} and Nc individually. However, this ambiguity can be removed by using relational pattern directed from T_{48} to Nc.

$$\mathbf{C}_{test}^{T_{48}} = \left(\begin{array}{c|ccccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ \hline 1 & 47 & 1 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 18 & 2 & 22 & 3 & 0 & 5 & 0 & 0 \\ 3 & 0 & 1 & 12 & 0 & 15 & 5 & 11 & 6 & 0 \\ 4 & 11 & 11 & 0 & 21 & 5 & 0 & 2 & 0 & 0 \\ 5 & 0 & 6 & 15 & 2 & 15 & 2 & 5 & 5 & 0 \\ 6 & 0 & 0 & 4 & 0 & 0 & 21 & 0 & 16 & 9 \\ 7 & 0 & 5 & 11 & 3 & 17 & 2 & 11 & 1 & 0 \\ 8 & 0 & 0 & 9 & 0 & 3 & 19 & 2 & 13 & 4 \\ 9 & 0 & 0 & 0 & 0 & 0 & 11 & 0 & 3 & 36 \end{array}\right)$$

$$\mathbf{C}_{test}^{Nc} = \left(\begin{array}{c|ccccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ \hline 1 & 17 & 0 & 0 & 17 & 8 & 0 & 1 & 7 & 0 \\ 2 & 1 & 23 & 1 & 0 & 10 & 6 & 0 & 1 & 8 \\ 3 & 0 & 2 & 40 & 0 & 0 & 6 & 0 & 0 & 2 \\ 4 & 11 & 0 & 0 & 29 & 0 & 0 & 7 & 3 & 0 \\ 5 & 6 & 8 & 0 & 1 & 19 & 0 & 0 & 6 & 10 \\ 6 & 0 & 5 & 16 & 0 & 2 & 17 & 0 & 0 & 10 \\ 7 & 0 & 0 & 0 & 14 & 0 & 0 & 35 & 1 & 0 \\ 8 & 10 & 0 & 0 & 9 & 11 & 0 & 2 & 17 & 1 \\ 9 & 1 & 6 & 4 & 0 & 5 & 13 & 0 & 3 & 18 \end{array}\right)$$

$$\mathbf{C}_{test}^{T_{48}->Nc} = \left(\begin{array}{c|ccccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ \hline 1 & 47 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 46 & 1 & 0 & 3 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 43 & 0 & 0 & 7 & 0 & 0 & 0 \\ 4 & 1 & 0 & 0 & 44 & 4 & 0 & 1 & 0 & 0 \\ 5 & 0 & 3 & 1 & 0 & 40 & 1 & 0 & 5 & 0 \\ 6 & 0 & 0 & 2 & 0 & 0 & 39 & 0 & 0 & 9 \\ 7 & 0 & 0 & 0 & 4 & 1 & 0 & 43 & 2 & 0 \\ 8 & 0 & 0 & 0 & 0 & 7 & 2 & 1 & 36 & 4 \\ 9 & 0 & 0 & 0 & 0 & 0 & 5 & 0 & 5 & 40 \end{array}\right)$$

$$\mathbf{C}_{test}^{Nc->T_{48}} = \left(\begin{array}{c|ccccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ \hline 1 & 49 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 40 & 2 & 3 & 5 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 48 & 0 & 0 & 2 & 0 & 0 & 0 \\ 4 & 6 & 1 & 0 & 40 & 3 & 0 & 0 & 0 & 0 \\ 5 & 0 & 3 & 0 & 2 & 43 & 0 & 0 & 2 & 0 \\ 6 & 0 & 0 & 3 & 0 & 0 & 45 & 0 & 0 & 2 \\ 7 & 0 & 0 & 0 & 1 & 0 & 0 & 46 & 3 & 0 \\ 8 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 40 & 7 \\ 9 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 2 & 46 \end{array}\right)$$

4 Summary, Conclusions and Future Plans

This chapter dealt with the issue of feature level fusion of multiple sensor data, which is very important for data-driven modeling techniques. A multi-sensor data interpretation and fusion methodology is presented under the symbolic dynamic filtering (SDF)-based framework and has been used for data-driven detection of component level faults and actuator faults in aircraft gas turbine engines. In the proposed method, the abstract semantic representation of sensor data enables feature level fusion of heterogeneous and disparate sensors. It is also shown that identification of cross-dependencies among different sensors prevents loss of significant information compared to set-theoretic information fusion methods. The hierarchical architecture of this method reduces computational complexity, allowing real-time operability. The basic hierarchical architecture of this framework allows a tractable solution to the problem of dealing with a large number of sensors and checks dimensionality explosion as one moves up the knowledge base levels only when structure in the lower levels has been identified.

The following research areas should be pursued in future.

- Development of algorithms to extract relational dependencies among three or more symbol sequences.
- Exploration of other statistical analysis tools (e.g., Copula distribution [93]) as an alternative to extraction of relational dependencies.
- Identifying the effects of lack of synchronization among sensor observations on xD-Markov machine construction.
- Comparative evaluation of semantic information fusion framework with other information fusion techniques, e.g., Dempster-Shafer or Bayesian network approaches. These concepts may also be used to formalize the construction of composite patterns.
- Construction of *Hierarchical Decision Engine* to expedite the process of detection and reduce false alarm rates while using a large sensor suite.
- Identification of *Optimal Sensor Suite* (small number of strategically located sensors) which can serve the purpose of decision making.

Part - II
Distributed Decision & Control
of
Multi-agent Complex Systems

Chapter 5

Equilibrium Thermodynamics for Heterogeneous Packet Transmission in Communication Networks

This chapter deals with both modeling and control aspects of heterogeneous packet transmission in communication networks. A novel equilibrium thermodynamics formalism is presented here to characterize phase transition and size scaling in large-scale communication networks. In particular, the qualitative nature of phase transitions in communication network systems is characterized via defining network analogs of thermodynamic quantities (e.g., order parameter, temperature, and pressure). Phase transition phenomena are investigated for multiple intensive parameters, namely, the external packet load on the communication network system and the packet transmission probability for heterogeneous packet types. Network phase diagrams are constructed based on the network parameters. Such phase diagrams are useful for building control strategies for heterogeneous packet transmission in communication network systems. Basic control objectives are discussed along with closed form solutions for a special network system considered in this chapter.

1 Motivation

Concepts of Statistical Mechanics are used to characterize thermodynamic systems in terms of the relationship between micro and macro behaviors (see Appendix-B for details). From such a perspective, thermodynamic systems have significant similarity with complex networks. A common feature across all these multi-agent systems is that their global behavior emerges from local dynamics of the participating agents. This has triggered the interest of many researchers to investigate complex networks from the perspectives of Thermodynamics and Statistical Mechanics [44][94].; see [27] for a review of recent literature in this field. Complex networks have been shown to characterize the behavior and topological organization of many natural and engineered systems, such as those found in the disciplines of sociology [95], biology [96], finance [97], communication networks [98], and sensor networks [99]. Phase transition is a characteristic phenomenon of complex systems, consisting of interacting and interdependent dynamics, where a non-smooth change in the operating characteristics may take place with a relatively small variation of the system parameter(s). In this context, tools of Statistical Mechanics are useful for characterization of critical phenomena corresponding to dependence of the global behavior (e.g., connectivity, average rate of change of queue length, and average packet drop rate) of communication networks on their local parameters (e.g., communication radius, packet load, and transmission probability) [100][101][102][103].

A key task in the analysis of phase transitions is identification of the system behavior in the vicinity of a critical point, where a global parameter quantifies the presence of order in the underlying system. Usually, this *order parameter* [44] is negligibly small or zero in the disordered phase and may have non-zero values in the ordered phase. Thus, a phase transition is realized as a discontinuous (or possibly a continuous) change in the order parameter from zero to a non-zero value when an intensive system parameter (e.g., temperature T) is varied across its critical value. For example, in the well-known case of ferromagnetism, the order parameter is magnetization $M(T)$ that has a non-zero value leading to spontaneous magnetization at temperatures below the Curie point [44]. The nature of phase transition is often broadly classified into two types: (i) first-order phase transition, where

the order parameter changes discontinuously with the intensive parameter at the critical point and, (*ii*) Continuous (also called second or higher order) phase transition, where the order parameter varies continuously with the intensive parameter during the phase transition. The congestion phenomenon in communication networks is an example of a phase transition. Typically, the concept of bottleneck buffers (routers) is used to detect and control (i.e., mitigate or prevent) congestion in communication networks. The network structure could be reduced to a multi-source and single destination (or vice-versa) through usage of bottleneck buffers and thus mean-field models can be developed for such cases [104]. However, analysis of certain problems (e.g., distributed decision-making, perimeter surveillance by static sensor networks, statistical estimation of parameter distribution over the entire space) is apparently very difficult and often intractable. Moreover, in many applications, on-line adaptation of the packet routing sequence is required to maintain acceptable performance in the presence of uncertain external perturbations and channel fluctuations. As an example, sensor networks are often employed in environments with limited communications capability. In such applications, information travels between sensor nodes through a combination of relay nodes and other sensor nodes. For communication-constrained sensor networks, the information that is transmitted is usually event-driven (such as detection of intruders) and is thus intermittent in nature. For this reason, it is important to have a controllable routing procedure that takes advantage of the available network structure to avoid bottlenecks and other congestion issues.

From the above perspective, this chapter presents a Statistical Mechanics-inspired concept of modeling critical phenomena and formulation of finite-size scaling laws in communication networks that is more suitable in event-driven situations compared to the traditional time-driven modeling approaches. The underlying theories have been tested and validated on a simulation test bed that consists of a simplified network model consisting of a 2-D square-grid network. This is justified as the objective here is to obtain an unambiguous understanding of the critical behavior on a relatively simple network, which would help establishing a knowledge base for synthesis of decision & control laws in different classes (e.g., wired and wireless) of real-life communication networks. Network phase diagrams are constructed based on the network parameters. Such phase diagrams are useful

for building control strategies for heterogeneous packet transmission in communication network systems. Basic control objectives are discussed along with closed form solutions for a special network system considered in this chapter.

The chapter is organized in five sections including the present one. Section 2 describes the model of the network along with the architecture of the simulation test bed. Then, the qualitative nature of phase transition in the underlying system is characterized, and the effects of network size is analyzed. In these analyses, phase transitions are investigated based on a single intensive parameter, namely the external packet load of the communication network system. Section 3 investigates the effects of two intensive parameters on the network performance. Phase diagrams are constructed for these cases by defining network analogs of thermodynamic quantities, such as order parameter, temperature, and pressure. Based on such equilibrium phase diagrams, Section 4 presents a novel thermodynamic approach for controlling heterogeneous packet transmission; the basic philosophy of such a control approach along with a representative numerical experiment on the network architecture of the current study is presented. The chapter is summarized and concluded in Section 5 along with recommendations for future research.

2 Congestion in Communication Networks: A Phase Transition Phenomenon

The network test bed simulates a two-dimensional square grid [101] as shown in Fig. 5.1, where the nodes (routers) are placed at the grid points. For a square grid network with $N \times N$ nodes, there are $(4N - 4)$ boundary nodes (shown as squares in Fig. 5.1) and $(N^2 - 4N + 4)$ internal nodes (shown as circles in Fig. 5.1). Only boundary nodes are assumed to be the sources and/or the sinks for packet generation and destruction; internal nodes can only transmit the packets. Each node receives packets in a queue with finite buffer length from its neighboring nodes and packets are terminated after reaching their respective destinations. In each time unit, packets are created in the boundary nodes with a Poisson arrival rate λ. For simplicity, all packets are considered to be of unit length. Destination nodes are chosen randomly from the boundary nodes, including their source nodes.

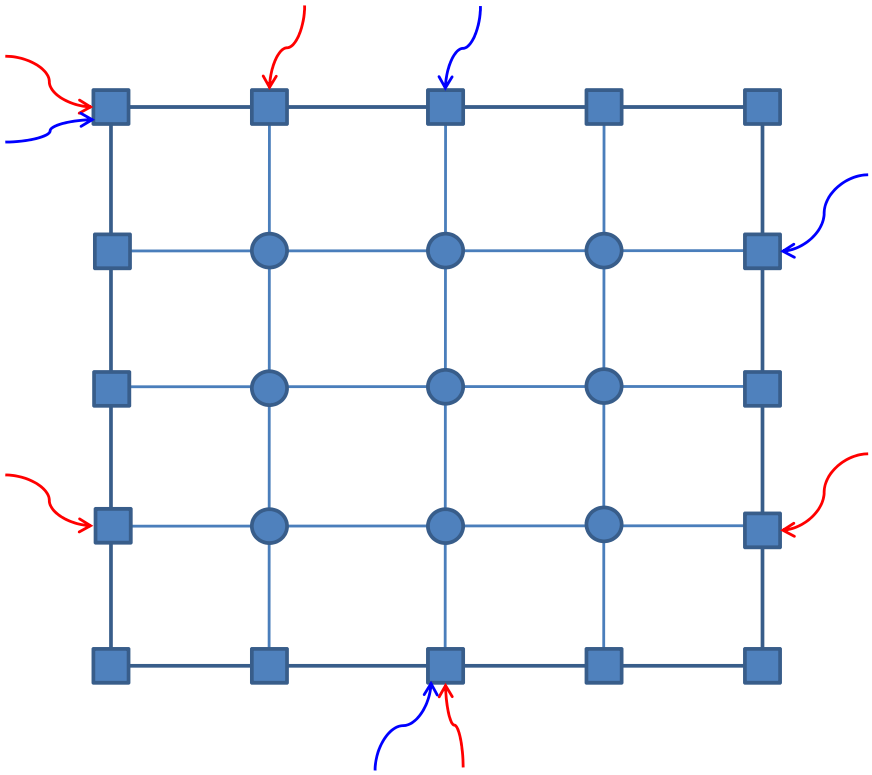


Figure 5.1. Network structure in the simulation model

Each node transmits one packet from the head of its queue to a deterministically chosen neighboring node at each time unit. The node chosen to forward a data packet is selected so that the packet travels via the shortest path to its destination. When there are more than one candidate nodes for the shortest path, the node with a smaller queue length is chosen to prevent early congestion of the network. A node drops the oldest packet from its fully occupied queue to accommodate a new packet. This notion is related to sensor-network operation where newer information carry more weight compared to older ones.

In communication networks, the phenomenon of most interest is known as congestion. Before congestion i.e., when the network is capable of handling the average number of packet arriving, the average packet drop rate remains negligibly small. However, as the packet influx rate crosses a critical threshold, the drop rate becomes nonzero. Thus, at a steady state, the non-zero average packet drop rate can serve as an index of degree of congestion. The phenomenon of congestion can be viewed as a continuous phase transition from a steady phase to an unsteady

phase of network communications, which is characterized in this chapter based on the concepts of equilibrium thermodynamics. Previous papers [100, 105] presented similar characterization based on average rate of change of queue length. However, those analyses required the assumption of infinite queue length, which is obviously impractical. This work removes that assumption and performs the analysis in a more realistic scenario.

As discussed earlier, a global order parameter of the system under consideration needs to be identified for investigating a phase transition from the perspectives of Statistical Mechanics. Based on the discussion above it is clear that packet drop rate is a feasible candidate for the order parameter. Furthermore, global intensive parameters that trigger the network phase transition need to be identified and, in this problem, packet influx rate λ can be one such parameter. As only $(4N-4)$ boundary nodes are generating packets and all N^2 nodes are sharing them, a *surface correction* is needed to accurately define the intensive parameter. To avoid introduction of the surface effect, an effective load per node, λ_{eff} is defined as follows:

$$\lambda_{eff} = \lambda \cdot \frac{4(N-1)}{N^2} \tag{5.1}$$

A few scaling adjustments are made to define the order parameter. Let $d(t)$ be the total number of packets dropped in the network at time t. Thus, average packet drop per node at time t is $d(t)/N^2$ and finally, let the time-expected value of average packet drop per node be denoted as $D = \langle \frac{d(t)}{N^2} \rangle_t$. The order parameter, M, is defined based on normalization of D with respect to the effective load per node, λ_{eff}.

$$M = \frac{D}{\lambda_{eff}} \tag{5.2}$$

Note that the order parameter M is the fraction of incoming packets that drop out from the network nodes per unit time. Therefore, in the absence of congestion, M could be assumed to be negligible, i.e., there is no packet drop. As the network becomes congested, the worst-case scenario is dropping of all incoming packets, i.e., M approaches 1.

Following the above procedure, M has been computed for given values of λ_{eff}. To eliminate the effects of transience, expected value of D is calculated using only steady-state time series data. The plot of M vs. λ_{eff} for the 10×10 simulated

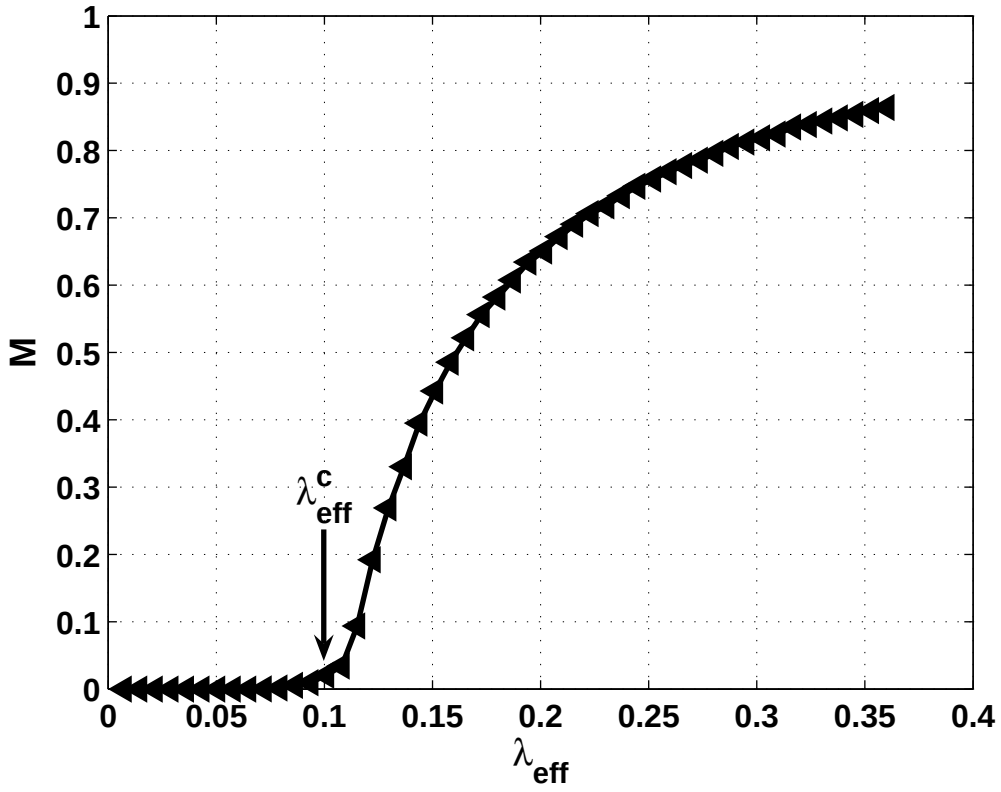


Figure 5.2. Continuous phase transition in a square-grid communication network

network is depicted in Fig. 5.2. It is seen that there is a critical value $\lambda_{eff}^{c} \approx 0.11$ of effective load per node such that, for $\lambda_{eff} < \lambda_{eff}^{c}$, the order parameter M is almost negligible; in contrast, for $\lambda_{eff} > \lambda_{eff}^{c}$, M takes on non-zero values. This abrupt change of network behavior across the critical value λ_{eff}^{c} is identified as a continuous *phase transition*, where the network moves from a *uncongested phase* (**U**) of negligible M to an *congested phase* (**C**) of finite positive M.

2.1 Construction of a Size Scaling Law

Unlike for natural thermodynamic systems, it is very important to understand the size scaling laws for human engineered multi-agent systems due to their inherent finite and smaller sizes compared to their natural counterpart. In general for finite sized systems, the critical value of the intensive parameter, say $T_c(N)$, is a function of size N, and as N goes to infinity, $T_c(N)$ converges to $T_c(\infty)$ of the

corresponding infinite sized system in the thermodynamic limit. In Statistical Mechanics literature, the space correlation length $\xi(t)$ behaves as a function of $|t|$ in a power law. Also the following size dependency form of $\xi(t)$ is assumed [106], which conforms to the nature of $\xi(t)$ function.

$$\xi(T_c(N) - T_c(\infty)) = aN \tag{5.3}$$

where a is a constant. Thus the following law can be derived.

$$T_c(N) = T_c(\infty) + bN^{-\frac{1}{\nu}} \tag{5.4}$$

where b is some constant. This model has been verified for 2-D Ising models. Similar formulation is intended to be used for the current application. However, in the present problem, $\lambda_{eff} \sim \frac{1}{N}$. Thus, it is impossible to provide a finite effective load per node to an infinite sized network, which means there cannot be any finite load phase transition in an infinite sized network, i.e., $\lambda^c_{eff}(\infty) = 0$, in the current architecture. Hence, for this case

$$\lambda^c_{eff}(N) = bN^{-\frac{1}{\nu}} \tag{5.5}$$

for a coefficient b and an exponent ν. Figure 5.3 shows a good fit of this proposed model with $b = 1.08$ and $\nu = 0.9$.

Figure 5.4 shows the possibility of generating a size-independent global model for phase transition in square-grid communication networks. Such a model can be constructed by using a reduced (i.e., normalized) effective load $\lambda_{red} = \frac{\lambda_{eff}}{\lambda^c_{eff}}$ instead of λ_{eff}. This construction draws inspiration from classical Thermodynamics, where compressibility curves for different pure substances can be unified by using reduced pressure or reduced temperature instead of directly using pressure or temperature.

Remark 2.1. *Critical Slowing Down Note, there is a slight mismatch among the phase transition curves in Fig. 5.4. This can be attributed to the fact of* Critical slowing down *[106]. Let the normalized time-correlation function $\phi_x(t_1 - t_2)$ of an*

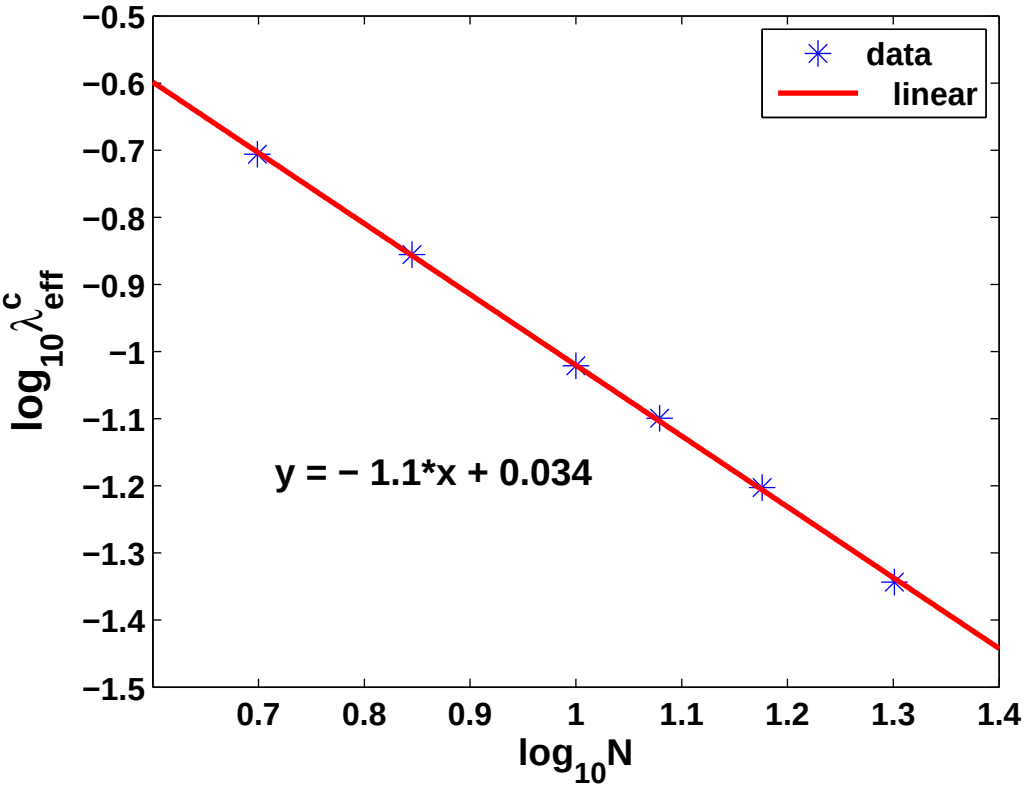


Figure 5.3. Size dependency of critical network load

observable x(t) be defined as [107].

$$\phi_x(t_1 - t_2) = \frac{[\langle x(t_1)x(t_2)\rangle - \langle x\rangle^2]}{[\langle x^2\rangle - \langle x\rangle^2]} \approx e^{-\frac{|t_1-t_2|}{\tau}} \tag{5.6}$$

where, τ is the slowest relaxation time or temporal correlation length of the observable x(t). It is known that near a phase transition point, the relaxation time of the slowest mode of a system diverges, i.e., $\tau \to \infty$. This phenomenon is known as the Critical slowing down *and as a consequence, it takes a long time to obtain two consecutive independent observations near a phase transition point. Therefore, it is very difficult to simulate a large system in the vicinity of the phase transition point.*

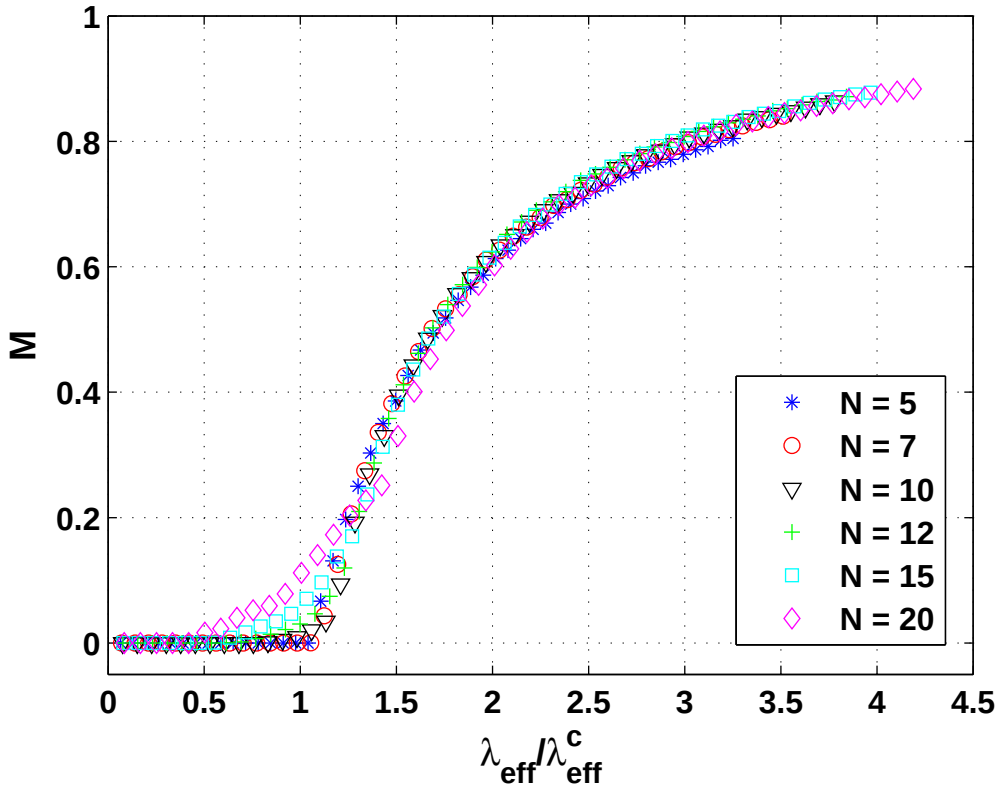


Figure 5.4. Size-independent global model for network phase transition

3 Heterogeneous Packet Transmission: Introduction of a Second Intensive parameter

Previous analyses studied the effects of variation of a single parameter, i.e., packet arrival rate λ, on the average packet drop rate in the network. This subsection introduces a second network parameter namely, the packet transmission probability P for networks transmitting heterogeneous packets. Given that each node has multiple independent queues (buffers) for different packet types, packet transmission probability is defined as the probability of transmitting a particular packet in a time unit.

In the previous case, with a single packet type, P was set to unity. It is obvious that when P is decreased from unity, the network is expected to move from the steady phase to the unsteady phase for lower values of effective load λ_{eff}.

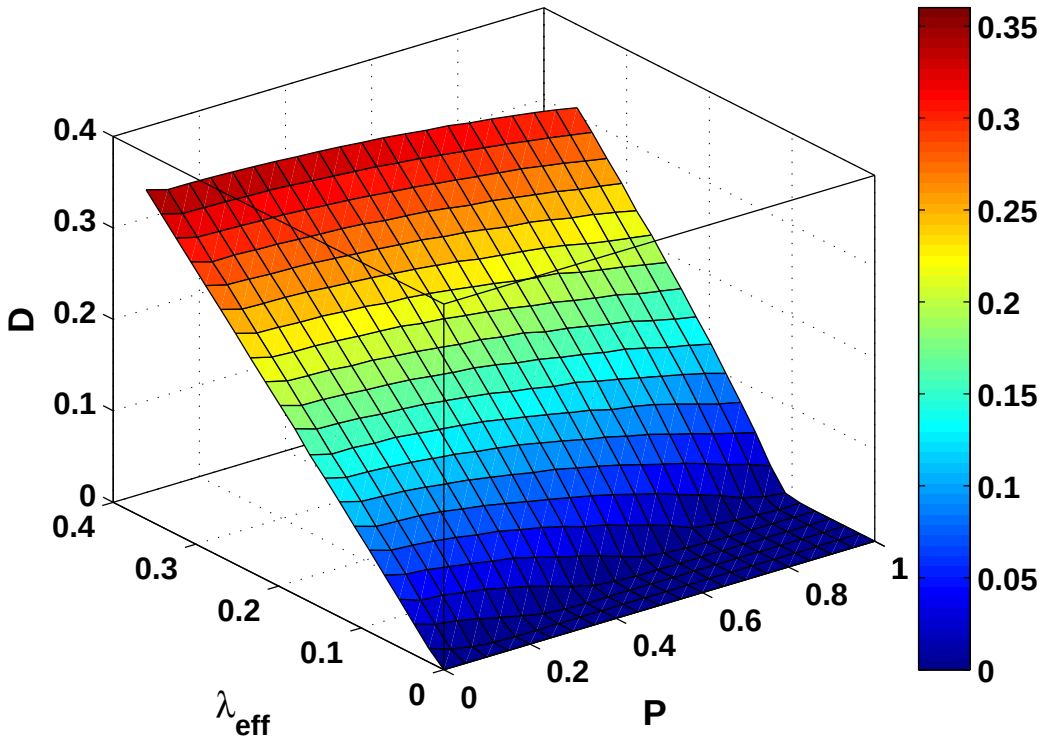


Figure 5.5. Packet drop rate as a function of packet arrival rate and transmission probability

Similar phenomena take place in thermodynamic (e.g., solid-liquid and gas-liquid) systems, where the critical temperature of solid-liquid (or gas-liquid) transition can be altered by a change of super-incumbent pressure; lower pressures lead to lower values of melting (or boiling) point. In this context, the packet transmission probability P of the nodes is called the *network pressure*, while λ_{eff} is called the *network temperature*. Phase transition in the network is therefore a function of a combination of network temperature λ_{eff} and network pressure P. Figure 5.5 presents a 3-D plot of average packet drop rate as a function of λ_{eff} and P and it can be observed that λ^c_{eff} decreases with decrease in P.

3.1 Problem Setup

The problem of heterogeneous packet transmission in networks is formalized in this subsection. Let there be K independent queues for K types of packets for each node of the network. In terms of packet arrival, let the effective Poisson arrival rate of packet type i be simply (subscript *eff* is omitted) denoted by λ_i $(0 \leq \lambda_i \leq 1)$ for $i = 1, 2, ...K$. However, only one channel is used for packet transmission, i.e., at a time instant a node can at most transmit one packet. The probability that a node in the network transmits a packet of type i is denoted by P_i $(0 \leq P_i \leq 1)$. Single transmission channel leads to the constraint $\sum_{i=1}^{K} P_i \leq 1$. With this setup, the network is analogous to multi-component materials in a thermodynamic sense. Thus, phase diagrams can be constructed for the equilibrium states for such networks which eventually lead to a probabilistic (static) control strategy of heterogeneous packet transmission. The next subsection presents the idea of phase diagrams for networks transmitting heterogeneous packets with a representative example of two packet types.

3.2 Construction of Phase Diagrams

In the current setup of independent queues for different packet types, the uncongested phase **U** (congested phase **C**) for a particular packet type is characterized by zero (non-zero) packet drop rate. Thus, with respect to just a particular packet type, a network can be in an uncongested or a congested phase. This leads to the existence of mixed phases where the network remains in uncongested phase for some packet types while being in congested phase for the rest of the packet types. For a network carrying K packet types, there can be 2^K possible phases among which two will be pure phases (fully uncongested or fully congested) and rest of them will be mixed phases. The purpose of a network phase diagram will be to determine the network phase for given λ_i and P_i distributions. A representative example with two packet types is presented here. As discussed above, there can be four different network phases for this example and they are:

- *Completely Uncongested phase* (**U1+U2**): Both packet types are in uncongested phase, which signifies negligibly small values of average packet drop rates D_1 and D_2 for both queues.

- *Congested packet type 1 and Uncongested packet type 2* (**C1+U2**): Packet type 1 has a relatively large (i.e., nonzero) drop rate D_1 while packet type 2 still maintains a negligibly small D_2.

- *Uncongested packet type I and Congested packet type II* (**U1+C2**): Packet type 1 maintains a negligibly small drop rate D_1 while packet type 2 has a relatively large (i.e., nonzero) drop rate D_2.

- *Completely Congested phase* (**C1+C2**): Both packet types are in congested phase, i.e., values of average packet drop rates D_1 and D_2 are nonzero for both queues.

Remark 3.1. *When compared to a multi-component material in thermodynamic sense, the network with two packet types has a remarkable similarity with two-component mixtures, e.g., a mixture of olivine (i.e. isolated tetrahedra) and pyroxene (i.e. single chain tetrahedra). The corresponding phase diagram is known as the binary eutectic phase diagram [108] that explains the chemical process of generating the two immiscible crystals from a completely miscible liquid based on temperature and pressure variation. This process also has four possible phases, which are: completely liquis phase (analogous to **C1+C2**), mixture of liquid olivine with pyroxene crystals (analogous to **C1+U2**), mixture of olivine crystals with liquid pyroxene (analogous to **U1+C2**) and finally, the completely crystalized phase (analogous to **U1+U2**).*

Figure 5.6 shows a 3-D phase diagram for the above example that is constructed via Monte-carlo simulation. Four different phases are properly color coded. Both λ_1 and λ_2 are varied independently from 0 to 1. Planes for three different values of P_1, $P_1 = 0.2(P_2 = 0.8)$, $P_1 = 0.5(P_2 = 0.5)$, $P_1 = 0.8(P_2 = 0.2)$ are provided. Note the change in the sizes of the phases across the range of P_1. The regularity in shapes of different phase zones can be attributed to the independence consideration for the queues of different packet types. This type of phase diagrams lead to the idea of controlling heterogeneous packet transmission from a equilibrium thermodynamics point of view. The basic idea is to move as close as possible to the phase boundary of the **U1+U2** phase from the outside by choosing a proper value of P_1. On the other hand, if the network is already inside the **U1+U2** phase then moving as

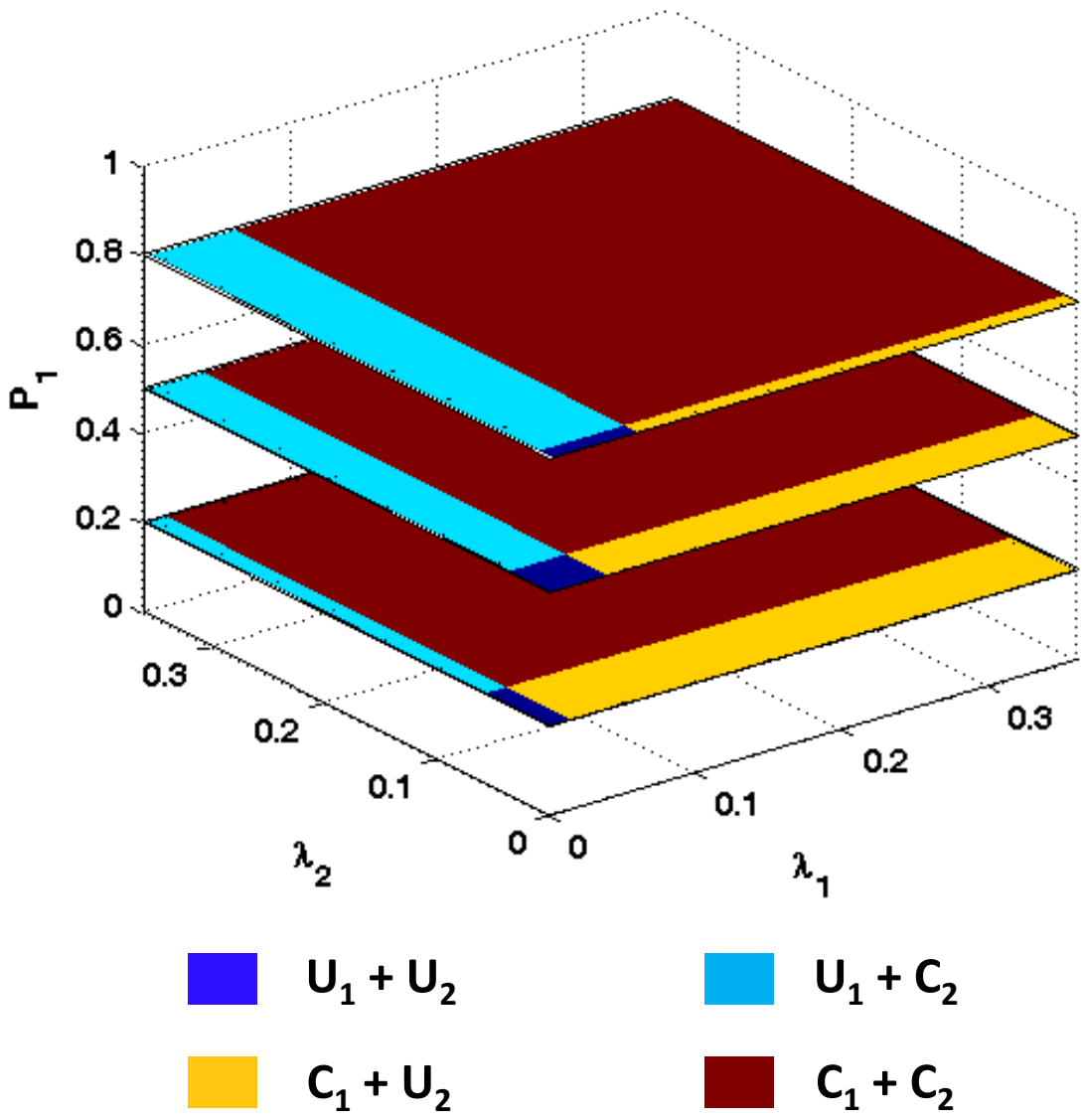


Figure 5.6. Network phase diagram with two packet types

far as possible (by choosing a proper value of P_1) from its phase boundary to be robust against the uncertainty in λ_1 and λ_2 or any other internal disturbance.

4 Control of Heterogeneous Packet Transmission

The thermodynamic approach presented above to analyze and understand congestion phenomenon in communication networks can lead to a novel approach for

controlling heterogeneous packet transmission. This section presents the basic philosophy of such a control approach along with representative numerical experiments on the network architecture of the current study. From control perspective, the network (transmitting heterogeneous packets) congestion states can be categorized into two types of phases. They are (i) Phase type I: At least queue for one packet type is in the congested phase, (ii) Phase type II: Queues for all packet types are in the uncongested phase. The fundamental control objectives are different for these two phase types. For phase type I, the drop rate of at least one packet type is nonzero. The control objective chosen here is choose a distribution of packet transmission probability that minimizes the worst case drop rate among packet types that have nonzero drop rate. The drop rates can be scaled by respective packet arrival rates and as well as by a user defined packet priority distribution. On the other hand, in phase type II, drop rates for all packet types are already zero. Thus, any transmission probability distribution that keeps all the drop rates at zero is an optimal distribution according to the objective described for phase type I. However, a unique solution can be achieved considering a different objective. This objective allows to incorporate robustness against the uncertainty and/or fluctuation in packet arrival rates. The idea here is to choose a transmission probability distribution that pushes the network deeper into the steady phase (far from the phase boundary) so that even with fluctuation or uncertainty in packet arrival rate, the chance to fall into the unsteady phase is minimized. In other words, the objective is to maximize the least uncertainty tolerance in packet arrival rate. As before, the uncertainty tolerance can be scaled by respective packet arrival rates and the user defined packet priority distribution. In the sequel, the control objectives are described formally along with optimal solutions in general and in the special case of the current case study.

4.1 Control Objectives and Optimal Solutions

A user defined packet priority distribution α_i ($0 \leq \alpha_i \leq 1$ and $\sum_{i=1}^{K} \alpha = 1$) is considered for defining the control objectives in both phase types. A higher value of α_i signifies higher priority for packet type i while lowering the drop rate in phase type I or increasing the uncertainty tolerance in phase type II.

4.1.1 Control Objective for Phase Type I

In phase type I, the following model for drop rate D_i of packet type i is assumed:

$$D_i = f_1(P_i, \lambda_i) \quad \forall i \tag{5.7}$$

where, P_i and λ_i are defined as before. As discussed before, the goal here is to find an optimal transmission probability distribution P_i for given packet arrival distribution λ_i and packet priority distribution α_i that minimizes the (weighted) worst case packet drop rate. The cost functional may be written as

$$\begin{aligned} C(P_1, P_2, ...P_K) &= \max_{i=1,..,K} \frac{\alpha_i D_i}{\lambda_i} && (5.8) \\ &= \max_i \frac{\alpha_i f_1(P_i, \lambda_i)}{\lambda_i} && (5.9) \end{aligned}$$

The optimal packet transmission probabilities for each packet type i is denoted as P_i^* and is given by constrained optimization solution to

$$\begin{aligned} (P_1^*, P_2^*, ...P_K^*) &= \arg \min_{\sum_i P_i \leq 1} C(P_1, P_2, ...P_K) && (5.10) \\ &= \arg \min_{\sum_i P_i \leq 1} \max_i \frac{\alpha_i f_1(P_i, \lambda_i)}{\lambda_i} && (5.11) \end{aligned}$$

Although the constraint on the transmission probability distribution is $\sum_i P_i \leq 1$, it is easily seen that for optimality, the inequality should be replaced by an equality sign. The rationale for involving λ_i in the objective function is to normalize the drop rates for each packet type as it was done to define the order parameter M in Section 2. In this context, the solution of the above optimization problem is fairly straight forward. For the optimal packet transmission probability distribution, the following needs to be satisfied:

$$\frac{\alpha_i D_i}{\lambda_i} = \frac{\alpha_j D_j}{\lambda_j} \quad \forall i, j \tag{5.12}$$

Note that given the above condition, if one wants choose a packet transmission

probability distribution to reduce $\frac{\alpha_i D_i}{\lambda_i}$ even further for some value of i, value of $\frac{\alpha_j D_j}{\lambda_j}$ increases for at least one value of j. Thus the *minimax* condition is no longer satisfied.

4.1.2 Control Objective for Phase Type II

In phase type II, $D_i = 0\ \forall i$, i.e., $f_1(P_i, \lambda_i) = 0\ \forall i$. By solving this equation, one gets the equation of the critical curve that gives the value of critical transmission probability P^c for a given value of packet arrival rate λ. Let the critical curve be described as:

$$P^c = f_2(\lambda) \tag{5.13}$$

For a given packet arrival distribution λ_i, if the network is in phase type II then that means:

$$\sum_i P_i^c = \sum_i f_2(\lambda_i) \leq 1 \tag{5.14}$$

This implies that the packet transmission probability distribution P_i^c ($\sum_i P_i^c \leq 1$) is enough for the network to be in the phase type II. However, the queues for different packet types can still use the remaining packet transmission capability of the network expressed as $1 - \sum_i P_i^c$. Therefore essentially the goal here is to distribute this remaining packet transmission capability among the queues in an optimal sense. To formalize this problem, a virtual packet arrival distribution λ_i^U is defined. λ_i^U ($\lambda_i \leq \lambda_i^U \leq 1$) denotes one distribution that includes uncertainty/fluctuation in packet arrival rates for which the network just remains in phase type II. The corresponding packet transmission probability P_i is given as $f_2(\lambda_i^U)$. With this consideration, the following equation can be written:

$$\lambda_i^U = \lambda_i + \tilde{\lambda}_i \tag{5.15}$$

$$\Rightarrow \tilde{\lambda}_i = f_2^{-1}(P_i) - \lambda_i \tag{5.16}$$

where, $\tilde{\lambda}_i (\geq 0)$ uncertainty tolerance provided for packet type i, for which the network still remains in the steady phase (phase type II). Given the above notations, the objective of the optimization problem for phase type II is to choose a transmission probability distribution P_i that maximizes the minimum uncertainty

tolerance for packet arrival statistics. Thus the objective functional is the worst case (weighted) tolerance $\tilde{\lambda}_i$ that is given as:

$$O(P_1, P_2, ..., P_K) = \min_{i=1,..,K} \frac{\tilde{\lambda}_i}{\alpha_i} \tag{5.17}$$

$$= \min_i \frac{f_2^{-1}(P_i) - \lambda_i}{\alpha_i} \tag{5.18}$$

The constrained optimization problem that has to be solved to obtain the optimal packet transmission probability is given as

$$(P_1^*, P_2^*, ...P_K^*) = \arg \max_{\substack{\sum_i P_i \leq 1 \\ P_i \geq f_2(\lambda)}} O(P_1, P_2, ...P_K) \tag{5.19}$$

$$= \arg \max_{\substack{\sum_i P_i \leq 1 \\ P_i \geq f_2(\lambda)}} \min_i \frac{f_2^{-1}(P_i) - \lambda_i}{\alpha_i} \tag{5.20}$$

The solution for the current optimization problem is very similar to that of the optimization for phase type I, i.e.,

$$\frac{\tilde{\lambda}_i}{\alpha_i} = \frac{\tilde{\lambda}_j}{\alpha_j} \quad \forall i, j \tag{5.21}$$

Note, the use of the packet priority distribution α_i in the optimization problem. The queue for packet type i gains more uncertainty tolerance for a higher value of α_i.

4.2 Approximation of Functional Forms

The formulation of the general control strategy is presented above. For a given network structure and routing strategy, one needs to estimate the functions f_1 and f_2 in order to implement the control strategy. An example procedure is shown in this subsection with respect to the network considered in the current study.

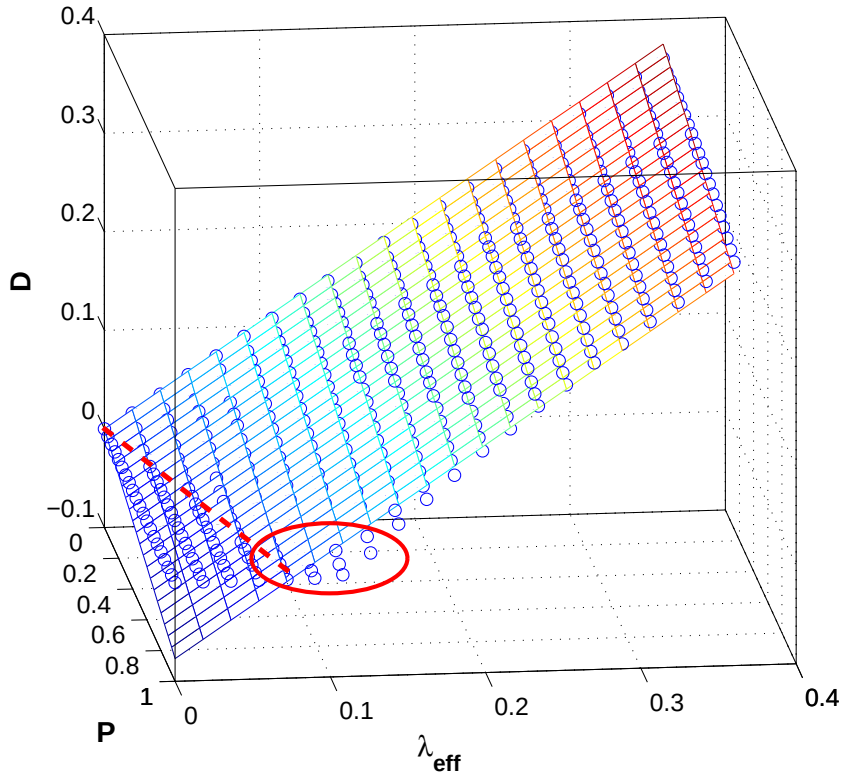


Figure 5.7. Approximation of functional forms for packet drop rate in square grid networks

Figure 5.5 gives an idea about the functional forms for both f_1 and f_2. For phase type I ($D > 0$), function f_1 can be approximated with a linear function, i.e., with a plane. As there is no bias, the plane equation for D as a function of P and λ is:

$$D = aP + b\lambda \tag{5.22}$$

The linear least square fit of the experimental data is shown in Fig. 5.7 and the error is found to be sufficiently low. However, error is observed to increase near the critical curve (see the circled zone in the figure). This can be attributed to the problem *critical slowing down* near the critical points as discussed earlier. Note that D decreases with increase in P and D increases with increase in λ. Therefore, while a will have negative sign, b will be a positive constant.

The critical curve equation (for phase type II) can be found by setting $D = 0$. In other words, the critical curve is the straight line at which the least square fit

plane intersects the $D = 0$ plane. Thus we can obtain the following equation for the critical curve:

$$P^c = c\lambda \tag{5.23}$$

where, $c = -b/a$. It is clear that c will be a positive constant as P^c will increase with increase in λ (also, a is negative and b is positive). Figure 5.7 also shows the critical line (dotted) for the current network model.

The above approximate functional forms allow to derive the closed form solutions (for both phase types) of the optimal packet transmission probability distribution by solving simple algebraic equations.

4.2.1 Solution for Phase Type I

In Phase type I, the optimal distribution is given by the solution of the following system of algebraic equations:

$$\frac{\alpha_i a}{\lambda_i} P_i + \alpha_i b = \frac{\alpha_j a}{\lambda_j} P_j + \alpha_j b \quad \forall i, j \tag{5.24}$$

For a network with K packet types, the above system provides $K - 1$ equations and the K^{th} equation is provided by the constraint $\sum_i P_i = 1$. Equation 5.24 can be rearranged to obtain

$$Q = \frac{\alpha_i |a|}{\lambda_i}(P_i^c - P_i) \ \forall i \tag{5.25}$$

where, Q is the value of the expression on the right hand side and $P_i^c = c\lambda_i$ (also note that $a = -|a|$). By summing the expressions for all i and using $\sum_i P_i = 1$, Q can be evaluated as

$$Q = \frac{\sum_{i=1}^{K} P_i^c - 1}{\sum_{i=1}^{K} (\frac{\lambda_i}{\alpha_i |a|})} \tag{5.26}$$

Using the value of Q, P_i can be evaluated as

$$P_i = P_i^c - \frac{Q\lambda_i}{\alpha_i |a|} \tag{5.27}$$

However, for certain choices of α_i, solution of P_i may be negative for a given λ_i distribution, which in turn does not satisfy the constraint $P_i \geq 0 \ \forall i$. Therefore,

the negative solutions should be changed to zeros and the optimization should be performed again for the rest of the packet types in order to satisfy the second constraint $\sum_i P_i = 1$. The algorithm for such a sequential optimization procedure is given below.

Algorithm for Sequential Optimization

$Q^{(1)} = \frac{\sum_{i=1}^{K} P_k^c - 1}{\sum_{i=1}^{K} (\frac{\lambda_k}{\alpha_k |a|})}$;
$P_j^{(1)} = P_j^c - \frac{Q^{(1)} \lambda_j}{\alpha_j |a|}$; (Evaluate $P_j^{(1)}$ $\forall j$)
$i = 1$;
while $P_j^{(i)} < 0$ for some j **do**
  for all $j : P_j^{(i)} < 0$ **do**
    $P_j^{(i)} \leftarrow 0$;
  end for
  $i \leftarrow i + 1$
  $Q^{(i)} = \frac{\sum_{k=\{j:P_j^{(i-1)} \neq 0\}} P_k^c - 1}{\sum_{k=\{j:P_j^{(i-1)} \neq 0\}} (\frac{\lambda_k}{\alpha_k |a|})}$;
  for all $j : P_j^{(i-1)} \neq 0$ **do**
    $P_j^{(i)} = P_j^c - \frac{Q^{(i)} \lambda_j}{\alpha_j |a|}$;
  end for
end while

Note that as a consequence of the above algorithm, the transmission probability distribution may have $P_i = 0$ for some values of i and rest of the elements will have $P_j > 0$ (where, $j \in \{1, 2, \cdots, K\} \setminus \{i\}$). The scaled drop rates can be written as

$$\frac{\alpha_i D_i}{\lambda_i} = \alpha_i b - \frac{\alpha_i |a|}{\lambda_i} P_i \tag{5.28}$$

Therefore, $\forall j, k \in \{1, 2, \cdots, K\} \setminus \{i\}$ (i.e., for all packet types with nonzero transmission probability), the solution obtained from the algorithm will satisfy

$$\frac{\alpha_j D_j}{\lambda_j} = \frac{\alpha_k D_k}{\lambda_k} \quad \forall j, k \tag{5.29}$$

Thus, due to the constraints on the transmission probability distribution, the optimal condition described in Eqn. 5.12 will only be satisfied on a subset (containing packet types with nonzero transmission probability) of the set of all packet types. Rest of the packet types will have zero transmission probability essentially due to their low $\frac{\alpha}{\lambda}$ ratio. More over, it is ensured that scaled drop rates for packet types with $P_i = 0$ will be less than those of their counterparts with $P_j > 0$, i.e.,

$$\frac{\alpha_i D_i}{\lambda_i} < \frac{\alpha_j D_j}{\lambda_j} \tag{5.30}$$

where, $i \in \{m : P_m = 0\}$ and $j \in \{m : P_m > 0\}$. As an example, for a network with two packet types, the optimal packet transmission probability for type 1 is given by:

$$P_1 = \begin{cases} 0 & \text{if } \frac{\alpha_1}{\alpha_2} < \max\left[1 - \frac{1}{c\lambda_2}, 0\right] \\ \frac{\alpha_2 + c\lambda_2(\alpha_1 - \alpha_2)}{\lambda_2\left[\frac{\alpha_1}{\lambda_1} + \frac{\alpha_2}{\lambda_2}\right]} & \text{if } \max\left[1 - \frac{1}{c\lambda_2}, 0\right] \leq \frac{\alpha_1}{\alpha_2} \\ & \quad \leq \frac{1}{\max\left[1 - \frac{1}{c\lambda_1}, 0\right]} \\ 1 & \text{if } \frac{\alpha_1}{\alpha_2} > \frac{1}{\max\left[1 - \frac{1}{c\lambda_1}, 0\right]} \end{cases}$$

and $P_2 = 1 - P_1$. Note, in phase type I, $\sum_i P_i^c = c \sum_i \lambda_i > 1$ which ensures that $\max\left[1 - \frac{1}{c\lambda_2}, 0\right] \leq \frac{1}{\max\left[1 - \frac{1}{c\lambda_1}, 0\right]}$.

4.2.2 Solution for Phase Type II

For phase type II, as discussed earlier, the remaining transmission capability (beyond the critical curve) $1 - \sum_i P_i^c$ is distributed among the queues of different packet types. To achieve the closed form solution, we begin with the solution for phase type II (given in Eqn. 5.21). Using the model in Eqn. 5.23, the optimal solution can be written as:

$$\frac{\frac{P_i}{c} - \lambda_i}{\alpha_i} = R \ \ \forall \ i \tag{5.31}$$

where, R is a constant. Therefore,

$$P_i = c(R\alpha_i + \lambda_i) \tag{5.32}$$

The value of constant R is evaluated using the constraint $\sum_k P_k = c\sum_k(R\alpha_k + \lambda_k) = 1$ as

$$R = \frac{1}{c} - \sum_k \lambda_k \tag{5.33}$$

Thus, the optimal packet transmission probability distribution for phase type II is given by,

$$P_i = c\lambda_i + \alpha_i \left[1 - c\sum_k \lambda_k\right] \tag{5.34}$$

For example, the exact solution for network with two packet types is:

$$P_1 = \alpha_1 + c\left[\lambda_1 - \alpha_1(\lambda_1 + \lambda_2)\right] \tag{5.35}$$

and $P_2 = 1 - P_1$.

4.3 Representative Experimentation

A representative experimentation is presented in this subsection to validate the effectiveness of the control strategy developed above. The simulation test bed is the same network system analyzed in this chapter with two different packet types: Packet type 1 and Packet type 2. Following the notations used in this chapter, the packet arrival statistics is described by λ_1 and λ_2; packet priority distribution is denoted by α_1 (taken as 0.6) and α_2 (taken as 0.4); drop rates are denoted by D_1 and D_2. The goal here is to estimate λ_1 and λ_2 over a time window, then to detect the phase type with respect to the estimated arrival rates and implement the optimal packet transmission probability distribution P_1 and P_2. Let the time-scale of network dynamics be denoted as fast time-scale (denoted by $Time$) and the time-scale of arrival statistics estimation be denoted as slow time-scale. In this experiment, a slow time-scale epoch contains a window of 2000 fast time-scale epochs.

Figure 5.8 shows the steady state position of the network in the equilibrium

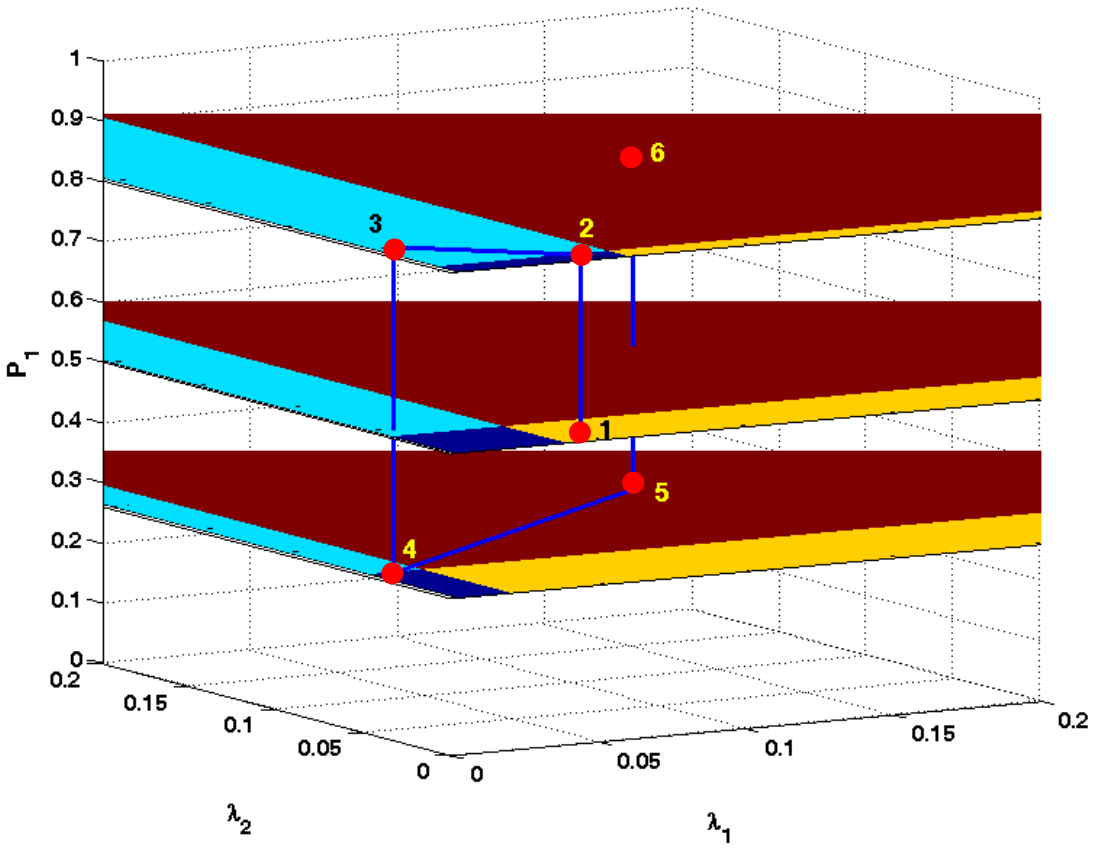


Figure 5.8. System trajectory in the phase diagram for the representative experimentation

phase diagram at different slow time-scale epochs. The phase diagram is same as the one shown in Section 3. Also the time-series response of D_1 and D_2 is shown in Fig. 5.9 along with the corresponding control set point P_1.

For model identification purpose, parameters are identified to be a = -0.0771, $b = 1.0404$, and $c = -b/a = 13.4942$. Initially ($Time = 1$), the network is at state 1 (as shown in Fig. 5.8) with $\lambda_1 = 0.05$ and $\lambda_2 = 0.01$ and $P_1 = P_2 = 0.5$. Over the slow time-scale epoch 1, λ_1 and λ_2 are estimated and at the beginning of slow time-scale epoch 2 ($Time = 2000$), the network moves to the optimal location with $P_1 \approx 0.8$ (state 2 in Fig. 5.8). As a consequence, the very small ripples noticed in D_1 time series in slow time-scale epoch 1 disappears. Note, that this is a phase type II control. Although, lower P_1 would have been fine, it went to $P_1 = 0.8$ to optimally move away from the phase boundary. At the beginning of slow time-scale epoch 3 ($Time = 4000$), the packet arrival statistics is changed to $\lambda_1 = 0.01$

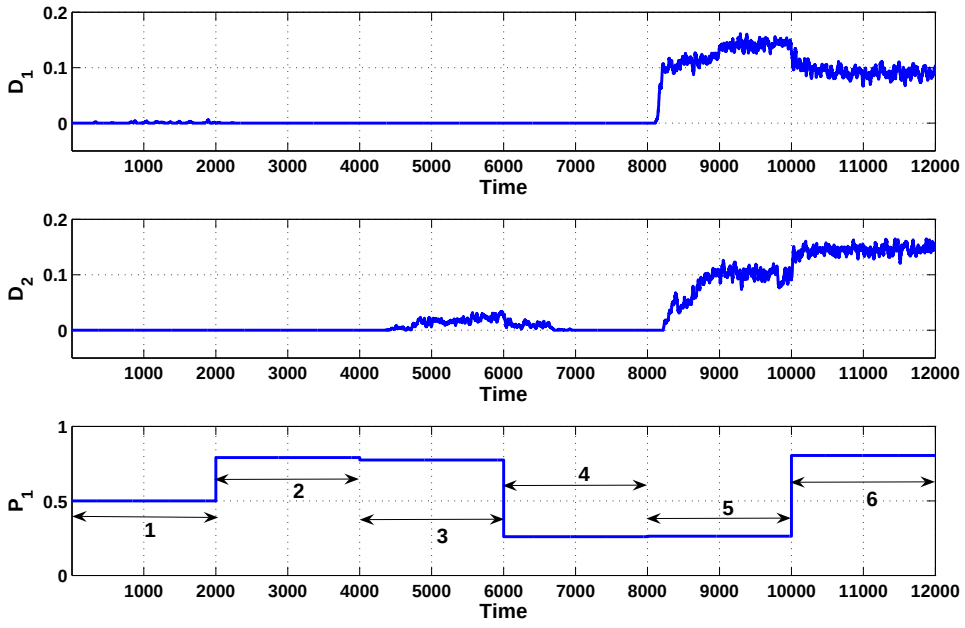


Figure 5.9. Time series observation of packet drop rates for packet type 1 (D_1) & packet type 2 (D_2) and transmission probability for packet type 1 (control input) P_1

and $\lambda_2 = 0.05$ (state 3 in Fig. 5.8) and it can be observed that after some initial transience response D_2 settles to a nonzero value during slow time-scale epoch 3 whereas, D_1 remains at zero. Upon estimation of λ_1 and λ_2 over slow time-scale epoch 3, the controller takes the network to the optimal location with $P_1 \approx 0.26$ (state 4 in Fig. 5.8) at the beginning of slow time-scale epoch 4 ($Time = 6000$). As a consequence, D_2 response dies down keeping the $D_1 = 0$ response. The network still in phase type II. Finally, at the beginning of slow time-scale epoch 5 ($Time = 8000$), the packet arrival statistics is changed to $\lambda_1 = 0.15$ and $\lambda_2 = 0.15$ (state 5 in Fig. 5.8); both D_1 and D_2 responses settles in a nonzero value over the slow time-scale epoch 5. However, the steady state value of D_1 is higher than that of D_2 as the packet transmission probability distribution is still in favor of packet type 2 (with $P_1 = 0.26$ and $P_2 = 0.74$). From control point of view, the network is in phase type I and the controller drives the network to the optimal location with $P_1 \approx 0.8$ (state 6 in Fig. 5.8) at the beginning of slow time-scale epoch 6 ($Time = 10000$). Higher value of P_1 compared to that of P_2 is due to the higher

priority of packet type 1 over packet type 2($\alpha_1 > \alpha_2$). Consequently, D_1 settles to a lower value compared to D_2 over the slow time-scale epoch 6.

5 Summary, Conclusions and Future work

This chapter presents Statistical Mechanics-inspired analysis of critical phenomena in square-grid wired communication networks and validates the pertinent results on a simulation test bed consisting of a wired communication network model. In order to unambiguously elucidate the underlying concepts, the notion of order parameter and other global intensive parameters (e.g., network analogs of order parameter, temperature and pressure) is introduced. The global order parameter defined here can be used in general as a measure of performance and stability. A comprehensive finite-size scaling analysis has been performed for a particular type of network structure. Phase diagrams are constructed for networks transmitting heterogeneous packet types that lead to a novel approach for controlling heterogeneous packet transmission. The basic philosophy of such a control approach along with representative numerical experiments are presented. The control strategies are formulated based on the objectives of incorporation of robustness in the fully uncongested phase and minimizing the worst-case packet drop rate in congested phases. Closed form solutions for optimal control are developed for the square grid communication network structure considered in this chapter.

Although the network model used here is rather simple compared to real-life complex network architectures, the algorithms for critical behavior analysis are potentially useful for different classes of communication networks in real life. For example, the current network model with boundary-node packet generation and multi-source multi-destination distributed routing is not necessarily constrained by the wired structure of the model. Therefore, this analysis can be extended to ad-hoc wireless sensor networks for boundary surveillance problems, where a spatially distributed network of autonomous agents with (possibly multi-modal) sensing capabilities collaboratively monitor a given environment. In general, different sensors sense the environment in a distributed manner and need to communicate with other types of sensors for information fusion to make a comprehensive decision. Therefore, complex multi-hop routing of (possibly multi-priority) infor-

mation packets in a sensor network bears significant relevance. In addition, such a simple network structure may serve as subsets of large complex networks, where a thorough understanding of small subsets is necessary for control of the entire network. In this chapter, the packet arrival statistics is estimated in a centralized fashion and a global packet transmission probability distribution is broadcasted for optimal control. In this context, the (statistical mechanics-inspired) *ensemble approach* may allow the control algorithm to be executed over a network in a communication-constrained environment via sparse statistical sampling for the estimation of network performance parameters. It will be more useful if each node takes its own decision based on local (its own or of its neighborhood) estimation of packet arrival statistics and yet the global objective is satisfied. Due to *ensemble approach* of the current study, such a distributed control problem is not intractable as well and will be an important topic of future research. Sensor networks have a wide variety of applications in military (e.g., urban surveillance, and antisubmarine warfare), industrial (e.g., process health monitoring), and civilian (e.g., home automation and human health monitoring) sectors [109].

Apart from the issue of distributed realization, the following topics are recommended for future research from the perspectives of stability, performance analysis, and decision & control.

- Validation of current findings in more complex network scenarios with various features (e.g., layered architectures and industry-standard protocols).
- Investigation of the effects of various packet arrival statistics, various distribution of source and destination nodes.
- Investigation of the effects of network topology (e.g. rectangular instead of square grid)
- Dynamic analysis for convergence and stability of the network system as augmentation of the equilibrium behavior analysis reported in the chapter.

Chapter 6

Distributed Decision Propagation in Mobile-agent Networks

This chapter investigates the problem of distributed decision propagation in mobile-agent networks. Clearly, there are two aspects of the problem: time varying network topology and distributed agent interaction policy (see Appendix-B for details). It is found that message broadcast duration or state updating interval can be an actuation parameter for changing time-averaged network topology. Two agent interaction policies (one linear and one nonlinear) are studied in this chapter. The linear strategy is essentially a generalized gossip algorithm under the language-measure-theoretic framework. It is shown that a single parameter in the language-measure-theoretic policy can be used to control the tradeoff between *Propagation Radius* and *Localization Gradient* as well as the temporal convergence properties. The nonlinear strategy is a threshold based policy, known as binary decisions with externalities. The threshold parameter can be used to trigger or restrain the decision propagation. The influence of (large) seed size on the propagation phenomenon has been exploited to control the threat level threshold, beyond which the decision/awareness propagates throughout the network.

1 Motivation

Analysis and development of distributed decision propagation and control mechanisms in mobile-agent networks have drawn much attention recently due to their

relevance in engineering problems, such as surveillance and reconnaissance by autonomous vehicles with limited capabilities, trust establishments in mobile ad hoc networks (MANETs) [110] and threat monitoring by mobile sensor networks. However, in many applications, diffusion of aggregated information is more relevant compared to individual sensor information [111] mostly due to its robustness to individual agent's failure in detection/communication. In a resource constrained environment, mobile agents have potential advantages over static networks in terms of coverage and time-criticality. This chapter deals with global propagation of a localized awareness in a leaderless environment in a robust and completely distributed manner.

In general, there are two aspects of interacting agent systems, namely (i) network topology and (ii) agent interaction dynamics. Network topology is inherently time varying in the present context, which makes the analysis of such complex systems much harder compared to their static counterparts. Usually, similar time-varying situations arise in social network studies [112] and they are modeled by various graphical structures, such as: multiple instances of uniform random graphs, scale-free networks and small world networks [27]. Synchronization problems have been solved for time-varying networks where essentially the network topology is modeled as fast switching among a finite number of instances of random graphs with same specifications [113]. However, all such models do not necessarily consider the agent mobility statistics or inter-agent communications due to proximity. Recently, the analysis of so called proximity networks [114] (also called the moving neighborhood networks [115]) has been performed to model contact/collision based disease spreading. This may be considered as the first step towards analyzing the mobile-agent scenario in an actual sense. This chapter adopts such developments reported in social network studies, to analyze the expected network topology in a mobile-agent framework.

Distributed agent interaction dynamics for decision propagation has several mechanisms available in literature, examples are game theoretic [110], biology inspired, physics inspired (Ising/Potts models) [116], bootstrap percolation [117] and majority voting [118]. Gossip algorithms are the most studied interaction dynamics in the context of consensus. However, in many applications, large group of agents do not seek consensus. Often localized percolation of decision is desired

to localize the the information source. This chapter proposes a generalized gossip algorithm based on the recently developed *language-measure theory* [55, 119] in a time-varying network topology for distributed global propagation of decision/awareness in mobile-agent networks and it is shown that the generalization parameter controls the tradeoff between *Propagation Radius* and *Localization Gradient*. Analysis of moment dynamics [120] (up to second order) is presented and verified using simulation experiments. Variance analysis is performed under two conditions: (i) *Congruous time-scale:* when network topology evolution and agent interaction dynamics has similar time-scales and (ii) *Disparate time-scale:* when (faster) agent interaction dynamics can be considered as a singular perturbation with respect to (slower) network topology evolution.

While the linear strategies (such as the one described above) may be analytically tractable, their nonlinear counterparts (especially for mobile-agent scenario) are mostly analytically intractable; hence most results are simulation-based. However, interesting emergent behaviors (e.g., phase transition [121]) may be seen for strategies of this genre [122]. Watts [123], developed a simple threshold-based neighborhood interaction model, known as the binary decisions with externalities, that can be used to explain global cascading phenomena in large electrical power grids and opinion spreading in a large community. This model shows that global cascading from a very small localized seed in a complex (time-invariant) network can be triggered by a very small change in a single agent interaction parameter, which is known as a phase transition phenomenon. This study adopts this mechanism in a time-varying network topology for global propagation of decision/awareness in mobile-agent networks. It is shown that a global propagation of decision can be triggered depending on the hot-spot strength [124].

The chapter is organized in eight sections including the present one. The representation of a mobile-agent scenario in terms of proximity networks is formulated in Section 2. Section 3 presents a brief review of language-measure theory and formulates the problem in a language-measure-theoretic setting. While critical observations are made using simulation experiments in Section 4, they are explained via analytical results obtained in Section 5. Section 6 introduces the nonlinear threshold based policy that is used for the decision propagation problem in Section 7. Finally, the chapter is summarized and concluded in Section 8 with

recommendations of future work.

2 Proximity Networks

In the context of surveillance and reconnaissance by mobile agents, decentralized operations involving mobile agents is very useful. However, the network-theoretic modeling becomes challenging for such cases due to the time varying topology, which is a consequence of agent (node) mobility. Recent literature in social network studies have addressed similar problems by introducing the concept of proximity networks [114, 115]. This section describes a mobile-agent scenario in terms of proximity networks [114] and analyzes the relevant network parameters including degree distribution and coordination number (expected degree) [125].

2.1 Model Description

Let the area of a two dimensional (Euclidean) operational region be A. In the present case, A is assumed to be a square area with side length L, i.e., $A = L^2$. Initially, N agents are distributed randomly in the given area, and the agent density is defined as $\rho = N/A$. The uniform radius of communication for each agent is denoted by R, i.e., two agents can only communicate (to say, exchange their state values) when the distance between them is less than R. The agents move in a 2-D random walk fashion where the velocity v is same for all the agents in the current setup. The random walk is realized by independently choosing a random direction of motion by all agents at every time step. During its motion, every agent is assumed to broadcast a message (e.g., its own state) over a certain time window that is known as the message lifetime L_m. At the same time, the agent receives messages (e.g., state values) from other agents, which may come within the distance R. After expiry of a message, an agent possibly undergoes a state updating and then it starts broadcasting the new state for another window of message lifetime. Figure 6.1 presents an illustration of this scenario.

Based on realistic scenarios, the following assumptions are made regarding the network parameters mentioned above. Firstly, the goal of an intelligent surveillance and reconnaissance by mobile agents mission is to monitor a sufficiently large

area with relatively fewer agents, where the agents may not form a fully connected network in terms of communication. This fact translates into the assumption of $R/L << 1$, with sufficiently small ρ. Secondly, the consideration of physical motion of an agent is always a very expensive option. Hence, the length scale of the physical movement is kept small compared to the communication length scale, i.e., $x/R < 1$, where x denotes displacement in one time unit. These considerations lead to the assumption of $x/L << 1$. In the present study, the chosen parameters are: $L = 1000$, $N = 100$, $R = 100$, $x = 20$. If the message lifetime L_m is very small, then the effective network will be equivalent to multiple instances of static random graphs. On the other hand, the network eventually becomes fully connected as $L_m \to \infty$. Thus, to model realistic situations, L_m should be chosen appropriately based on the other network parameters. Let the life status of the current message of an agent i at a time instant t be denoted as $l_i(t)$. If the initial life status of agent i is drawn from the uniform distribution $U[0, L_m]$ $\forall i$, then for a reasonable value of L_m, one can obtain a steady-state condition of the mobile-agent network after the initial transience, where the total number of inter-agent communication links fluctuates around a constant value. Also, the message/state updating occurs in a non-synchronous manner in the agent population. However, for analytical tractability of the agent interaction policy, only synchronous message/state updating is considered. Although obstacle avoidance is a natural component in any agent mobility model, it is not considered in the current setup. To analyze the present network, certain non-dimensional quantities can be constructed involving ρ, R, v and L_m (please see [125] for details). However, in real life, change of R and v for an agent may involve expensive procedures. To change ρ, either new agents need to be deployed or some agents have to be withdrawn; hence, ρ is not a suitable control parameter. Thus, L_m is considered to be the only practical tunable parameter for control purposes in the present context. The following section analyzes the effects of L_m on the time averaged network topology that is primarily represented by the expected degree and the degree distribution of the network.

2.2 Degree Distribution

The degree of a node is defined to be the number of nodes in the network it is connected to and the degree distribution $P(k)$ for a network is defined to be

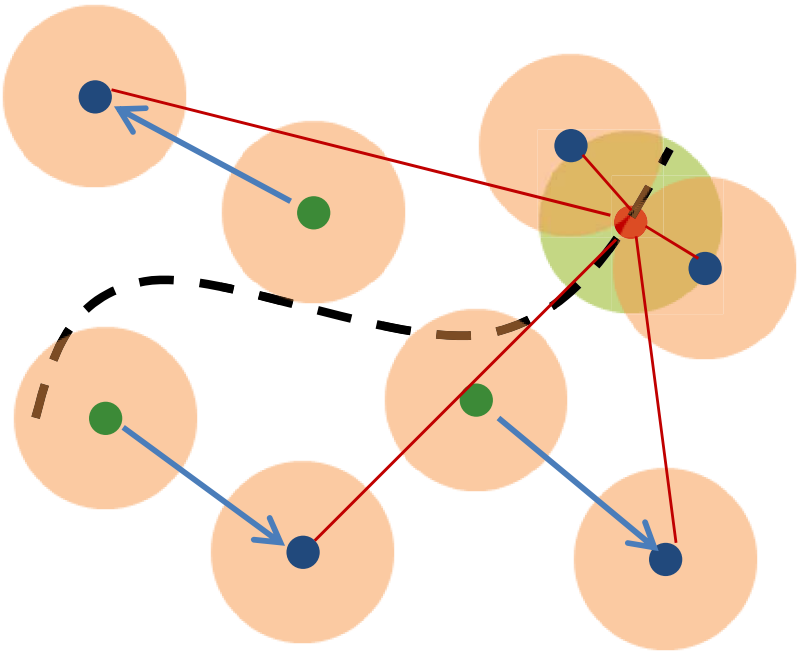


Figure 6.1. Illustration of mobile-agent Proximity Network; Communication zones are shown as discs around the agents; While simple lines denote the communication links between agents, motion of agents is shown by uni-directional arrows; Note, effective communication link remains between two agents until the end of message lifetime even if they move out of each other's communication zones.

the probability distribution function of degrees over the entire network. These definitions are straight forward for static networks. In the present context of dynamic networks, let $P_i(k, L_m(i))$ be defined as the distribution (computed over time) of the number of distinct agents that communicate with a given agent (node) i within its one message lifetime $L_m(i)$. The degree distribution as a function of L_m is defined as:

$$P(k, L_m) = \frac{1}{n(L_m)} \sum_{\{i:L_m(i)=L_m\}} P_i(k, L_m(i)) \tag{6.1}$$

where, $n(L_m)$ is number of agents in the network with message lifetime L_m. Finally, the overall network degree distribution can be defined as the expected value of $P(k, L_m)$, i.e.,

$$P(k) = \frac{1}{N} \sum_{\{L_m\}} n(L_m) \cdot P(k, L_m) \tag{6.2}$$

where N is total number of agents in the network. As described earlier, the mobile-agent network model considered in this chapter follows the structure of proximity networks that is used for modeling several social network phenomena. It has been shown in literature [114, 125] that the nature of degree distributions for proximity networks can be different (e.g., Poisson distribution and Power law distribution as in scale-free networks) depending on the parameters of the mobile-agent dynamics. For the parameter considerations made in this chapter (described in 2.1), it is expected that the time-averaged degree distribution $P(k, L_m)$ will follow a Poisson distribution. A brief sketch of the proof is provided here for completeness of the chapter.

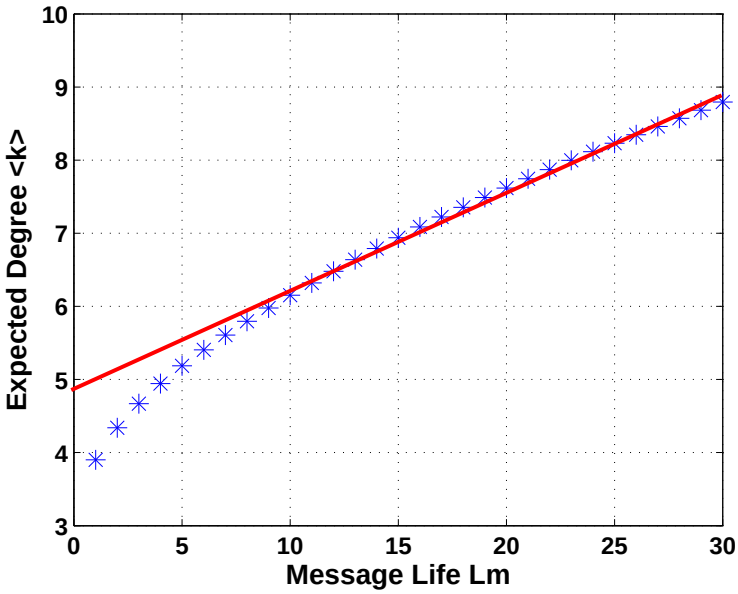


Figure 6.2. Variation of Expected Degree $< k >$ of the Network with Homogeneous Message Life L_m.

Let an agent i with message lifetime L_m be considered for analysis and the probability that an agent j $(j \neq i)$ comes inside the zone of communication of i within the time window L_m be denoted as $p_{ij}(L_m)$. Clearly, $p_{ij}(L_m)$ is an increasing function of L_m. Now, given any stage of network evolution, the probability that i and j communicate in the next time step is modeled as $\alpha g_i g_j (\leq 1)$, where g_l is called the *gregariousness* (i.e., tendency to communicate with other agents) of agent l in accordance with social network literature and α is a parameter that incorporate spatial information of the network, such as the agent density. Also,

it is understood that g_l is a function of the radius of communication and velocity of agent l. With this model and assuming independent activity at each time step, the probability that agent i and j do not communicate within L_m is $(1-\alpha g_i g_j)^{L_m}$. Therefore, $p_{ij}(L_m) = 1 - (1 - \alpha g_i g_j)^{L_m}$. The expected degree $< k >_i$ can be calculated as:

$$\begin{aligned} < k >_i &= \sum_{j=1}^{N} p_{ij}(L_m) \\ &\simeq \alpha L_m g_i (\sum_j g_j) \quad for\ \alpha L_m << 1 \end{aligned} \tag{6.3}$$

The assumption of $\alpha L_m << 1$ is realized if α is very small and at the same time L_m does not have a very large value; small value of α provides an upper bound on the maximum number of agents that agent i can communicate in one time step (this is also known as the *exclusion constraint* [114]). In this chapter, radius of communication and velocity are the same for every agent. Thus, all agents share a uniform gregariousness, say g. Therefore, $p_{ij}(L_m)$ is independent of agent specifications i and j and is denoted as $p(L_m)$ or simply p. Also, all agents are assumed to have same message lifetime L_m. With these assumptions, numerical experiments are performed to calculate the expected degree $< k >$ of the network for various values of homogeneous L_m. Figure 6.2 shows the result obtained from these experiments and an approximately linear relation between $< k >$ and L_m (as derived before) can be observed beyond $L_m = 9$. Now, with homogeneous p and L_m across the network, the degree distribution $P(k, L_m)$ is written as:

$$\begin{aligned} P(k, L_m) &= \binom{N}{k} p^k (1-p)^{N-k} \\ &\simeq \frac{< k >^k}{k!} e^{-<k>} \quad for\ N >> 1 \end{aligned} \tag{6.4}$$

Figure 6.3 shows the degree distribution $P(k, L_m)$ obtained from numerical experiments performed for $L_m = 1, 20, 30$ to be Poisson in nature. Note that the degree distribution for $L_m = 1$ represents the characteristics of a static proximity network. However, it is shown in [114] that by choosing non homogeneous $p_{ij}(L_m)$, one may obtain other types of degree distributions (e.g., power-law distribution)

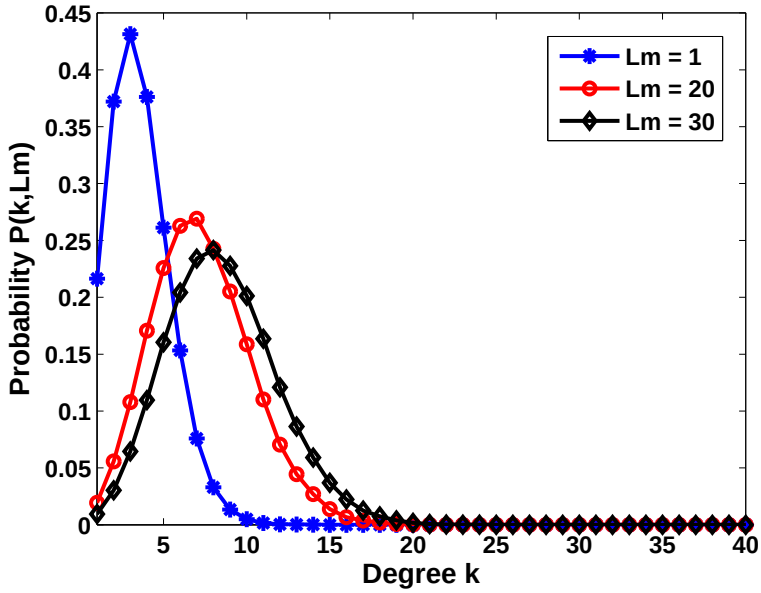


Figure 6.3. Plot of Degree distribution $P(k, L_m)$ of the mobile-agent network for $L_m = 1, 20, 30$.

as well. Thus, degree distribution and expected degree of the network (i.e., the expected network topology) can be controlled by varying L_m.

3 Linear Agent Interaction Dynamics: Language-measure-theoretic Policy

This section presents a brief review of language-measure theory for completeness of the chapter followed by the problem formulation and relevant analytical results in a language-measure-theoretic setting.

3.1 Brief Review

This section summarizes the concept of signed real measure of probabilistic regular languages generated by probabilistic finite state automata(PFSA) [55, 119]. However, since the topic of regular languages is beyond the scope of this chapter, the concept of real measures have been defined only on Markov chains. More details are provided in Appendix-C.

Consider a Markov chain, denoted by the 3-tuple (Q, Π, χ), with a set of states

$Q = \{q_1, q_2, ..., q_N\}$. The state transition function $\Pi : Q \times Q \rightarrow [0, 1]$ depicts the transition probabilities of the Markov chain. Π in matrix from is a $N \times N$ stochastic matrix. The state characteristic function $\chi : Q \rightarrow \mathbb{R}$ assigns a signed real weight to each state. As the number of states is finite, the vector form of the characteristic function may be written as $\chi = \left[\chi^1, \chi^2, ...\chi^N\right]^T$.

A real measure ν_θ^i for state i is defined as

$$\nu_\theta^i = \sum_{\tau=0}^{\infty} \theta \left(1 - \theta\right)^\tau \mathbf{v}^i \Pi^\tau \chi \tag{6.5}$$

where $\theta \in (0, 1]$ is a user-specified parameter. $\mathbf{v}^i$, defined as the $1 \times N$ vector $[v_1^i, v_2^i, ..., v_N^i]$, is given by

$$v_j^i = \begin{cases} 1 & if \quad i = j \\ 0 & if \quad i \neq j \end{cases} \tag{6.6}$$

Remark 3.1. *(*Physical Significance of Real Measure*) Assume that the current state of the Markov precess is i, i.e., the state probability vector is $\mathbf{v}^i$. At an instant τ time-steps in the future, the state probability vector is given by $\mathbf{v}^i\Pi^\tau$. Further, the expected value of the characteristic function is given by $\mathbf{v}^i\Pi^\tau\chi$. The measure of state i, as described by Equation (6.5), is the weighted expected value of χ over the all the time-steps in future for a the Markov process that begins in state i. The weights for each time-step ($\theta\left(1 - \theta\right)^\tau$) is a function of the single parameter θ. In addition, the weights form a decreasing geometric series whose sum equals 1. As a result, the measure ν_θ^i is a convex combination of all the elements of the χ vector and $\max_j \chi^j \leq \nu_\theta^i \leq \max_j \chi^j \ \forall i \leq N$.*

The expression for the the measure in Equation (6.5) may be simplified to yield

$$\nu_\theta^i = \theta \mathbf{v}^i \left(I - (1 - \theta)\Pi\right)^{-1} \chi \tag{6.7}$$

The inverse is guaranteed to exist for $\theta \in (0, 1]$.

From Equation (6.5), the measure of all the states denoted by the vector $\nu_\theta = \left[\nu_\theta^1, \nu_\theta^2, ..., \nu_\theta^N\right]^T$ is written as

$$\nu_\theta = \theta \left(I - (1 - \theta)\Pi\right)^{-1} \chi \tag{6.8}$$

Remark 3.2. *(*The effect of θ*) The parameter θ determines the weights ($\theta(1-\theta)^{\tau}$) assigned to the expected characteristic function for time step τ. In particular, θ controls the the rate at which the weights decrease with increasing values of τ. Large values of θ forces the weights to decay rapidly, thereby placing more importance to the characteristic functions of states that are adjacent (connected with fewer hops) to the initial state i. In fact, $\theta = 1$ implies that $\nu_{\theta}^{i} = \chi^{i}$. On the other hand, small values of θ captures the interaction with a large neighborhood of connected states. As $\theta \to 0$, the dependence of on the initial state i is slowly lost (provided Π is irreducible) and all the state converge to the same value of measure.*

3.2 Language-measure-theoretic Problem Formulation

Consider the case of multiple agents performing surveillance in a given region, where the agents are tasked with detection of threats in the region. A typical example of such a threat could be plumes of harmful chemicals that have to be detected. Taking into account the nature of these threats, they may be modeled as local hotspots within the surveillance region. Only a few agents that search areas within the hotspot have a non-zero probability of detecting the threat. The aim of this chapter is to develop a distributed and leader-less algorithm for mobile agents that is able to disseminate the information of a significant threat to other agents that may be far off from the local hotspot. However, if the threat is not significant then agents only in the local vicinity should become aware of that. Previous literature [126] have extensively studied the gradient based approaches for detection of hotspots. These approaches primarily focus on the moving agents towards the hotspots based on distributed estimation of gradients. However, in this application, it is required that all agents should become cognizant of the presence of the threat while operating and monitoring in their own respective local areas. In our approach, the presence of a hotspot does not affect the motion of the agents. Instead, the information states of other agents are updated to reflect the required level of awareness that the agents should posses regarding the threat. The motivation here is to disseminate information away from the local hotspot to the entire population of agents. A language-measure-theoretic approach to this problem is developed in the sequel.

Problem Setup: The mobile-agent network, described in Section 2 is modeled as a PFSA to accommodate the language-measure-theoretic algorithms reviewed above. The set of states Q in the PFSA model corresponds to the set of agents (nodes of the network). Consider the adjacency matrix of the mobile-agent network after the expiry of the message lifetime L_m; hence, the adjacency matrix will have ones in the ij^{th} position if agent i communicates with agent j in the period of L_m, otherwise the matrix element will be zero. All the diagonal elements of the adjacency matrix are kept as zeros. Let the graph be denoted as G and the corresponding adjacency matrix (as defined above) be denoted as A. The Laplacian matrix ($\mathcal{L}$) of G is defined as: $\mathcal{L} = D - A$, where, D denotes the degree matrix. Degree matrix D is defined as the diagonal matrix with $D_{ii} = d_i$, where d_i is the degree of node i. Finally, the matrix Π that corresponds to the state transition matrix in the PFSA setting is defined as [127]:

$$\Pi = I - \beta\mathcal{L} \tag{6.9}$$

The parameter β is chosen appropriately such that Π becomes a stochastic matrix and its second largest eigenvalue satisfies the condition $|\lambda_2(\Pi)| < 1$. In the context of Proximity networks, this requirement of keeping Π stochastic is achieved in the following fashion: β is taken as $\beta = \frac{1}{d_{max}+1}$, where, $d_{max} = \max_i(d_i)$. Now, β is a parameter that is chosen off-line and hence so is d_{max}. Therefore to satisfy the above condition on-line, an agent ignores communications with distinct agents beyond d_{max} number of agents within an message lifetime L_m (note, d_{max} is predetermined). However, given the expected degree distribution $P(k, L_m)$, d_{max} is chosen to be large enough such that the probability of the above situation happening is very low, say ϵ (In this chapter, $\epsilon = 0.001$). Note that Π is a symmetric and stochastic (i.e., also doubly stochastic) matrix due to the above construction procedure.

In the present context, hotspots are detected only by agents proximal to them. Detailed detection model will be discussed in Section 4. From this perspective, agents (nodes) have a state characteristic function $\chi : Q \to \{0, 1\}$. While $\chi^i = 1$ denotes that agent i has detected a hotspot, $\chi^i = 0$ denotes otherwise. Agents (nodes) also have real measure values $\nu : Q \to [0, 1]$. In this context, a higher

value of ν^i denotes a higher level of awareness of agent i about a hotspot in the operational area.

Decentralized Strategy: The decentralized strategy proposed here involves synchronous updating of measures of all agents after the expiry of one message lifetime, L_m. Naturally, L_m is also homogeneous in the agent population. Although the global objectives can be achieved through asynchronous updating with heterogeneous distribution of L_m, simpler condition is considered here for the sake of analytical tractability. Correspondingly, if an agent i detects a hotspot at some instant, $\chi^i = 1$ is kept till the next global measure updating even if the agent does not see the hotspot anymore. The slow time scale of measure updating is denoted by τ and it is clear that both ν and χ are functions of τ. According to the definition provided above, the matrix Π is also a function of τ.

The goal here is to develop a decentralized strategy for the mobile-agent population that leads to the overall measure distribution given in [119]. One such decentralized measure updating dynamics with user defined control parameter θ is:

$$\nu_\theta^i|_{\tau+1} = (1-\theta) \sum_{j \in \{i, Nb(i)\}} \Pi_{ij} \nu_\theta^j|_\tau + \theta \chi^i \tag{6.10}$$

where, $Nb(i)$ denotes the set of agents in the neighborhood of agent i i.e., agents that communicate with agent i between τ and $\tau + 1$. However, in the formulation reviewed above, matrix Π and vector χ are time invariant. In the mobile-agent scenario, both are functions of the slow time scale τ. Hence, the corresponding measure updating dynamics will be:

$$\nu_\theta^i|_{\tau+1} = (1-\theta) \sum_{j \in \{i, Nb(i)\}} \Pi_{ij}|_\tau \nu_\theta^j|_\tau + \theta \chi^i|_\tau \tag{6.11}$$

Expansion of the above equation gives:

$$\nu_\theta^i|_{\tau+1} = (1-\theta) \left[(1 - \beta d_i) \nu_\theta^i|_\tau + \sum_{j \in \{Nb(i)\}} \beta \nu_\theta^j|_\tau \right] + \theta \chi^i|_\tau \tag{6.12}$$

The above equation signifies that the self-influence for an agent reduces with increase in its degree. In other words, the more neighbors an agent communicates

to, the less it relies just on its own observation. Vectorially the dynamics can be expressed as:

$$\nu_\theta|_{\tau+1} = (1-\theta)\Pi|_\tau \nu_\theta|_\tau + \theta\chi|_\tau \tag{6.13}$$

Note that the above algorithm can be called a *generalized gossip algorithm* with θ as the generalizing parameter.

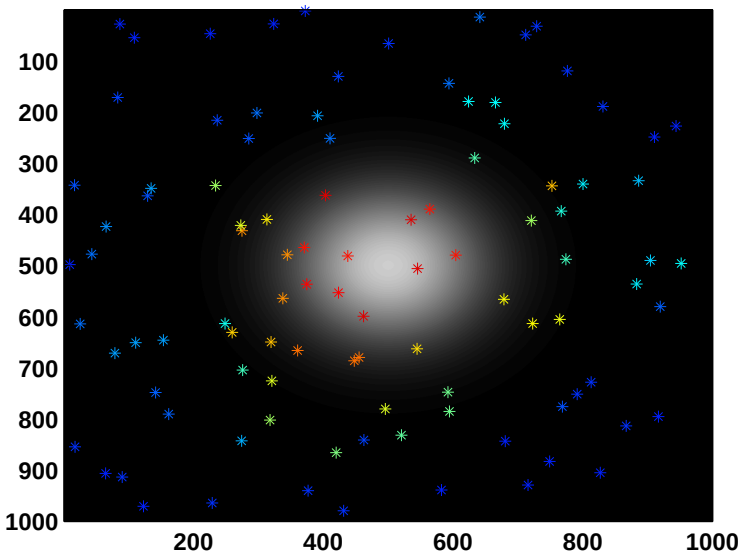


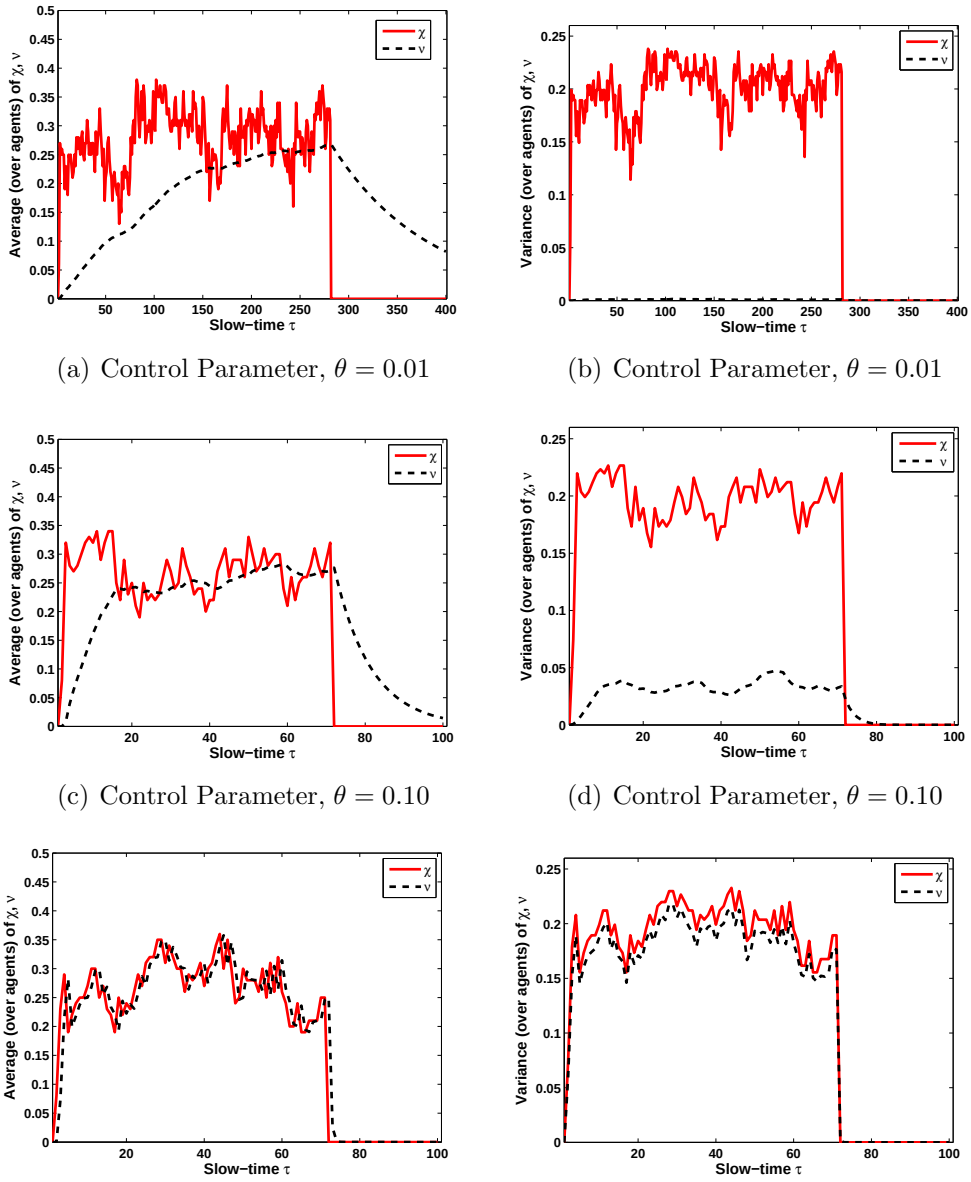
Figure 6.4. Illustration of Hotspot surveillance example; in color, hotspot is shown as the bright patch in the central area, color of agents vary from deep blue to bright red, denoting corresponding agent measure values ranging from 0 to 1.

4 Observations from Simulation Experiments

The mobile multi-agent network and the interaction policy explained in Sections 2 and 3 is considered for an example surveillance and reconnaissance problem. The area of the surveillance region (A), number of mobile agents (N), radius of communication (R), displacement per unit time (x) carry the same values as before. The message lifetime (L_m) is taken to be 30 time units for simulation purpose. A hotspot (i.e. a region where threat exists) is modeled as a map for probability of detection of the threat. The probability of detection, P_D is maximum at the center of the hotspot and decays to zero in a radially symmetric manner; it is character-

ized by two parameters: i) The maximum probability of detection of threat, P_{Dmax} ($= 0.8$ in this study) and ii) the effective radius (r_{hs}) of the circular region within which the detection probability of the threat is greater than 0.5, i.e., agents further than a distance of r_{hs} from the center of the hot-spot have less than 0.5 probability of detecting the threat. For convenience, a hotspot length scale λ is defined as the non-dimensional quantity r_{hs}/L ($= 0.1$ in this study). As described earlier, state characteristic function χ^i of agent i becomes 1 upon detecting a hotspot; clearly, detection depends on the proximity of the agent to the center of the hotspot, i.e., value of P_D at its current location. Otherwise, χ^i remains 0. After the expiry of message lifetime L_m, χ value of an agent resets to 0. Agent measure (ν) values are updated based on the language-measure-theoretic policy described earlier. A typical illustrative example of the operational area is presented in Fig. 6.4.

Experiments are performed with different values of θ. In this chapter, observations regarding convergence of statistical moments (mean and variance) of ν_θ with $\theta = 0.01, 0.10, 0.90$ are presented. Top three plates in Fig. 6.12 presents the time evolution of average (over agents) and the bottom three plates present the corresponding variance (over agents) time-series of χ and ν. Hotspot is switched on at $\tau = 2$ for all experiments and switched off at $\tau = 280$ for $\theta = 0.01$ and at $\tau = 70$ for $\theta = 0.10, 0.90$. First observation (in plates (a), (c), (e)) is that after the appearance of hotspot, average (over agents) ν converges to average (over agents) χ at the steady state. Also, the convergence time decreases with increase in θ. This observation can be explained as follows: It is clear from Eqn. 6.13 that the system dynamics depends on the largest Eigenvalue of $(1-\theta)\Pi|_\tau$. As $\Pi|_\tau$ is a stochastic matrix, Perron-Frobenius theorem ensures that its largest Eigenvalue is 1; thus, largest Eigenvalue of $(1-\theta)\Pi|_\tau$ is $(1-\theta)$. Therefore, it is expected that the convergence time will increase with decrease in θ. Moreover, first order dynamics can be observed in the time evolution of average ν; this can be attributed to the uniqueness of the largest Eigenvalue of Π. Plates (b), (d), (f) show that the steady state variance (over agents) of ν increases with increase in θ; also, $\mathbb{V}_a[\nu] \to 0$ as $\theta \to 0$ and $\mathbb{V}_a[\nu] \to \mathbb{V}_a[\chi]$ as $\theta \to 1$. The dependence of steady-state statistical moments of agent measure on system parameters is analyzed in detail in the following section.



(a) Control Parameter, $\theta = 0.01$

(b) Control Parameter, $\theta = 0.01$

(c) Control Parameter, $\theta = 0.10$

(d) Control Parameter, $\theta = 0.10$

(e) Control Parameter, $\theta = 0.90$

(f) Control Parameter, $\theta = 0.90$

Figure 6.5. Propagation of Global awareness for Hotspot length scale $\lambda = 0.10$ on a mobile-agent network with Message lifetime $L_m = 30$. Plates (a), (c), (e) show the time evolution of average (over agents) of χ and ν and plates (b), (d), (f) show the time evolution of variance (over agents) of χ and ν; Hotspot is switched on at $\tau = 4$ and switched off at $\tau = 267$ for $\theta = 0.01$ and at $\tau = 67$ for $\theta = 0.10, 0.90$.

5 Convergence of Statistical Moments

The convergence result presented here naturally involves expected quantities due to the inherent stochastic nature of the problem. Thus even in steady state, ν_θ will always fluctuate in the slow time-scale due to the fluctuation in Π and χ. However, interesting observations regarding slow time-scale evolution of the system can be made in terms of statistical moments of ν_θ computed over the agent population. In this chapter, both average (over agents), $\mathbb{M}_a[\cdot]$ and variance (over agents), $\mathbb{V}_a[\cdot]$ of ν_θ are considered at steady state. Note, $\nu_\theta|_\tau$ at any slow time instant τ is an N-dimensional vector, where N is the number of agents in the population. Hence, $\mathbb{M}_a[\nu_\theta|_\tau]$ and $\mathbb{V}_a[\nu_\theta|_\tau]$ are respectively scalar average and variance values where $\nu_\theta|_\tau$ is considered as a random variable with N samples. In general, the functions $\mathbb{M}_a[\cdot]$ and $\mathbb{V}_a[\cdot]$ defined on an N dimensional column vector $\mathbf{x} = [x_1, x_2, ..., x_N]^T$ as follows:

$$\mathbb{M}_a(\mathbf{x}) = \frac{1}{n}\mathbf{1}\mathbf{x} = \mathbf{x}^{avg} \tag{6.14}$$

where, $\mathbf{1}$ is a row vector with all elements as 1. After mean subtraction, let the resulting vector be denoted as $\tilde{\mathbf{x}}$, i.e., $\tilde{\mathbf{x}} = \mathbf{x} - \mathbf{x}^{avg}\mathbf{1}^T$. Therefore, $\mathbb{V}_a(\mathbf{x}) = \tilde{\mathbf{x}}^T\tilde{\mathbf{x}}$. Note,

$$\begin{aligned} \tilde{\mathbf{x}}^T\tilde{\mathbf{x}} &= \left[\mathbf{x} - \frac{1}{n}\underline{\mathbf{1}}\mathbf{x}\right]^T\left[\mathbf{x} - \frac{1}{n}\underline{\mathbf{1}}\mathbf{x}\right] \\ &= \mathbf{x}^T\left[I - \frac{1}{n}\underline{\mathbf{1}}\right]\mathbf{x} \end{aligned} \tag{6.15}$$

where, $\underline{\mathbf{1}}$ is an $n \times n$ matrix with all elements as 1. The notations such as, $\mathbf{x}^{avg}$ and $\tilde{\mathbf{x}}$ will be used frequently in the sequel.

5.1 Convergence of Measure Average over Agents

Recall the system dynamics as given in Eqn. 6.13.

$$\nu_\theta|_{\tau+1} = (1-\theta)\Pi|_\tau\nu_\theta|_\tau + \theta\chi|_\tau \tag{6.16}$$

The following equation is obtained by pre-multiplying $\frac{1}{n}\mathbf{1}$ on both sides of Eqn. 6.16.,

$$\nu_\theta^{avg}|_{\tau+1} = (1-\theta)\nu_\theta^{avg}|_\tau + \theta\chi^{avg}|_\tau \tag{6.17}$$

Note, $\mathbf{1}\Pi|_\tau = \mathbf{1}$, as $\Pi|_\tau$ is doubly stochastic. Expanding Eqn. 6.17, one obtains

$$\begin{aligned}\nu_\theta^{avg}|_{\tau+1} = (1-\theta)^{\tau+1}\nu_\theta^{avg}|_0 + \theta\chi^{avg}|_\tau \\ +\theta(1-\theta)\chi^{avg}|_{\tau-1} + \theta(1-\theta)^2\chi^{avg}|_{\tau-2} \\ + \cdots + \theta(1-\theta)^\tau\chi^{avg}|_0\end{aligned} \tag{6.18}$$

Let the ensemble expectation of $\chi^{avg}|_k$ be denoted as $E[\chi^{avg}]\ \forall k$. Note, the expectation removes the time dependency under the Ergodicity assumption. In this case, $E[\chi^{avg}]$ signify the fraction of agents that visit hotspot(s) on average. Therefore, it is evident that with a constant strength of hotspot(s), $E[\chi^{avg}]$ remains constant over time. Taking (ensemble) expectation on both sides of Eqn. 6.18, the following relation is obtained at steady state (as $\tau \to \infty$).

$$\begin{aligned}E[\nu_\theta^{avg}|_\infty] &= \theta[1 + (1-\theta) + (1-\theta)^2 + \cdots]E[\chi^{avg}] \\ &= \theta[1-(1-\theta)]^{-1}E[\chi^{avg}] \\ &= E[\chi^{avg}] \ \text{ for } \theta \in (0,1]\end{aligned} \tag{6.19}$$

Therefore, using the notation of steady-state average (over agents) introduced before, the steady-state expected measure average (over agents) is obtained as:

$$E[\mathbb{M}_a(\nu_\theta)] = E[\mathbb{M}_a(\chi)] \tag{6.20}$$

Convergence of average measure to average χ implies that at steady state, sum of χ values over agents is same as the sum of ν values over agents. In general, the physical significance is that the detection decision of hotspot by few agents is getting redistributed as awareness over a (possibly) larger number of agents, where the total awareness measure is conserved. From this perspective, it is interesting to know the nature of measure distribution in the population and measure variance (over agents) provides an insight in this aspect. For example, an extreme case would be when measure variance is zero, that is all agents have the same measure

value and it is equal to the average measure value of the population. In literature, this scenario is known as *consensus*. The opposite extreme case would be when there is no awareness propagation; only those agents have nonzero measure values that have detected hotspot(s) (i.e., have nonzero χ values). The measure variance will be equal to the variance of χ in this case and the hotspot(s) can be localized very well following the measure distribution due to a sharp localization gradient. Thus, measure distribution essentially dictates a tradeoff between *Propagation Radius* and *Localization Gradient* and variance of ν over agents quantifies the position of the system in this tradeoff scale. But the more interesting part is that the user defined parameter θ does not only control the temporal dynamics, it also controls the measure distribution among agents. In the sequel, a few crucial assumptions are discussed first that are required variance analysis.

5.2 Time scales of Network Evolution and Agent State Dynamics

It is evident from the discussion till now that there exists two fundamentally different time-scales, one related to network evolution and the other related to agent state dynamics. The analytical developments in the sequel will be presented for two special cases of relations between these two time-scales, namely: (i) Congruous Time-Scale (CTS) case and (ii) Disparate Time-Scale (DTS) case.

Congruous Time-Scale case: The two time-scales are equivalent in this case, which means at each slow-time epoch τ (when the agent measures are updated), the system has an independent state transition matrix Π as well as an independent state characteristic vector χ. More formally, the following *assumptions* are made under the CTS case.

- By problem setup, Π at any slow-time epoch depends on the mobility characteristics of the agent population and the message life L_m, neither of which is affected by the presence of hotspot(s). On the other hand, vector χ at any slow time instant captures the information regarding hotspot detection by agents irrespective of inter-agent communication. Hence, it is assumed that $\Pi|_i$ and $\chi|_k$ are independent $\forall i, k$.

- In this setup, agent motion is a fast time-scale dynamics and Π captures

the inter-agent communication characteristics (due to agent motion) for a window of fast time-scale. Now, for a large enough window (i.e., large value of L_m), it can be assumed that the fast time-scale mobility correlation dies out. As a consequence, $\Pi|_i$ and $\Pi|_j$ are independent $\forall i, j$.

- The agents move fast enough (or in other words, the hotspot length scale is reasonably small compared to the scale of agent motion) such that $\chi|_i$ and $\chi|_j$ are independent $\forall i, j$.

Note, the first two assumptions are feasible under fairly general conditions whereas the third one requires a special condition of agent mobility.

Disparate Time-Scale case: In this case, the two time-scales are very different such that, the network evolution (the slow dynamics) and the agent state updating (the fast dynamics) can be treated independently as it is done in the *Singular Perturbation theory*. This leads to the assumption that Π and χ remain time-invariant over the course of transience in agent state dynamics, i.e., agent measures converge before there is a change in Π and χ.

5.3 Convergence of Measure Variance over Agents

First, the analysis is performed under the CTS assumptions. The recursive relation in Eqn. 6.13 can be expanded as:

$$\begin{aligned}\nu_\theta|_{\tau+1} = (1-\theta)^{\tau+1}[\Pi|_\tau \Pi|_{\tau-1} \cdots \Pi|_0]\nu_\theta|_0 + \theta\chi|_\tau \\ +\theta(1-\theta)\Pi|_\tau \chi|_{\tau-1} + \theta(1-\theta)^2 \Pi|_\tau \Pi|_{\tau-1}\chi|_{\tau-2} \\ + \cdots + \theta(1-\theta)^\tau \Pi|_\tau \Pi|_{\tau-1} \cdots \Pi|_1 \chi|_0 \end{aligned} \tag{6.21}$$

Let the ensemble expectation of $\Pi|_i$ be denoted as $E[\Pi]$ $\forall i$ and the ensemble expectation of $\chi|_k$ be denoted as $E[\chi]$ $\forall k$. With these notations and only the first two assumptions for CTS case, (ensemble) expectation is taken on both sides of Eqn. 6.21 and the following relation is obtained as $\tau \to \infty$.

$$\begin{aligned} E[\nu_\theta|_\infty] &= \theta[1 + (1-\theta)E[\Pi] + (1-\theta)^2 E[\Pi]^2 + \cdots]E[\chi] \\ &= \theta[I - (1-\theta)E[\Pi]]^{-1}E[\chi] \quad \text{for } \theta \in (0,1] \end{aligned} \tag{6.22}$$

It is clear that $E[\Pi]$ depends on the agent mobility characteristics and $E[\chi]$ depends on the strength of hotspot(s). Moreover, under the Ergodicity assumption, $E[\Pi]$ and $E[\chi]$ can be approximated by the time expectation of Π and χ over the slow time-scale. Such estimated time expected quantities are denoted by $\hat{E}_\tau[\Pi]$ and $\hat{E}_\tau[\chi]$.

For variance calculation, consider post-multiplication of $\mathbf{1}^T$ on both sides of Eqn. 6.17,

$$\begin{aligned} \nu_\theta^{avg}|_{\tau+1}\mathbf{1}^T &= (1-\theta)\nu_\theta^{avg}|_\tau\mathbf{1}^T + \theta\chi^{avg}|_\tau\mathbf{1}^T \\ \Rightarrow \nu_\theta^{avg}|_{\tau+1}\mathbf{1}^T &= (1-\theta)\nu_\theta^{avg}|_\tau\Pi|_\tau\mathbf{1}^T + \theta\chi^{avg}|_\tau\mathbf{1}^T \end{aligned} \tag{6.23}$$

The above equation presents the mean dynamics for the system. Now, the following equation is obtained by subtracting the mean dynamics (in Eqn. 6.23) from the system equation (in Eqn. 6.16).

$$\tilde{\nu}_\theta|_{\tau+1} = (1-\theta)\Pi|_\tau\tilde{\nu}_\theta|_\tau + \theta\tilde{\chi}|_\tau \tag{6.24}$$

For calculation of variance (over agents),

$$\begin{aligned} (\tilde{\nu}_\theta|_{\tau+1})^T(\tilde{\nu}_\theta|_{\tau+1}) &= (1-\theta)^2(\tilde{\nu}_\theta|_\tau)^T(\Pi|_\tau)^T(\Pi|_\tau)(\tilde{\nu}_\theta|_\tau) \\ &+\theta^2(\tilde{\chi}|_\tau)^T(\tilde{\chi}|_\tau) + 2\theta(1-\theta)(\tilde{\nu}_\theta|_\tau)^T(\Pi|_\tau)^T(\tilde{\chi}|_\tau) \end{aligned} \tag{6.25}$$

Taking ensemble expectation (given $\tilde{\nu}_\theta|_\tau$) on both sides,

$$\begin{aligned} E[(\tilde{\nu}_\theta|_{\tau+1})^T(\tilde{\nu}_\theta|_{\tau+1})|\tilde{\nu}_\theta|_\tau] = \\ (1-\theta)^2(\tilde{\nu}_\theta|_\tau)^T E[(\Pi|_\tau)^T(\Pi|_\tau)](\tilde{\nu}_\theta|_\tau) + \\ \theta^2 E[(\tilde{\chi}|_\tau)^T(\tilde{\chi}|_\tau)] + 2\theta(1-\theta)(\tilde{\nu}_\theta|_\tau)^T E[(\Pi|_\tau)^T]E[(\tilde{\chi}|_\tau)] \end{aligned} \tag{6.26}$$

Since all the agents perform a random walk motion, they are equally likely to visit the hot spot. This implies that $E[(\tilde{\chi}|_\tau)] = 0$. Furthermore,

$$(1-\theta)^2(\tilde{\nu}_\theta|_\tau)^T E[(\Pi|_\tau)^T(\Pi|_\tau)](\tilde{\nu}_\theta|_\tau) \geq 0 \tag{6.27}$$

Therefore, for the lower bound

$$\begin{aligned} E[(\tilde{\nu}_\theta|_{\tau+1})^T(\tilde{\nu}_\theta|_{\tau+1})|\tilde{\nu}_\theta|_\tau] &\geq \theta^2 E[(\tilde{\chi}|_\tau)^T(\tilde{\chi}|_\tau)] \\ \Rightarrow E[(\tilde{\nu}_\theta|_{\tau+1})^T(\tilde{\nu}_\theta|_{\tau+1})] &\geq \theta^2 E[(\tilde{\chi}|_\tau)^T(\tilde{\chi}|_\tau)] \end{aligned} \tag{6.28}$$

The expected (steady-state) variance can expressed as: $E\left[\mathbb{V}_a[\nu_\theta]\right] = E[(\tilde{\nu}_\theta|_{\tau+1})^T(\tilde{\nu}_\theta|_{\tau+1})]$. Using similar notation for χ, one has:

$$\frac{E\left[\mathbb{V}_a[\nu_\theta]\right]}{E\left[\mathbb{V}_a[\chi]\right]} \geq \theta^2 \tag{6.29}$$

Note, by construction $\tilde{\nu}_\theta|_\tau) \perp \mathbf{1}^T$ [128]. Also, $\mathbf{1}$ is the stationary vector (left eigenvector corresponding to the unity eigenvalue) of a doubly stochastic matrix. Therefore,

$$(\tilde{\nu}_\theta|_\tau)^T E[(\Pi|_\tau)^T(\Pi|_\tau)](\tilde{\nu}_\theta|_\tau) \leq \Lambda_2(\tilde{\nu}_\theta|_\tau)^T(\tilde{\nu}_\theta|_\tau) \tag{6.30}$$

where, $\Lambda_2 = \lambda_2\left(E[(\Pi|_\tau)^T(\Pi|_\tau)]\right)$. Therefore, for the upper bound

$$\begin{aligned} E[(\tilde{\nu}_\theta|_{\tau+1})^T(\tilde{\nu}_\theta|_{\tau+1})|\tilde{\nu}_\theta|_\tau] &\leq (1-\theta)^2\Lambda_2(\tilde{\nu}_\theta|_\tau)^T(\tilde{\nu}_\theta|_\tau) \\ &\quad +\theta^2 E[(\tilde{\chi}|_\tau)^T(\tilde{\chi}|_\tau)] \\ \Rightarrow E[(\tilde{\nu}_\theta|_{\tau+1})^T(\tilde{\nu}_\theta|_{\tau+1})] &\leq (1-\theta)^2\Lambda_2 E[(\tilde{\nu}_\theta|_\tau)^T(\tilde{\nu}_\theta|_\tau)] \\ &\quad +\theta^2 E[(\tilde{\chi}|_\tau)^T(\tilde{\chi}|_\tau)] \end{aligned} \tag{6.31}$$

At steady-state, $E\left[\mathbb{V}_a[\nu_\theta]\right] = E[(\tilde{\nu}_\theta|_{\tau+1})^T(\tilde{\nu}_\theta|_{\tau+1})] = E[(\tilde{\nu}_\theta|_\tau)^T(\tilde{\nu}_\theta|_\tau)]$. Therefore,

$$\begin{aligned} E\left[\mathbb{V}_a[\nu_\theta]\right]\left[1-(1-\theta)^2\Lambda_2\right] &\leq \theta^2\mathbb{V}_a[\chi] \\ \Rightarrow \frac{E\left[\mathbb{V}_a[\nu_\theta]\right]}{E\left[\mathbb{V}_a[\chi]\right]} &\leq \frac{\theta^2}{1-(1-\theta)^2\Lambda_2} \end{aligned} \tag{6.32}$$

Figure 6.6 presents the plot of variance ratio $\frac{\mathbb{V}_a[\nu_\theta]}{\mathbb{V}_a[\chi]}$ with θ for three possible values of Λ_2. Note, the upper bound coincides with the lower bound for $\Lambda_2 = 0$. An experimental verification is also presented in Fig. 6.7 that shows the experimental data to closely follow the upper bound for the particular case. While the expected degree of the network is kept as 3, high velocity is assumed for agents to achieve the conditions described in the CTS assumptions.

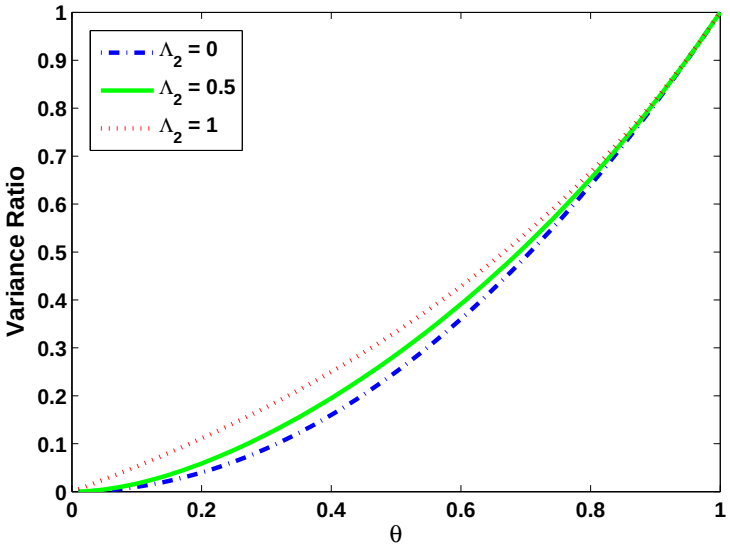


Figure 6.6. Variance Ratio as a function of θ and $\Pi|_\tau$ under CTS assumptions

Next, the analysis is performed under the DTS assumptions. As discussed earlier, Π and χ are not functions of τ in this case. From Eqn. 6.21, as $\tau \to \infty$, one has:

$$\begin{aligned}\nu_\theta|_\infty = \theta\chi + \theta(1-\theta)\Pi\chi + \theta(1-\theta)^2\Pi^2\chi \\ +\theta(1-\theta)^3\Pi^3\chi\cdots\end{aligned} \tag{6.33}$$

The following equation is obtained by subtracting the mean dynamics from Eqn. 6.33.

$$\begin{aligned}\tilde{\nu}_\theta|_\infty = \theta\tilde{\chi} + \theta(1-\theta)\Pi\tilde{\chi} + \theta(1-\theta)^2\Pi^2\tilde{\chi} \\ +\theta(1-\theta)^3\Pi^3\tilde{\chi}\cdots\end{aligned} \tag{6.34}$$

Using the above equation, the measure variance over agents is calculated as:

$$\begin{aligned}\mathbb{V}_a[\nu_\theta] = \theta^2\tilde{\chi}^T\tilde{\chi} + \theta^2(1-\theta)\tilde{\chi}^T\Pi\tilde{\chi} + \theta^2(1-\theta)\tilde{\chi}^T\Pi^T\tilde{\chi} \\ +\theta^2(1-\theta)^2\tilde{\chi}^T\Pi^T\Pi\tilde{\chi} + \theta^2(1-\theta)^2\tilde{\chi}^T\Pi^2\tilde{\chi} \\ +\theta^2(1-\theta)^2\tilde{\chi}^T(\Pi^2)^T\tilde{\chi}\cdots\end{aligned} \tag{6.35}$$

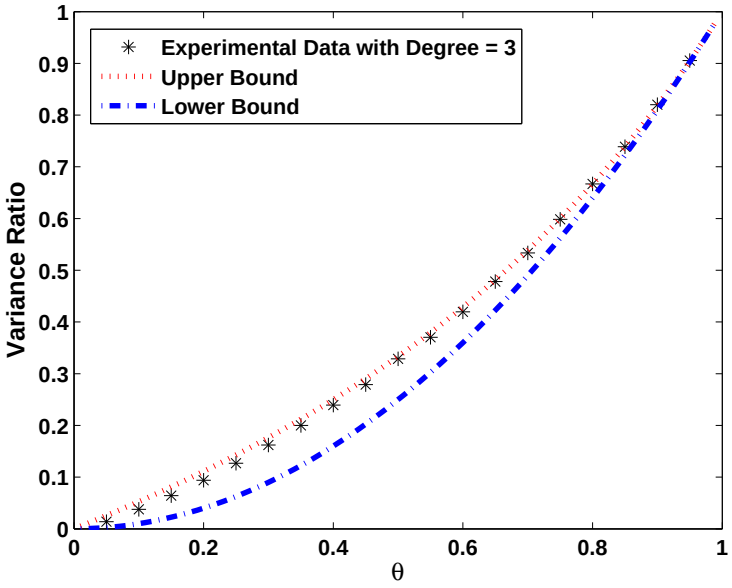


Figure 6.7. Experimental Verification of Variance Ratio Bounds under CTS assumptions

As Π is symmetric, one has:

$$\begin{aligned}\mathbb{V}_a[\nu_\theta] = \theta^2\tilde{\chi}^T\tilde{\chi} + 2\theta^2(1-\theta)\tilde{\chi}^T\Pi\tilde{\chi} \\ +3\theta^2(1-\theta)^2\tilde{\chi}^T\Pi^2\tilde{\chi}\cdots\end{aligned} \tag{6.36}$$

Since Π^k s are positive definite for $k \in \mathbb{N}$, the lower bound is obtained as

$$\frac{\mathbb{V}_a[\nu_\theta]}{\mathbb{V}_a[\chi]} \geq \theta^2 \tag{6.37}$$

Using the same logic as before, it is evident that $\tilde{\chi}^T\Pi^k\tilde{\chi} \leq \lambda_2(\Pi^k)\tilde{\chi}^T\tilde{\chi}$ for $k \in \mathbb{N}$. Also, $\lambda_2(\Pi^k) = \lambda_2^k(\Pi)$ and $\lambda_2(\Pi)$ is denoted simply as λ_2 in the sequel. Therefore,

$$\begin{aligned}\mathbb{V}_a[\nu_\theta] \leq \theta^2\mathbb{V}_a[\chi] + 2\theta^2(1-\theta)\lambda_2\mathbb{V}_a[\chi] \\ +3\theta^2(1-\theta)^2\lambda_2^2\mathbb{V}_a[\chi]\cdots\end{aligned} \tag{6.38}$$

By calculating the infinite sum, the upper bound is obtained as

$$\frac{\mathbb{V}_a[\nu_\theta]}{\mathbb{V}_a[\chi]} \leq \frac{\theta^2}{[1-(1-\theta)\lambda_2]^2} \tag{6.39}$$

The upper bound for the variance ratio calculated above is valid for a particular Π. Figure 6.8 presents the plot of variance ratio $\frac{\mathbb{V}_a[\nu_\theta]}{\mathbb{V}_a[\chi]}$ with θ for three possible

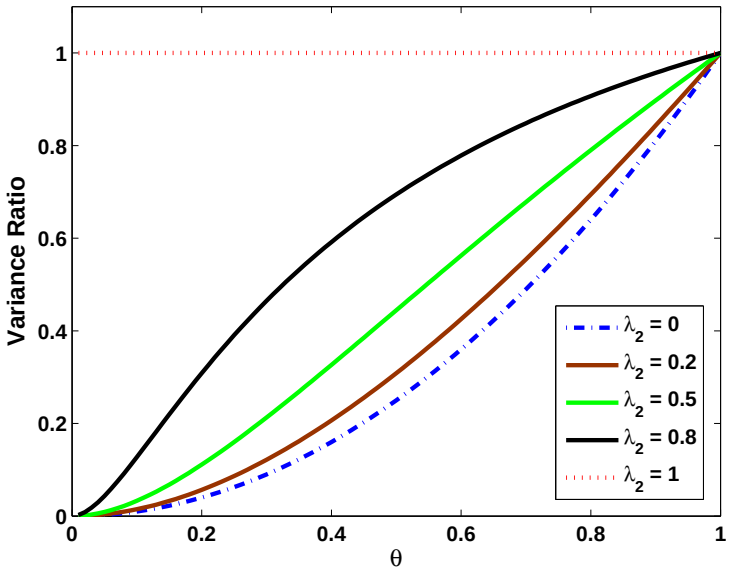


Figure 6.8. Variance Ratio as a function of θ and Π under DTS assumptions

values of λ_2. Note, the upper bound coincides with the lower bound for $\lambda_2 = 0$. Experimental verification is presented in Fig. 6.9 that shows the experimental data for two cases with expected degree of the network as 3 and 7. Agent velocity is kept very low (but not zero) to achieve the conditions described in the DTS assumptions.

It is observed in both cases that upper bound and lower bound coincide as θ approaches extreme values, 0 or 1 and as seen in Section 4, $\mathbb{V}_a[\nu_\theta] \to 0$ as $\theta \to 0$ and $\mathbb{V}_a[\nu_\theta] \to \mathbb{V}_a[\chi]$ as $\theta \to 1$. In other words, the agent population approaches *consensus* as $\theta \to 0$. In this case, although the entire population becomes aware of the hotspot(s), there is no localization gradient as every agent has same measure value. On the other hand, with $\theta \to 1$, the localization gradient improves at the cost of propagation radius. In general, $\mathbb{V}_a[\nu_\theta]$ decreases with decrease in θ. The other system component affecting the variance ratio is the Π matrix. In both CTS and DTS cases, this effect is realized through second largest eigenvalue of Π. Lower second largest eigenvalue of Π signifies more connectivity among agents. This fact explains the decrease in variance ratio with decrease in the second largest

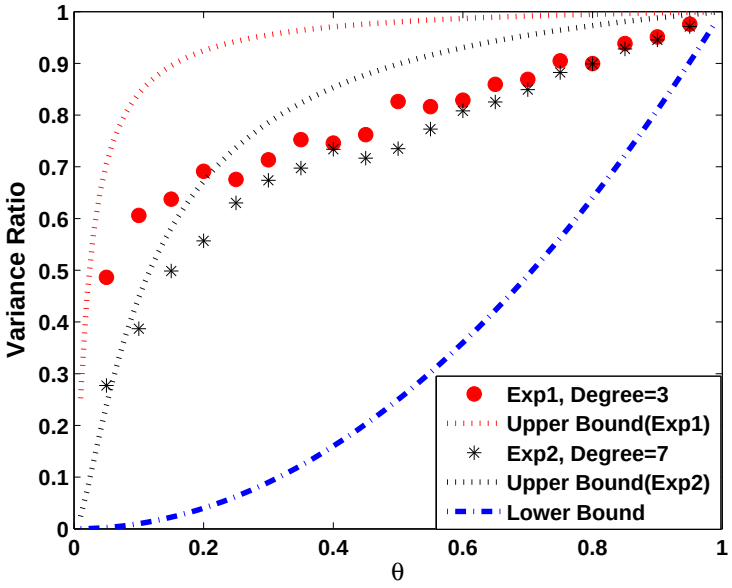


Figure 6.9. Experimental Verification of Variance Ratio Bounds under DTS assumptions

eigenvalue. It is evident that second largest eigenvalue is a function of degree of the network. However, degree certainly is not the only network parameter that controls the system as it is observed in the experimental data for CTS and DTS cases. Therefore, apart from network degree, compatibility between time-scales of network evolution and agent state dynamics also plays a key role in determining the system characteristics.

6 Nonlinear Agent Interaction Dynamics: Binary Decisions with Externalities Policy

Although simple, Binary decisions with externalities, a decentralized agent interaction policy can successfully model complex real life phenomena, such as cascading failure in power grids and opinion spreading in social networks [123]. Also, this policy is suitable for mobile agent networks, where lack of homogeneous neighborhood structure is inevitable (as opposed to regular lattices, where majority vote or random-field Ising model policies may be more effective). The following subsections discusses policy and necessary conditions for global cascading in the

mobile-agent networks.

6.1 Agent Interaction Policy

The state of every agent can either be -1 or $+1$. Let an agent A observes the states of k other agents that come within its communication radius during the lifetime L_m of its one message and if A comes across one agent multiple times, then it records the latest state information of that agent. Suppose, i of such k agents are observed to be in state $+1$. Then the state updating policy of A is governed by a threshold parameter ϕ such that, if $\phi < i/k$, then A will assume state $+1$ for next L_m time units, otherwise it will assume state -1, irrespective of its current state. Although, in general ϕ can be heterogeneous, i.e., different ϕ for different agents, it is assumed to be uniform in this chapter. It is shown in [123] that for static graphs, i) the critical value of ϕ below which global cascade occurs and ii) the expected size of the cascade depend only upon the degree distribution (expected degree or coordination number) and the distribution of ϕ among agents. Formally, to see the dynamics of global cascade, one may start with a population of agents where all are in state -1. Then states for a small number of randomly selected agents (known as seeds) are perturbed to $+1$ at time, $t = 0$. After that, the population is allowed to evolve according to the rules described above. A global cascade condition is said to have triggered if at least a threshold fraction C of the total agent population change state to $+1$. C is taken as 0.9 in this study. Usually, in theoretical analysis (as performed in [123]), the underlying network is assumed to be sufficiently large (for example, with 10^4 agents) where as the number seeds is considered to be sufficiently small (for example, approximately three orders of magnitude less than the total number of agents). However, mobile-agent applications demand analysis of large seeding where, unlike its counterpart, cascading will depend on the seed size. In that case, it is observed that the dependence of the cascading condition on the seed size is very small. Although, these assumptions may be relevant in social network or epidemiology studies, in the case of mobile-agent engineering applications, they may not be very accurate. Hence, effects of large number of seeding agents on a relatively smaller sized network need to be studied. However, these realistic considerations make theoretical analysis more complicated and in

some cases intractable.

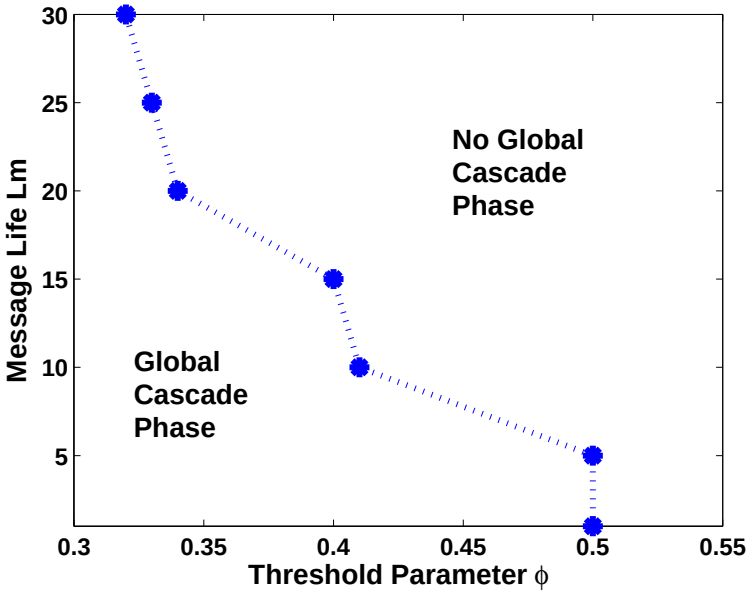


Figure 6.10. Phase Diagram plot of L_m vs. ϕ for the Global Cascading phenomenon in Mobile-agent networks

6.2 Global Cascade with Large Seed

Based on the motivation provided above, this subsection discusses the phenomenon of global cascading with large number of seeding agents. In contrast with the case of very large network with small seed, the global cascading phenomenon depends on the size of the seed when the relative seed size is high. Therefore, the phase boundary in the $< k >$ vs. ϕ phase diagram will be a function of the size of the initial seed. In the present simulation architecture, presence of a certain expected size of seed is realized in the following fashion. The agent state dynamics starts with all agents in state -1 and the agent states evolve according the rule described earlier. Suppose the desired fraction of seed is 0.02, i.e., 2% of agents should have state $+1$. To achieve the objective dynamically, a small modification is done in the agent state updating policy; after the state updating decision is taken by an agent following the threshold based neighborhood interaction rule, there remains a finite probability of 0.02, with which the agent can go against the decision made by its neighborhood influence. It should be clear that this policy modification

will dynamically generate an expected seed size of 2% even when the agent state dynamics starts with all agents in state -1. However, this modification has more implications from an engineering perspective that will be explained in Section 7. Also, as L_m is considered to be the sole parameter to dictate the value of $< k >$, given the other relevant network parameters constant, an L_m vs. ϕ phase diagram is constructed here instead of the $< k >$ vs. ϕ phase diagram. Monte-Carlo simulations have been performed (with a 2% seed) to construct the phase diagram, i.e., to identify the critical ϕ values for different L_m values. For each Monte-Carlo run with specific values of L_m and ϕ, the onset of global cascading is monitored. Probability of global cascading is calculated from the observations of 50 such runs for each L_m-ϕ combination. At a given value of L_m, the critical ϕ^* is considered to be the highest value of ϕ for which the probability of global cascading is above a threshold value of 0.95. The phase diagram is shown in Fig. 6.10 and it can be observed that it is characteristically very similar to the phase diagram for small seed case with very large static networks, shown in [123].

7 Decision Propagation Application

This section presents an application of global cascade phase transition in distributed decision making and propagation of the decision. The problem setup remains the same as before. Again, the motivation here is to disseminate information away from the local hotspot to the entire population of agents by applying the notion of global cascade phase transition to effect decision making and its propagation though out the network.

7.1 Problem Description

The area of the surveillance region (A), number of mobile agents (N), radius of communication (R), displacement per unit time (x) carry the same meaning/values as before. Also, message lifetime (L_m), and the threshold parameter (ϕ) is same as defined earlier. A hotspot (i.e. a region where threat exists) is modeled as a map for probability of detection of the threat. The probability of detection, P_D is maximum at the center of the hotspot and decays to zero in a radially symmetric

manner; it is characterized by two parameters: i) The maximum probability of detection of threat, P_{Dmax} (= 0.8 in this study) and ii) the effective radius (r_{hs}) of the circular region within which the detection probability of the threat is greater than 0.5, i.e., agents further than a distance of r_{hs} from the center of the hot-spot have less than 0.5 probability of detecting the threat. For convenience, a hotspot length scale λ is defined as the non-dimensional quantity r_{hs}/L. The two states $\{+1, -1\}$ are used to indicate the information acquired by the agents. An agent in state $+1$ is said be aware of the existence of a threat/hotspot somewhere in the surveillance regions (state of alert), whereas state -1 implies that the agent believe that there are no threats in the entire region. After the expiry of message lifetime L_m, the state of an agents may be updated based on the following rules.

- Agent state becomes $+1$ upon detecting a threat; clearly, detection depends on the proximity of the agent to the center of the hot-spot (P_D value).
- States are updated based on the threshold parameter ϕ and message lifetime L_m (see Section 6).
- There remains a finite probability of 0.02 as before, with which the agent can go against the decision made by its neighborhood or hotspot influence.

These state update rules exhibit global cascades that *propagate* information from the local hotspot. Such a cascade would certainly be dependent on the size of the hotspot and this can be used to design a decision propagating mechanism that is robust to spurious noise and false alarms but responds to hotspots of significant influence. For example, only a few agents (at a time) can move very close to the center of a small hotspot and possibly change their states to $+1$. However, soon after they move out of the hotspot influence zone, their states revert back to -1 due to interactions with other agents that are in state -1. As a consequence, there is no decision propagation and the agents far from the hotspot do not become aware of the existence of a threat. In contrast, a large hotspot may be visited by higher number of agents at once. These agents exit the hotspot with state $+1$ and have the combined dominance to influence other agents to change their states to $+1$, thereby propagating information regarding the presence of a threat to the entire network. Thus, this application exploits the large seed effects, where the

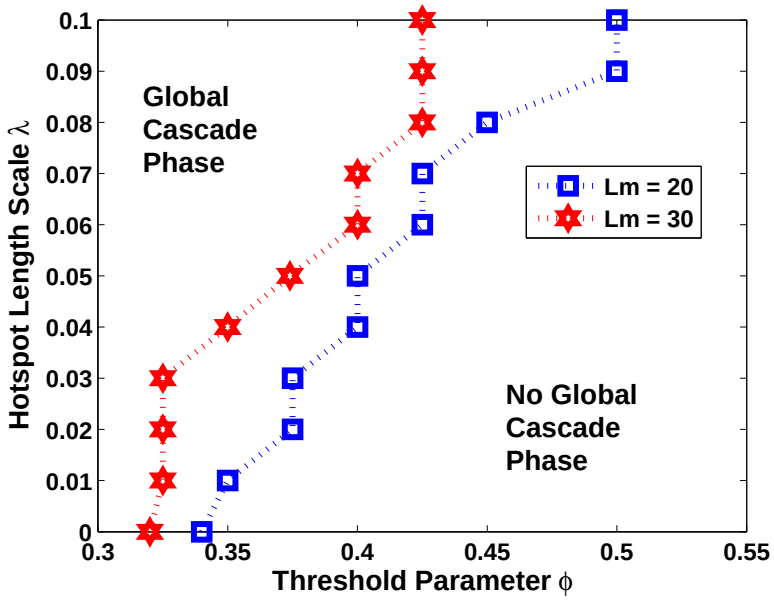


Figure 6.11. Phase Diagram plot of λ vs. ϕ for the Global Cascading phenomenon in Mobile-agent networks with different L_m

probability of occurrence of a global cascade is determined by the initial fraction of agents in state +1.

7.2 Results and Discussion

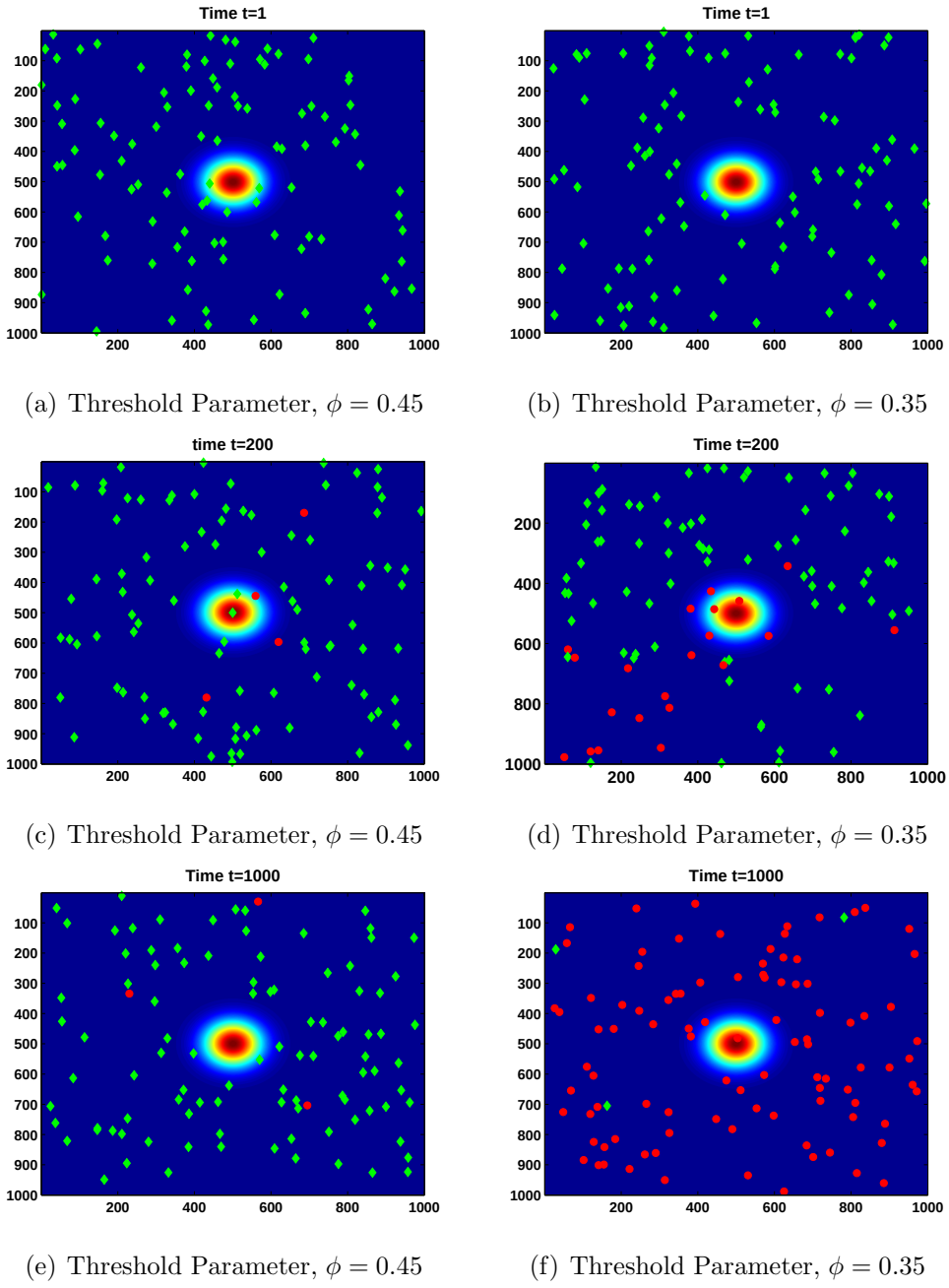
As evident from above, there exist a critical length scale λ beyond which the all the agents become cognizant of the hotspot. The choice of ϕ and L_m determines the value of this critical length scale. Since a larger value of ϕ decreases the propensity for the network to undergo a cascade, a larger hotspot is required to provide sufficient seeds to initiate the propagation. Additionally, a longer message lifetime implies a higher value of the expected degree of the network. As a consequence, a larger seed/hotspot is required for a global cascade. Figure 6.11 shows the λ vs. ϕ phase diagram for $L_m = 20, 30$. This phase diagram enables the user to choose values of ϕ and L_m to distributively make an implicit decision to propagate knowledge about the hotspot. Six plates in Fig. 6.12 show the effects of ϕ on the global cascade for a hotspot of a given length scale, $\lambda = 0.05$ on a network with $L_m = 30$. The upper three plates show the time transitions of the agent states for $\phi = 0.45$. No global cascading is observed as ϕ is too large for a hotspot of this

length scale. However, if ϕ is reduced to 0.35, a global cascade may be observed, as seen in the bottom three plates of Fig. 6.12.

Remark 7.1. *It should be noted that the global cascade phenomenon is irreversible, implying that an already alarmed network (corresponding to 90% of agents in $+1$ state) would not revert back to an unalarmed state (corresponding to 90% of agents in -1 state) on its own even when the threat is over. The address this issue, an increase of the global parameter ϕ is suggested to reset the network from alarmed to unalarmed state. The finite probability (0.02 in this case) of agents to go against the decision made by neighborhood or hotspot influence will ensure a minimum seed size for this transition.*

8 Summary, Conclusions and Future work

This chapter addresses the issue of distributed decision propagation in a mobile-agent network environment for surveillance and reconnaissance. Concepts of proximity network is used for model formulation and computation of the expected degree distribution. Among different network parameters, message lifetime is identified to be the critical parameter to control the expected network topology. Recently developed language-measure-theoretic approach is used for modeling the agent interaction dynamics. A completely decentralized implementation of this algorithm is shown to be useful for propagation of global awareness regarding a local hotspot in the operational area. Analytical results have been obtained for convergence of measure (awareness level) distribution in the agent population. A (user-defined) critical parameter θ, which has both temporal effects (e.g., convergence time) and ensemble effects (e.g., the measure distribution characteristics in the agent population), controls the tradeoff between the propagation radius and the localization gradient. In this setting, *consensus* can be achieved as $\theta \to 0$. Two cases (CTS and DTS) relating the time-scales network topology and agent interaction are presented and verified by numerical simulation on a test bed for a typical example problem. In this algorithm, the system is reset automatically upon removal of a hotspot. Another advantage of this approach is that it naturally extends to multiple hotspot scenarios; future work will involve detailed investigation with multiple hotspots. Few other future research directions are: (i) Analytical evaluation of



(a) Threshold Parameter, $\phi = 0.45$

(b) Threshold Parameter, $\phi = 0.35$

(c) Threshold Parameter, $\phi = 0.45$

(d) Threshold Parameter, $\phi = 0.35$

(e) Threshold Parameter, $\phi = 0.45$

(f) Threshold Parameter, $\phi = 0.35$

Figure 6.12. Propagation of Global awareness for Hotspot length scale $\lambda = 0.05$ on a Mobile-agent network with Message lifetime $L_m = 30$. Diamond (green in color) shape denotes agents with state -1 and Circle (red in color) shape denotes agents with state $+1$.

the expected characteristics of Π (hence, second largest eigenvalues), given the expected characteristics of the proximity network; (ii) Investigation of scenarios with asynchronous measure updating and heterogeneous message lifetime distribution; (iii) Relaxation of assumptions for variance calculation and identification of the network size-scaling laws; and (iv) Analysis of convergence dynamics/time under generalized gossip framework.

In a different approach, a nonlinear threshold based strategy is also studied for the same problem. Extensive numerical simulations have been performed to characterize the global cascading phase transition phenomenon using the binary decisions with externalities mechanism for such networks. It is found that the threshold parameter in the binary decision policy can be tuned to change the decision propagation nature. Also, the phase transition phenomenon is not independent of the (large) seed size and this fact has been exploited in the application example to show that the decision propagation can be triggered only beyond a user defined threshold threat level. This is an important (contrasting with respect to the linear strategy) observation for this strategy. Furthermore, the multiple hotspot scenario is not a straight forward extension in this case. Hence, this is an important future research direction. Other than that, agent dynamics with more than two possible agent states needs to be investigated.

Chapter 7

Summary, Conclusions and Future Research Directions

The research presented in this dissertation investigates various aspects of autonomous cyber-physical systems (CPSs). As discussed in the introductory chapter, basic research in CPSs can potentially address the issues related to a large class of future application areas that encompass many fields of science and engineering. Although the research works presented here deal with a few specific application areas, the theoretical contributions are rather general and compatible with the big picture. For example, Part-I deals with sensor data interpretation to extract meaningful features for classification and fusion. The techniques developed here are utilized primarily for diagnostic applications. However, these can be used for general decision support applications (e.g., Intelligent Surveillance and Reconnaissance (ISR) missions, Medical diagnostics) as well. On the other hand, Part-II deals with modeling of emergent behavior and distributed decision making in multi-agent complex systems. These concepts are validated on problems of network communication and information dissemination in mobile-agent networks, monitoring hotspots. However, it is evident that these ideas have much larger implications. For example, emergent behaviors are observed in many human-engineered complex systems and the approaches of defining order parameter, intensive parameters presented here can be used to characterize them. The decision propagation algorithms on the other hand can be used for ISR missions, trust establishments in mobile ad hoc networks (MANETs) and decision fusion applications.

In summary, the four primary research issues related to CPSs are identified. They are: *(i) Information processing: From sensor data to knowledge*, *(ii) Modeling and analysis*, *(iii) Distributed sensing, computation and control*, *(iv) Seamless integration between cyber and physical components, also CPS and human users*. While couple of these issues are directly addressed in this dissertation, the two other issues are dealt with as well to some extent. The work presented in Part-I essentially deals with information extraction from sensors. Also, abstraction of physical systems in the cyber domain serve the purpose of integration in a CPS as well. Similarly, in Part-II, the focus has been modeling and distributed control that have some implications for the distributed sensing and computation aspect. The key contributions of this dissertation are delineated in the sequel.

1 Contributions of the Dissertation

1. **Generalized Hilbert transform for time-series data processing**: Class separability information may not be fully reflected in the time domain of a signal. Therefore, time-series preprocessing is required to facilitate feature extraction to enhance classification performance. A fundamental work is presented in this dissertation that generalizes the classical Hilbert transform. Apart from extracting amplitude and phase information, this approach helps in noise rejection and reduction in model order. The proposed concept of generalization of Hilbert transform is tested and validated on experimental data from the nonlinear Duffing system.

2. **Optimization of feature extraction via partitioning**: Partitioning (the first level of abstraction) is extremely crucial for extracting features under the SDF framework. The dissertation proposes a data partitioning procedure to extract low-dimensional features from time series while optimizing the class separability. A trade-off between sensitivity and robustness of classification is achieved through the construction of the appropriate cost functionals. However, the optimization framework and technique presented here are general enough to incorporate other objectives. Multiple application examples are presented to validate the proposed method.

3. **Extraction of spatiotemporal correlation features**: The abstract semantic representation of sensor data (under the SDF framework) enables feature-level fusion of heterogeneous and disparate sensors. Identification of cross-dependencies among different sensors prevents loss of significant information during feature-level information fusion. The hierarchical architecture of this method reduces computational complexity, allowing real-time operability. The techniques are used for the purpose of engine fault diagnosis using multiple sensor responses.

4. **Emergent behavior modeling in communication networks**: A statistical mechanics-inspired emergent behavior modeling approach is developed here. The notion of order parameter and other global intensive parameters (e.g., network analogs of order parameter, temperature and pressure) is introduced. The global order parameter defined here can be used in general as a measure of performance and stability. A comprehensive finite-size scaling analysis has been performed for a particular type of network structure. Phase diagrams are constructed for networks transmitting heterogeneous packet types that lead to a novel approach for controlling heterogeneous packet transmission.

5. **Distributed decision propagation in mobile-agent networks**: This is an important innovation of this dissertation where the tough problem of handling networks with time varying topology has been dealt with. Two distributed agent interaction policies (one linear and one nonlinear) are studied. The linear strategy is essentially a generalized gossip algorithm under the language-measure-theoretic framework. Theoretical contributions are made that lead to the control of the tradeoff between *Propagation Radius* and *Localization Gradient*. The nonlinear strategy is a threshold based policy, known as binary decisions with externalities. In contrast with the linear strategy, the nonlinear one exhibits a useful critical behavior.

2 Future Research Directions

While future research directions specific to different topics are mentioned at the end of previous chapters, this section presents the broad research areas that can emerge as extensions of the work presented in this dissertation.

1. **Multi-dimensional data understanding**: Beyond understanding of one-dimensional time-series, autonomous perception requires analysis of multi-dimensional data. Such information can be multi-dimensional in different ways: either the domain can be multi-dimensional or the range can be multi-dimensional, or both. For example, an image has a two-dimensional domain and a one-dimensional range. In this case, partitioning can still be performed in the range space to generate a two-dimensional symbol image. Research should be performed for efficient compression of such data. Concepts of random fields may be useful in this regard. On the other hand, in a multi-sensor scenario, at each time-instant (one-dimensional domain) the system has a vector response (multi-dimensional range). In this case, efficient algorithms for multi-dimensional partitioning need to be developed.

2. **Formal construction of composite patterns**: From the perspectives of semantic sensor fusion, concepts of atomic and relational patterns have been developed in this dissertation. However, once the patterns are extracted, a straight forward concatenation was used to construct the composite pattern. Therefore, research should be conducted to develop sophisticated techniques for composite pattern construction. Recent developments regarding a vector-space formulation of PFSA [129, 130] show promise in this context. Concepts of Bayesian networks and entity relationship models may be useful as well. Apart from enhanced classification performance, this will lead to efficient data warehousing, visualization and eventually an intelligent decision support mechanism. For example, for situation assessment tasks, when atomic patterns may characterize individual entities (nouns), relational patterns may signify their inter-relationships (verbs) and finally, the composite pattern in that case provides a succinct description of the situation (sentence).

3. **Adaptive feature extraction**: Dynamic adaptation to changing environment and signal conditions is the key to autonomous perception. Adaptation can be made at every stage, i.e., sampling, data preprocessing, feature extraction and classification. In this context, adaptive feature extraction can be performed under the framework of partitioning optimization. Apart from adaptation of feature extraction (i.e., selection of important features), this framework can also accommodate adaptation of preprocessing and classification.

4. **Closing the control-loop in CPS**: Issues of perception and distributed control are dealt with separately in this dissertation. Therefore, the most important future work will be to close the control loop. Research should be done to identify how higher level perceptive knowledge can be used to make the control decisions and how to transform cyber domain control decisions to physical domain actuations. Moreover, better understanding of emergent behaviors can always push the limit of controlling complex systems. Biological inspirations can be extremely relevant for inducing order in human-engineered complex cyber-physical systems [131].

5. **Seamless human-CPS integration**: Recently, the research community is increasingly moving away from the conventional (ultimate) goal of artificial intelligence, i.e., *complete autonomy*. Importance of using human in the loop is getting more and more apparent. However, this integration is found to be a very tough problem. There are primarily two directions of this integration: interpreting human commands by cyber systems which involves natural language processing (NLP). The recent effort from IBM to develop WATSON is a big achievement in this regard. In the other direction, substantial effort is being expended to present cyber knowledge/decisions in an efficient way to human users. This will lead to decision support mechanisms that are extremely important for time-critical situations (e.g., battle field, safety and rescue missions).

Appendix A

Description of C-MAPSS

This appendix briefly describes the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) test-bed [1] that has been developed at NASA upon a commercial-scale two-spool turbofan engine and its control system. Figure A.1 shows a schematic diagram of the commercial aircraft gas turbine engine used in the C-MAPSS simulation test-bed. The gas turbine engine system consists of five major rotating components, namely, fan (F), low pressure compressor (LPC), high pressure compressor (HPC), high pressure turbine (HPT), and low pressure turbine (LPT), as seen in Fig. A.1. Apart from the rotating components, three actuators have been modeled in the simulation system; they are the Variable Stator Vane (VSV), the Variable Bleed Valve (VBV) and the Fuel Pump that controls the fuel flow rate (Wf). HPC, combustor and HPT form the core of the engine model; this subsystem is also referred to as the gas generator. In the turbofan engine, the engine core is surrounded by the fan, LPC in the front and an additional turbine, LPT at the rear; fan, LPC and LPT are mechanically connected by an additional shaft. The fan shaft passes through the core shaft and, due to this type of arrangement, the engine is called a two spool engine. In contrast to gas turbine engines for military aircraft [48, 59], a relatively small part of the incoming air at the engine inlet passes through the fan and continues on into the core compressor and then into the combustor, where it is mixed with fuel and combustion occurs; therefore, this type of engine is known as a high-bypass engine. The hot exhaust gas, called the core airflow, passes through the core and LPT and then exits through the nozzle; and the rest of the incoming air passes through the fan and bypasses,

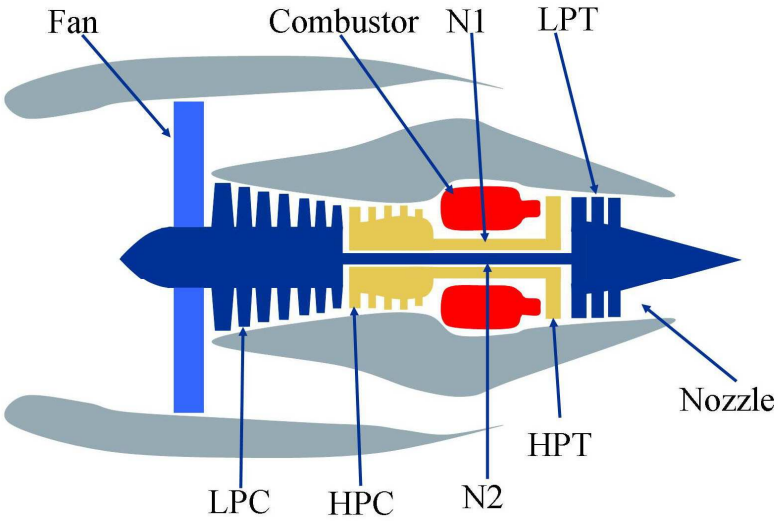


Figure A.1. Gas turbine engine schematic [1]

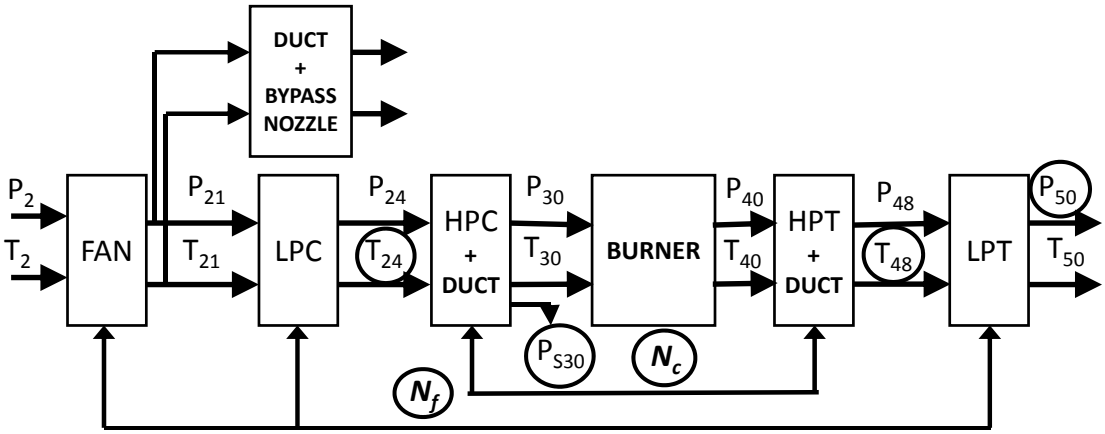


Figure A.2. Schematic diagram of the C-MAPSS engine model with Sensors or flows around the engine.

A gain-scheduled control system is incorporated in the engine system, which consists of speed controllers and limit regulators for engine components.

1. Fan-speed control for a specified TRA;

2. Three high-limit regulators that prevent the engine from exceeding its design limits for core-spool speed, engine-pressure ratio, and HPT exit temperature, and the fourth limit regulator that attempts to prevent the static pressure at the HPC exit from dropping too low;

3. Acceleration and deceleration limiters for the core-spool speed; and

4. A comprehensive logic structure that integrates these control-system components in a manner similar to that used in real engine controllers such that integrator-windup problems are avoided.

To achieve fast execution of simulation runs, the sensors and actuators are approximated to have instantaneous response and no computational time delays. The entire test-bed code is written on Matlab and Simulink platform.

To achieve fast execution of simulation runs, the sensors and actuators are approximated to have instantaneous response, no computational time delays, and no drift and or bias. Given the inputs of TRA, altitude (a) and Mach number (M), the interactively controlled component models at the simulation test-bed compute nonlinear dynamics of real-time turbofan engine operation. Both steady-state and transient operations are simulated in the continuous-time setting. Performance maps are used to provide steady-state representations of the engine's rotating components. Fluid momentum in the bypass duct and the augmentor, mass and energy storage within control volumes, and rotor inertias are also included to model transient operations. The entire test-bed code is written on Matlab and Simulink platform. The engine under consideration produces a thrust of approximately 400,000 N and is designed for operation at altitude (A) from sea level (i.e., 0 m) up to 12,200 m, Mach number (M) from 0 to 0.90, and temperatures from approximately -50°C to 50°C. The throttle resolving angle (TRA) can be set to any value in the range between 0° at the minimum power level and 100° at the maximum power level.

Out of the different types of sensors (e.g., pressure, temperature, and shaft speed) used in the C-MAPSS simulation model, Table A.1 lists those sensors that are commonly adopted in the Instrumentation & Control system of commercial aircraft engines as seen in Fig. A.2. The engine control system makes use of the data from sensors P_2, Ps_{30}, T_{48}, N_f and N_c as feedback signals.

In the current configuration of the C-MAPSS simulation test-bed, there are 13 component level health parameter inputs, namely, efficiency parameters (ψ), flow parameters (ζ) and pressure ratio modifiers, that simulate the effects of faults and/or degradation in the engine components. Ten, out of these 13 health pa-

Table A.1. Sensor Suite for the Engine System

Sensors	Description
P_2	Fan inlet pressure
T_2	Fan inlet temperature
P_{24}	LPC exit/ HPC inlet pressure
T_{24}	LPC exit/ HPC inlet temperature
Ps_{30}	HPC exit static pressure
T_{30}	HPC exit/ Burner inlet temperature
T_{48}	HPT exit temperature
N_f	Fan spool speed
N_c	Core spool speed

rameters, are selected to modify efficiency (η) and flow (ϕ) that are defined [132] as:

- $\eta \triangleq$ Ratio of actual enthalpy and ideal enthalpy changes.
- $\phi \triangleq$ Ratio of rotor tip and axial fluid flow velocities.

For the engine's five rotating components F, LPC, HPC, LPT, and HPT, the ten respective efficiency and flow health parameters are: (ψ_F, ζ_F), (ψ_{LPC}, ζ_{LPC}), (ψ_{HPC}, ζ_{HPC}), ((ψ_{HPT}, ζ_{HPT}), and (ψ_{LPT}, ζ_{LPT}). An engine component C is considered to be in nominal condition if both ψ_C and ζ_C are equal to 1 and fault can be injected in the component C by reducing the values of ψ_C and/or ζ_C. For example, $\psi_{HPC} = 0.98$ signifies a 2% relative loss in efficiency of HPC. Actuator faults can also be injected through the scale shift parameters for the three actuators, VSV, VBV and Wf.

In the C-MAPSS test-bed, sensor degradations are also realized as injected faults. Depending on the location and modality of the sensor, there could be several different degradation levels. For example, the degradation levels in a pressure sensor have different characteristics from those of a temperature sensor. Furthermore, depending on the location and operating environment, even sensors of the same modality could have different degradation characteristics. In general, sensor degradation is categorized as the following [133]: (i) Bias fault (i.e., a constant bias in the sensor observation); (ii) Drifting fault that is slowly varying; and (iii) Change in sensor-noise variance.

Appendix B

Science of Multi-agent Complex Systems: A Survey

The science of Multi-agent complex systems encompasses a wide variety of fields of both science and engineering. For example, biologists and environmentalists necessarily try to model multi-agent systems while investigating spread of epidemics, sexually transmitted diseases etc [134]. Social networks are one of most prominent examples of multi-agent complex systems present in nature [95]. Recently, application of multi-agent system concepts in the financial market are showing immense promise [97]. On the other hand, swarm robotics to solve numerous problems, such as pattern formation, aggregation, chain formation, self-assembly, coordinated movement, hole avoidance, foraging, self-deployment involves probably the most prominent research on multi-agent complex systems in engineering [135]. Science of networks and percolation is another important field in this regard [27]. To model propagation of damage, to avoid congestion in communication networks, to detect events by sensor networks are some of the problems of active research in this field. Furthermore, software engineering, signal processing, self-organization are the other areas to deal with this systems as mentioned in introduction. Thus, while researchers in basic science areas try to characterize the inherently multi-agent systems present in nature, engineers try to design multi-agent systems to solve certain problems more efficiently in a robust and adaptive fashion. This chapter provides a comprehensive review and taxonomy of science of multi-agent systems primarily from an engineering perspective.

A recent review [135] on development in robotic swarms presented a detailed classification of problem areas and tools in that field. This survey adopts similar approach towards reviewing the field multi-agent complex systems in more generality.

1 Primary Definitions

- ***Agent***: An agent is an autonomous entity (virtual or physical) that has a perception of dynamic conditions in the environment and acts to affect conditions in the environment (possibly own territory). An agent is usually able to communicate (in a limited manner) with other agents in the same system to achieve a common goal, that one agent alone could not achieve. Intelligent agents have (limited) capability interpret perceptions, solve problems, draw inferences, and determine actions. A rational agent always tries to optimize an appropriate performance measure. In general, agents are of three types (according to Distributed Artificial Intelligence (DAI) literature [136]):

  Reactive agent - A reactive agent (automaton) is a 3-tuple of input, processes and output.This type is typically used for robotic swarms.

  Deliberative agent - In contrast a deliberative agent has an internal view of its environment and is able to follow its own plans.

  Hybrid agent - A hybrid agent is a mixture of reactive and deliberative, that follows its own plans, but also sometimes directly reacts to external events without deliberation (according to authority management)

- ***Multi-agent system***: A multi-agent system (MAS) is a system composed of multiple interacting (intelligent) agents (homogeneous or heterogeneous) for the purpose of solving certain problems (that are which are difficult or impossible for a single/monolithic system to solve) in a flexible (adaptive) and robust fashion [137].

As mentioned in the definition, agents can interact with each other in an MAS, either for coordination among cooperative agents or negotiation among self interested (non-cooperative) agents. In many practical scenarios the interaction can be

limited, intermittent and/or delayed, which makes the problem much harder compared to perfect interaction. Inter-agent interaction can be categorized as follows:

Interaction via sensing: This is a simple and limited type of interaction strategy where an agent broadcast messages only to certain agents of interest and other agents receives the information via sensing. However, there is no purposeful direct messaging from one agent to another. Identification of agents of interest to broadcast messages is known as the 'kin recognition property' of the sender [138, 139]. This term is biologically inspired, as similar property is observed among animals which try convey information only to its kind but not to its enemies. This type of interaction is used for flocking/aggregation applications in robotics or event detection problem in sensor networks.

Interaction via environment: In this type of interaction the sender leaves information in the environment and receiving agents pick up the information by analyzing the environment. This is similar to the communication technology used by ants during foraging. Ants secrete pheromone on the path towards a food source for other ants to be able to get to the food. This method of communication is used for mobile robots in certain applications [140].

Interaction via communication: This is the most sophisticated and computationally intensive type of interaction among agents, where they use directed, purposeful communication with others [141]. This type is used in swarm robotics problem like chain formation. Also typical internet protocols use direct communication via IP addresses.

In MAS applications, environment with respect to an agent, knowledge-level of an agent and perception of an agent about the world are three crucial aspects, which are described below [137]

Environment: Environment can be static or dynamic. Artificial intelligence involving a single agent usually consider static environment primarily due to mathematical simplicity and existence of closed form solutions.However, for MAS applications, the environment is inherently dynamic due to presence of other agents and distinguishing dynamic effects created by other agents from the actual environment dynamics is a non trivial problem. Also, unstable behavior is observed for concurrently learning agents.

Knowledge: Agent-knowledge about the current world can be substantially

different from one to another due to limited observation. For example, homogeneous agents can share each others action set and current perception, and on the other hand, agents will be completely unable to infer adversarial agents action set and perception. Concept of common knowledge is a popular notion in MAS applications where, each agent knows a global fact and also knows that the fact is known by everybody.

Perception: Similar to knowledge, perception of different agents can also be very different due to spatially, temporally and semantically limited and different observation made by different agents. And, under this partial observability, optimal multi-agent planning is an intractable problem.

2 Modeling

Modeling of multi-agent systems is an important research issue which is required both for better understanding of any given MAS or to engineer a new MAS for a particular application. In general, modeling can be divided into two major categories, namely *microscopic* and *macroscopic* modeling. The models primarily try to capture the stability-convergence characteristics, emerging behavior, critical properties, scaling issues of the concerned MAS. However, in many cases it is not possible to obtain closed form analytical solutions for such models. Therefore, systematic simulation is an important aspect of analyzing multi-agent systems.

Microscopic: In this modeling technique, each agent, its interaction with others and the environment is modeled separately. A variation of such technique is known as sensor-based modeling [142, 143], which is a very popular method for robotic swarm. In such methods, models of sensors, actuators of agents and objects in the environment are the main components of the modeled system. Also, the interactions of the agents with the environment and the interactions between the agents are modeled. Such models can be either non-physical (no agent/environment dynamics is considered) or more realistic (agent/environment dynamics is considered). Modeling for aggregation, collaboration problems are some of the successful applications of this approach. Another mature microscopic modeling technique, that is extremely popular for biological systems, dynamical systems is cellular automata [144]. Cellular automata consists of (possibly multi-

dimensional) discrete lattice cells (generally homogeneous) where each cell has finite number of possible states and only local interaction is allowed. Cellular automata are known to have successfully modeled some complex non-equilibrium behavior. Sophisticated methods of microscopic modeling, e.g. Ising model, Potts model are developed in the field of Statistical mechanics [44]. Use of such techniques for modeling human engineered MAS applications are becoming more and more popular. However, these methods are more relevant for modeling equilibrium behavior of a system. Thus, microscopic models provide a realistic global behavior along with fluctuations and are typically used for designing a global controllers, handling noise etc.

Macroscopic: In contrast, macroscopic models provide an average global behavior. Usually, this can be understood as a mean field approach that uses difference equation of system states (average of agent states) for modeling. While the system needs to be iterated for each agent in microscopic models, macroscopic models are solved only once to obtain the average behavior of the whole system. Macroscopic models are specially useful for supervisory level controllers that modify agent level behaviors based on situational awareness or change in global objective. Recently, such a mean field modeling of the internet TCP (implementing RED through a bottleneck buffer) has been done [104], which is essentially a discrete time (ergodic theoretic) dynamical system showing bifurcation and other nonlinear instabilities. This application uses simple fluid flow model structure for network modeling. However, it is not always possible to obtain such a reliable model for general networks due to various complications, e.g. issue of multi-source, multi-destination etc.

3 Control of MAS

Agent behavior design or control of multi-agent systems can be achieved either in a centralized or in a decentralized manner. Centralized control actions guarantee operations of each individual agents. However, it is not feasible for real time control of large MAS due to extremely high computational complexity. On the other hand, decentralized control strategies have many advantages due to lower dimensionality, easier implementation, lower cost, etc. More importantly, robustness

and fault tolerance capabilities make them more useful. However, in most cases decentralized control only guarantees average behavior of agents. Moreover, limited closed form analytical results of stability and convergence are available for such strategies making them extremely dependent on simulation studies. Often, complete understanding of emergent behavior of certain decentralized strategies may be intractable. This section focuses on classification of decentralized strategies along with their application areas. Initially, the basic approaches for decentralized control are discussed and then ways of learning and adaptation of such strategies are discussed.

3.1 Basic Decentralized Control approaches

Following are the basic strategies of decentralized control that can also be used in conjunction with each other.

Subsumption: This is one of the most classical modular behavior-based robotics control architecture initially developed by Brooks et al. [145]. In this architecture, upper level behavior subsumes the lower level behavior, e.g. obstacle avoidance function can be subsumed by exploration which in turn can be subsumed by function of reaching a goal. This technique is advantageous in terms of modularity, speed and goal-oriented behavior. However, limited operating domain (cant accommodate conflicting interests), low flexibility are among its disadvantages. This strategy is widely used for single and small group of robots for various applications. Also, for a large group of robots, this strategy has been used for solving problems, such as flocking, aggregation, foraging etc.

Probabilistic Finite State Automata (PFSA): This approach involves modeling of agent behaviors as states of an PFSA and defining the state transitions with some external input and probabilities. For example, states may be search, retrieve, deposit, rest and give up for a typical foraging task. Successful application has been done for event detection in power constrained sensor networks. For a group of heterogeneous agents, a variation of PFSA, namely Petrinets are used [146].

Neural networks: An agent controller is designed as a preprogrammed perceptron in this strategy, which takes sensory inputs producing motor outputs.

Successful application have been shown for problems such as object clustering, self-assembly etc. However, scalability is a potential problem in this strategy.

Artificial potential methods: Artificial potential method is one of the most celebrated completely distributed control architecture used in almost every type of MAS problems. In fact, before application to multi-agent systems, it has been quite successful in single agent problems, e.g single robot path planning. This technique is primarily inspired by the notion of potential fields of nature, e.g. electro-magnetic field, gravitational field etc. The prime advantage of this method is its ability to incorporate conflicting interests in the artificial potential field which drives an agent towards the goal and keeps it away from adversity. The method's promise has been shown in various applications, such as static and mobile obstacle avoidance, shape formation, coverage, toll booth traffic control problems, routing in communication network etc. Recently, Kumar et al. [147] have shown some analytical results on stability and convergence of time invariant artificial potential based distributed algorithms for shape formation by robotic swarm. However, while incorporating conflicting interests in a single potential function, an artificial potential field often has the problem of having saddle points, where if reached, the agent fails to have a non-zero local gradient to move further. Also, time-varying potential fields are still very much under-explored. Ant-colony optimization [148] based or statistical mechanics (Ising/Potts model) inspired [149] agent behavior design approaches can also be rendered as variations of artificial potential based methods, which have been used for several applications.

3.2 Adaptation & Learning

Given a basic decentralized control algorithm functioning for for an MAS, an important question is, how does it adapt to small or large changes in environment, change in behavior of the neighboring agents, change in mission objective etc. This aspect brings about the notion of robustness and resilience of a controller. Robustness indicates the ability of the control algorithm to take care of fluctuations and uncertainties within a certain bound in an infinite time horizon. On the other hand, resilience property necessarily changes/adapts the control strategy to accommodate a drastic change in situation in a finite time horizon. In AI liter-

ature, learning is known as the method of automatically changing/adapting the control strategy based on situational awareness and system performance. Typically, when such adaptation occurs in the time-scale (fast time-scale) of system operation, it is called *learning* [150] and when it happens in a slower time-scale, it is called *evolution*. Following are the different types of learning procedure found in literature,

3.2.1 Learning

- *Supervised learning*: In this procedure, control strategies are generated to map current inputs to desired outputs, necessarily by retrieving information from an existing database.

- *Unsupervised learning*: There is no existing database in this case. Therefore, an agent tries to model a set of inputs. Understandably, this method is often not suitable for complex applications.

- *Reinforced learning*: This method is the most relevant and popular learning algorithm [151]. Suppose, environment has a certain set of states (the current state can be identified from sensor information) and agent has a certain set of actions (that is basically actuator information). Now, an efficiency index of the current control strategy can be obtained by evaluating the performance; the goal is to iteratively reach a high value of the efficiency index. Theoretical convergence results exist primarily for single agent systems. However, the number of required trial epochs grows significantly with the increase in state and action space. Also, large noise is a serious problem for convergence.

For single agent systems, learning with Markov decision processes, such as Q-learning [152], value iteration are quite successful. However, for an MAS, having multiple agents itself breaks some of the key assumptions of theoretical convergence. Therefore, substantial research is being pursued on MAS reinforcement learning algorithms. Concepts of stochastic/Markov games are being investigated, where the goal is to find a Nash equilibrium, that is a collection of strategies for all players such that every players selects a best response action. Algorithms like Q-learning with dynamic structuring of exploration space, based on GA (QDSEGAY)

have been developed and applied on heterogeneous mobile robots working on a load transportation task [153]. Other algorithms, such as independent learning, coupled learning, sparse cooperative learning, fuzzy logic based learning are also being investigated for this purpose. Recently, an ergodic theoretic non-equilibrium active queue management has been reported which was also validated on NS-2 simulator [154]. For MAS, reinforcement learning can be of two types which are as follows:

Local reinforcement: In this case, efficiency index is measured based on performance during achieving a sub-goal and rewards are given to only those agents who contributed to the sub-goal. This approach is more suitable considering limited communication among agents and the fundamental philosophy of decentralized approaches. Such technique has been used in Minimum Power Topology wireless networks for learning from local neighborhood information [155]. Furthermore, this has been used for foraging, cooperative stick pulling with homogeneous and heterogeneous agents etc. However, local rewarding often does not help in an overall collaboration.

Global reinforcement: In contrast to local reinforcement policy, all the agents (even if somebody who did not contribute) are rewarded for any success of the group in this strategy [156]. Although, this has been implemented successfully in foraging problems, it has drawbacks of reduced efficiency and construction a global reward function is often gets very complicated.

3.2.2 Evolution

Inspired by biological terminology, adaptation in a slower time-scale compared to the operating time-scale, is known as evolution [157]. Typically, genetic algorithm is used to change policies of a supervisory controller in this method. Such technique has been used for solving problems, such as aggregation, hole avoidance etc.

Appendix C

Language-measure Theory: A brief Background

This appendix summarizes the concept of signed real measure of regular languages (used in Chapter 6); the details are reported in [158].

Let $G_i = (Q, \Sigma, \delta, q_i, Q_m)$ be a finite-state automaton model that encodes all possible evolutions of the discrete-event dynamics of a physical plant, where $Q = \{q_k : k \in \mathcal{I}_Q\}$ is the set of states and $\mathcal{I}_Q \equiv \{1, 2, \cdots, n\}$ is the index set of states; the automaton starts with the initial state q_i; the alphabet of events is $\Sigma = \{\sigma_k : k \in \mathcal{I}_\Sigma\}$, having $\Sigma \bigcap \mathcal{I}_Q = \emptyset$ and $\mathcal{I}_\Sigma \equiv \{1, 2, \cdots, \ell\}$ is the index set of events; $\delta : Q \times \Sigma \to Q$ is the (possibly partial) function of state transitions; and $Q_m \equiv \{q_{m_1}, q_{m_2}, \cdots, q_{m_l}\} \subseteq Q$ is the set of marked (i.e., accepted) states with $q_{m_k} = q_j$ for some $j \in \mathcal{I}_Q$. Let Σ^* be the Kleene closure of Σ, i.e., the set of all finite-length strings made of the events belonging to Σ as well as the empty string ϵ that is viewed as the identity of the monoid Σ^* under the operation of string concatenation, i.e., $\epsilon s = s = s\epsilon$. The state transition map δ is recursively extended to its reflexive and transitive closure $\delta : Q \times \Sigma^* \to Q$ by defining

$$\forall q_j \in Q,\ \delta(q_j, \epsilon) = q_j \tag{C.1a}$$

$$\forall q_j \in Q, \sigma \in \Sigma, s \in \Sigma^\star,\ \delta(q_i, \sigma s) = \delta(\delta(q_i, \sigma), s) \tag{C.1b}$$

Definition 0.1. *The language $L(q_i)$ generated by a DFSA G initialized at the state*

$q_i \in Q$ is defined as:

$$L(q_i) = \{s \in \Sigma^* \mid \delta^*(q_i, s) \in Q\} \tag{C.2}$$

The language $L_m(q_i)$ marked by the DFSA G initialized at the state $q_i \in Q$ is defined as:

$$L_m(q_i) = \{s \in \Sigma^* \mid \delta^*(q_i, s) \in Q_m\} \tag{C.3}$$

Definition 0.2. *For every $q_j \in Q$, let $L(q_i, q_j)$ denote the set of all strings that, starting from the state q_i, terminate at the state q_j, i.e.,*

$$L_{i,j} = \{s \in \Sigma^* \mid \delta^*(q_i, s) = q_j \in Q\} \tag{C.4}$$

To complete the specification of a probabilistic finite state automata, the event generation probabilities and the state characteristic weight vector are defined next.

Definition 0.3. *The event generation probabilities are specified by the function $\tilde{\pi} : Q \times \Sigma^\star \to [0,\, 1]$ such that $\forall q_j \in Q, \forall \sigma_k \in \Sigma, \forall s \in \Sigma^\star$,*

(1) $\tilde{\pi}(q_j, \sigma_k) \triangleq \tilde{\pi}_{jk} \in [0, 1)$; $\sum_k \tilde{\pi}_{jk} = 1 - \theta$, with $\theta \in (0, 1)$;

(2) $\tilde{\pi}(q_j, \sigma) = 0$ if $\delta(q_j, \sigma)$ is undefined; $\tilde{\pi}(q_j, \epsilon) = 1$;

(3) $\tilde{\pi}(q_j, \sigma_k s) = \tilde{\pi}(q_j, \sigma_k)\ \tilde{\pi}(\delta(q_j, \sigma_k), s)$.

Notation 1. *The $n \times \ell$ event cost matrix $\widetilde{\Pi}$ is defined as: $\widetilde{\Pi}|_{ij} = \tilde{\pi}(q_i, \sigma_j)$*

Definition 0.4. *The state transition probability $\pi : Q \times Q \to [0, 1)$, of the DFSA G_i is defined as follows:*

$$\forall q_i, q_j \in Q, \quad \pi_{ij} = \sum_{\sigma \in \Sigma \text{ s.t. } \delta(q_i, \sigma) = q_j} \tilde{\pi}(q_i, \sigma) \tag{C.5}$$

Notation 2. *The $n \times n$ state transition probability matrix Π is defined as $\Pi|_{ij} = \pi(q_i, q_j)$*

The set Q_m of marked states is partitioned into Q_m^+ and Q_m^-, i.e., $Q_m = Q_m^+ \cup Q_m^-$ and $Q_m^+ \cap Q_m^- = \emptyset$, where Q_m^+ contains all *good* marked states that should be reached, and Q_m^- contains all *bad* marked states that should be avoided, although

it may not always be possible to completely avoid the *bad* states while attempting to reach the *good* states. To characterize this, each marked state is assigned a real value based on the designer's perception of its impact on the system performance.

Definition 0.5. *The characteristic function $\chi : Q \to [-1, 1]$ that assigns a signed real weight to state-based sublanguages $L(q_i, q)$ is defined as:*

$$\forall q \in Q, \quad \chi(q) \in \begin{cases} [-1, 0), & q \in Q_m^- \\ \{0\}, & q \notin Q_m \\ (0, 1], & q \in Q_m^+ \end{cases} \tag{C.6}$$

The state weighting vector, denoted by $\boldsymbol{\chi} = [\chi_1 \ \chi_2 \ \cdots \ \chi_n]^T$, where $\chi_j \equiv \chi(q_j)$ $\forall j \in \mathcal{I}_Q$, is called the $\boldsymbol{\chi}$-vector. The j-th element χ_j of $\boldsymbol{\chi}$-vector is the weight assigned to the corresponding terminal state q_j.

Remark 0.1. *The state characteristic function $\chi : Q \to [-1, 1]$ or equivalently the characteristic vector $\boldsymbol{\chi}$ is analogous to the notion of the reward function in Markov Decision Process (MDP) analysis. However, unlike MDP models, where the reward (or penalty) is put on individual state-based actions, in this model, the characteristic is put on the state itself. The similarity of the two notions is clarified by noting that just as MDP performance can be evaluated as the total reward garnered as actions are executed sequentially, the performance of a PFSA can be computed by summing the characteristics of the states visited due to transpired event sequences.*

Plant models considered here are *deterministic* finite state automata with well-defined event occurrence *probabilities*. In other words, the occurrence of events is probabilistic, but the state at which the plant ends up, *given a particular event has occurred*, is deterministic. No emphasis is laid on the initial state of the plant *i.e.* the plant is allowed to start from any state. Furthermore, having defined the characteristic state weight vector $\boldsymbol{\chi}$, it is not necessary to specify the set of marked states, because if $\chi_i = 0$, then q_i is not marked and if $\chi_i \neq 0$, then q_i is marked.

Definition 0.6. *(Control Philosophy) If $q_i \underset{\sigma}{\to} q_k$, and the event σ is disabled at state q_i, then the supervisory action is to prevent the plant from making a transition to the state q_k, by forcing it to stay at the original state q_i. Thus disabling any transition σ at a given state q results in deletion of the original transition and*

appearance of the self-loop $\delta(q,\sigma) = q$ with the occurrence probability of σ from the state q remaining unchanged in the supervised and unsupervised plants.

Definition 0.7. *(Controllable Transitions) For a given plant, transitions that can be disabled in the sense of Definition 0.6 are defined to be controllable transitions. The set of controllable transitions in a plant is denoted $\mathcal{C}$. Note controllability is state-based.*

It follows that plant models can be specified by the sextuplet:

$$G = (Q, \Sigma, \delta, \widetilde{\Pi}, \boldsymbol{\chi}, \mathcal{C}) \tag{C.7}$$

The formal language measure is first defined for terminating plants [159] with sub-stochastic event generation probabilities, i.e., the event generation probabilities at each state summing to strictly less than unity. In general, the marked language $L_m(q_i)$ consists of both good and bad event strings that, starting from the initial state q_i, lead to Q_m^+ and Q_m^- respectively. Any event string belonging to the language $L^0(q_i) = L(q_i) - L_m(q_i)$ leads to one of the non-marked states belonging to $Q - Q_m$ and L^0 does not contain any one of the good or bad strings. Based on the equivalence classes defined in the Myhill-Nerode Theorem [12], the regular languages $L(q_i)$ and $L_m(q_i)$ can be expressed as:

$$L(q_i) = \bigcup_{q_k \in Q} L_{i,k} \tag{C.8}$$

$$L_m(q_i) = \bigcup_{q_k \in Q_m} L_{i,k} = L_m^+ \cup L_m^- \tag{C.9}$$

where the sublanguage $L_{i,k} \subseteq L(q_i)$ having the initial state q_i is uniquely labelled by the terminal state $q_k, k \in \mathcal{I}_Q$ and $L_{i,j} \cap L_{i,k} = \emptyset \; \forall j \neq k$; and $L_m^+ \equiv \bigcup_{q_k \in Q_m^+} L_{i,k}$ and $L_m^- \equiv \bigcup_{q_k \in Q_m^-} L_{i,k}$ are good and bad sublanguages of $L_m(q_i)$, respectively. Then, $L^0 = \bigcup_{q_k \notin Q_m} L_{i,k}$ and $L(q_i) = L^0 \cup L_m^+ \cup L_m^-$.

A signed real measure $\mu^i : 2^{L(q_i)} \rightarrow \mathbb{R} \equiv (-\infty, +\infty)$ is constructed on the σ-algebra $2^{L(q_i)}$ for any $i \in \mathcal{I}_Q$; interested readers are referred to [55] [8] for the details of measure-theoretic definitions and results. With the choice of this σ-algebra, every singleton set made of an event string $s \in L(q_i)$ is a measurable set.

By Hahn Decomposition Theorem [160], each of these measurable sets qualifies itself to have a numerical value based on the above state-based decomposition of $L(q_i)$ into L^0(null), L^+(positive), and L^-(negative) sublanguages.

Definition 0.8. *Let $\omega \in L(q_i, q_j) \subseteq 2^{L(q_i)}$. The signed real measure μ^i of every singleton string set $\{\omega\}$ is defined as:*

$$\mu^i(\{\omega\}) = \tilde{\pi}(q_i, \omega)\chi(q_j) \tag{C.10}$$

The signed real measure of a sublanguage $L_{i,j} \subseteq L(q_i)$ is defined as:

$$\mu_{i,j} = \mu^i(L(q_i, q_j)) = \left(\sum_{\omega \in L(q_i, q_j)} \tilde{\pi}(q_i, \omega) \right) \chi_j \tag{C.11}$$

Therefore, the signed real measure of the language of a DFSA G_i initialized at $q_i \in Q$, is defined as

$$\mu_i = \mu^i(L(q_i)) = \sum_{j \in \mathcal{I}_Q} \mu^i(L_{i,j}) \tag{C.12}$$

It is shown in [55] [8] that the language measure in Eq. (C.12) can be expressed as

$$\mu_i = \sum_{j \in \mathcal{I}_Q} \pi_{ij}\mu_j + \chi_i \tag{C.13}$$

The language measure vector, denoted as $\boldsymbol{\mu} = [\mu_1 \ \mu_2 \ \cdots \ \mu_n]^T$, is called the $\boldsymbol{\mu}$-vector. In vector form, Eq. (C.13) becomes

$$\boldsymbol{\mu} = \Pi\boldsymbol{\mu} + \boldsymbol{\chi} \tag{C.14}$$

whose solution is given by

$$\boldsymbol{\mu} = (\mathbb{I} - \Pi)^{-1}\boldsymbol{\chi} \tag{C.15}$$

The inverse in Eq. (C.15) exists for terminating plant models [159][161] because Π is a contraction operator [55] [8] due to the strict inequality $\sum_j \pi_{ij} < 1$. The residual $\theta_i = 1 - \sum_j \pi_{ij}$ is referred to as the termination probability for state $q_i \in Q$. We extend the analysis to non-terminating plants [159][161] with stochastic

transition probability matrices (*i.e.* with $\theta_i = 0,\ \forall q_i \in Q$) by renormalizing the language measure [119] with respect to the uniform termination probability of a limiting terminating model as described next.

Let $\widetilde{\Pi}$ and Π be the stochastic event generation and transition probability matrices for a non-terminating plant $G_i = (Q, \Sigma, \delta, q_i, Q_m)$. We consider the terminating plant $G_i(\theta)$ with the same DFSA structure $(Q, \Sigma, \delta, q_i, Q_m)$ such that the event generation probability matrix is given by $(1-\theta)\widetilde{\Pi}$ with $\theta \in (0,1)$ implying that the state transition probability matrix is $(1-\theta)\Pi$.

Definition 0.9. *(Renormalized Measure) The renormalized measure* $\nu_\theta^i : 2^{L(q_i)} \to [-1,1]$ *for the* θ*-parametrized terminating plant* $G_i(\theta)$ *is defined as:*

$$\forall \omega \in L(q_i),\ \nu_\theta^i(\{\omega\}) = \theta \mu^i(\{\omega\}) \tag{C.16}$$

The corresponding matrix form is given by

$$\boldsymbol{\nu_\theta} = \theta\ \boldsymbol{\mu} = \theta\ [I - (1-\theta)\Pi]^{-1}\boldsymbol{\chi} \text{ with } \theta \in (0,1) \tag{C.17}$$

We note that the vector representation allows for the following notational simplification

$$\nu_\theta^i(L(q_i)) = \boldsymbol{\nu_\theta}\big|_i \tag{C.18}$$

The renormalized measure for the non-terminating plant G_i *is defined to be* $\lim\limits_{\theta \to 0^+} \nu_\theta^i$.

The following results are retained for the sake of completeness. Complete proofs can be found in [119].

Proposition 0.1. *The limiting measure vector* $\boldsymbol{\nu}_0 \triangleq \lim_{\theta \to 0^+} \boldsymbol{\nu}_\theta$ *exists and* $||\boldsymbol{\nu}_0||_\infty \leq 1$.

Proposition 0.2. *Let* Π *be the stochastic transition matrix of a non-terminating PFSA [159, 161]. Then, as the parameter* $\theta \to 0^+$*, the limiting measure vector is obtained as:* $\boldsymbol{\nu}_0 = \mathcal{C}(\Pi)\boldsymbol{\chi}$ *where the matrix operator* $\mathcal{C}(\Pi) \triangleq \lim\limits_{k \to \infty} \dfrac{1}{k} \sum\limits_{j=0}^{k-1} \Pi^j$ *is the Cesaro limit [56] of the stochastic transition matrix* Π.

Corollary 0.1. *(to Proposition 0.2) The expression $\mathcal{C}(\Pi)\boldsymbol{\nu}_\theta$ is independent of θ. Specifically, the following identity holds for all $\theta \in (0,1)$.*

$$\mathcal{C}(\Pi)\boldsymbol{\nu}_\theta = \mathcal{C}(\Pi)\boldsymbol{\chi} \tag{C.19}$$

Notation 3. *The linearly independent orthogonal set $\{v^i \in \mathbb{R}^{\text{CARD}(Q)} : v^i_j = \delta_{ij}\}$ is denoted as $\mathcal{B}$ where δ_{ij} denotes the Krönecker delta function. We note that there is a one-to-one onto mapping between the states $q_i \in Q$ and the elements of $\mathcal{B}$, namely,*

$$q_i \mapsto \alpha \iff \alpha_k = \begin{cases} 1 & \text{if } k = i \\ 0 & \text{otherwise} \end{cases} \tag{C.20}$$

Definition 0.10. *For any non-zero vector $v \in \mathbb{R}^{\text{CARD}(Q)}$, the normalizing function $\mathcal{N} : \mathbb{R}^{\text{CARD}(Q)} \setminus \mathbf{0} \to \mathbb{R}^{\text{CARD}(Q)}$ is defined as $\mathcal{N}(v) = \frac{v}{\sum_i v_i}$.*

Bibliography

[1] D. K. Frederick, J. A. DeCastro, and J. S. Litt, "Users guide for the commercial modular aero-propulsion system simulation (C-MAPSS)," October 2007. NASA/TM2007-215026.

[2] "The cyber-physical systems (cps) summit report," in *CPS Week multiconference*, (St. Louis, MO), 2008.

[3] P. Marwedel, *Embedded System Design*. Kluwer Academic Publishers, 2003.

[4] E. A. Lee, "Computing foundations and practice for cyber-physical systems: A preliminary report," *Technical Report UCB/EECS-2007-72, EECS Department, University of California, Berkeley*, May 2007.

[5] P. Marwedel, "Embedded and cyber-physical systems in a nutshell," *DAC.COM Knowledge Center Article*, 2010.

[6] K. Miettinen, *Nonlinear Multiobjective Optimization*. Kluwer Academic, Boston, MA, USA, 1998.

[7] P. Ramadge and W. Wonham, "Supervisory control of a class of discrete event processes," *SIAM J. Control and Optimization*, vol. 25, no. 1, pp. 206 – 230, 1987.

[8] A. Ray, V. Phoha, and S. Phoha, *Quantitative measure for discrete event supervisory control*. Springer, New York, 2005.

[9] J. Lewis, E. Rashba, and D. Brophy, "Vhdl-2006-d3.0, tutorial, design, automation, and test in europe (date)," 2007.

[10] "IEEE standard for systemverilog- unified hardware design, specification, and verification language," 2009.

[11] "Open systemc initiative, IEEE 1666 LRM," 2005.

[12] J. E. Hopcroft, R. MOTWANI, and J. D. ULLMAN, *Introduction to Automata Theory, Languages, and Computation.* 2nd ed., Addison-Wesley, 2001.

[13] S. Sarkar, X. Jin, and A. Ray, "Data-driven fault detection in aircraft engines with noisy sensor measurements," *Journal of Engineering for Gas Turbines and Power-Transactions of the ASME*, vol. 133, no. 8, p. 081602 (10 pages), 2011.

[14] X. Jin, Y. Guo, S. Sarkar, A. Ray, and R. M. Edwards, "Anomaly detection in nuclear power plants via symbolic dynamic filtering," *IEEE Transactions on Nuclear Science*, vol. 58, no. 1, pp. 277–288, 2011.

[15] C. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems.* Kluwer Academic Publishers, 1999.

[16] J. Bengtsson and W. Yi, "Timed automata: Semantics, algorithms and tools," *Desel, J. and Reisig, W. and Rozenberg, G. (eds.), Proc. 4th Advanced Course on Petri Nets (ACPN 2003), LNCS 3098, Springer*, p. 87124, 2004.

[17] A. Ray, "Symbolic dynamic analysis of complex systems for anomaly detection," *Sig. Process.*, vol. 84, no. 7, pp. 1115–1130, 2004.

[18] W. Reisig, *Petri Nets.* Springer Verlag, 1985.

[19] J. Lygeros, *Lecture Notes on Hybrid Systems.* Department of Electrical and Computer Engineering, University of Patras, Rio, Patras, GR-26500, Greece, 2004.

[20] P. Antsaklis, J. Stiver, and M. Lemmon, "Hybrid system modeling and autonomous control systems," in *Hybrid Systems* (R. Grossman, A. Nerode, A. Ravn, and H. Rischel, eds.), vol. 736 of *Lecture Notes in Computer Science*, pp. 366–392, Springer Berlin / Heidelberg, 1993.

[21] A. Balluchi, L. Benvenuti, G. M. Miconi, U. Pozzi, T. Villa, M. D. DiBenedetto, H. Wong-Toi, and A. L. Vincentelli, *Maximal safe set computation for idle speed control of an automotive engine*, vol. 1790. Hybrid Systems: Computation and Control, ser. Lecture Notes in Computer Science, Lynch, N. and Krogh, B. H. Eds. Springer-Verlag, 2000.

[22] G. Meyer, "Design of fight vehicle management systems," in *Proceedings of the IEEE Conference on Decision and Control*, (Lake Buena Vista, FL), 1994.

[23] A. Back, J. Guckenheimer, and M. Myers, *A dynamical simulation facility for hybrid systems*, vol. 736. Hybrid Systems, ser. Lecture Notes in Computer Science, Grossman, R. L. and Nerode, A. and Ravn, A. P. and Rischel, H., Eds. Springer-Verlag, 1993.

[24] T. Bak, J. Bendtsen, and R. A. P., *Hybrid control design for a wheeled mobile robot*, vol. 2623. Hybrid Systems: Computation and Control, ser. Lecture Notes in Computer Science, Malr, O. and Pnueli, A. Eds. Springer-Verlag, 2003.

[25] D. N. Godbole, J. Lygeros, and S. Sastry, "Hierarchical hybrid control: A case stuy," in *Proceedings of the IEEE Conference on Decision and Control*, (Lake Buena Vista, FL), pp. 1592–1597, 1994.

[26] P. P. Varaiya, "Smart cars on smart roads: Problems of control," *IEEE Transactions on Automatic Control*, vol. 38, no. 2, pp. 195–207, 1993.

[27] R. Albert and A.-L. Barabasi, "Statistical mechanics of complex networks," *Review of Modern Physics*, vol. 74, pp. 47–97, January 2002.

[28] F. V. Jensen, *Bayesian Networks and Decision Graphs*. Springer-Verlag, NY, 2001.

[29] I. Chattopadhyay and A. Ray, "Structural transformation of probabilistic finite state machines," *Int. J. Control*, vol. 81, no. 5, pp. 820–835, 2008.

[30] D. Angluin and C. H. Smith, "Inductive inference: Theory and methods," *Computing Surveys*, vol. 15, no. 3, pp. 237–269, 1983.

[31] J. P. Crutchfield and K. Young, "Inferring statistical complexity," *Phys. Review Lett.*, vol. 63, pp. 105–108, 1989.

[32] V. Rajagopalan, S. Chakraborty, and A. Ray, "Estimation of slowly-varying parameters in nonlinear systems via symbolic dynamic filtering," *Signal Processing*, vol. 89, no. 2, pp. 339–348, 2008.

[33] D. Lind and M. Marcus, *An Introduction to Symbolic Dynamics and Coding*. Cambridge University Press, Cambridge, U.K., 1995.

[34] D. Heckerman and D. Geiger, "Learning Bayesian Networks," Tech. Rep. MSR-TR-95-02, Microsoft Research, Redmond, WA, December 1994.

[35] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1988.

[36] F. Jelinek, J. D. Lafferty, and R. L. Mercer, "Basic methods of probabilistic context free grammars," *Speech Recognition and Understanding. Recent Advances, Trends, and Applications*, vol. F75, pp. 35–360, 1992.

[37] K. Murphy, "Passively learning finite automata," *Technical report, Santa Fe Institute*, p. citeseer.ist.psu.edu/murphy95passively.html, 1996.

[38] D. Ron, Y. Singer, and N. Tishby, "Learning probabilistic automata with variable memory length," in *Computational Learing Theory*, pp. 35–46, 1994.

[39] D. Ron, Y. Singer, and N. Tishby, "On the learnability and usage of acyclic probabilistic finite automata," *Journal of Computer and System Sciences*, vol. 56, no. 2, pp. 133–152, 1998.

[40] V. Rajagopalan and A. Ray, "Symbolic time series analysis via wavelet-based partitioning," *Signal Processing*, vol. 86, no. 11, pp. 3309–3320, 2006.

[41] A. Subbu and A. Ray, "Space partitioning via hilbert transform for symbolic time series analysis," *Applied Physics Letters*, vol. 92, no. 8, pp. 084107–1 to 084107–3, February 2008.

[42] S. Sarkar, K. Mukherjee, and A. Ray, "Generalization of hilbert transform for symbolic analysis of noisy signals," *Signal Processing*, vol. 89, no. 6, pp. 1245–1251, 2009.

[43] S. Gupta and A. Ray, "Statitical mechanics of complex systems for pattern identification," *to appear in Journal of Statistical Physics*, 2009.

[44] R. Pathria, *Statistical Mechanics*. Oxford, UK: Butterworth-Heinemann, 2nd ed., 1996.

[45] R. Duda, P. Hart, and D. Stork, *Pattern Classification*. John Wiley, New York, 2001.

[46] T. Cover and J. Thomas, *Elements of Information Theory*. New York, NY, USA: John Wiley, 1991.

[47] C. Rao, A. Ray, S. Sarkar, and M. Yasar, "Review and comparative evaluation of symbolic dynamic filtering for detection of anomaly patterns," *Signal, Image and Video Processing*, vol. 3, no. 2, pp. 101–114, 2009.

[48] S. Gupta, A. Ray, S. Sarkar, and M. Yasar, "Fault detection and isolation in aircraft gas turbine engines: Part i - underlying concept," *Proceedings of the I Mech E Part G: Journal of Aerospace Engineering*, vol. 222, no. 3, pp. 307–318, May 2008.

[49] C. Beck and F. Schlögl, *Thermodynamics of chaotic systems: an introduction*. Cambridge University Press, United Kingdom, 1993.

[50] R. Badii and A. Politi, *Complexity hierarchical structures and scaling in physics*. Cambridge University Press, United Kingdom, 1997.

[51] C. S. Daw, C. E. A. Finney, and E. R. Tracy, "A review of symbolic analysis of experimental data," *Review of Scientific Instruments*, vol. 74, no. 2, pp. 915–930, 2003.

[52] H. Kantz and T. Schreiber, *Nonlinear Time Series Analysis, 2nd ed.* Cambridge, U.K.: Cambridge University Press, 2004.

[53] M. Kennel and M. Buhl, "Estimating good discrete partitions from observed data: Symbolic false nearest neighbors," *Physical Review Letters*, vol. 91, no. 8, p. 084102, 2003.

[54] C. Rao, K. Mukherjee, S. Sarkar, and A. Ray, "Statistical estimation of multiple parameters via symbolic dynamic filtering," *Signal Processing*, vol. 89, no. 6, pp. 981–988, June 2009.

[55] A. Ray, "Signed real measure of regular languages for discrete-event supervisory control," *Int. Journal of Control*, vol. 78, no. 12, pp. 949–967, 2005.

[56] R. Bapat and T. Raghavan, *Nonnegative Matrices and Applications*. Cambridge, U.K: Cambridge University Press, 1997.

[57] S. Gupta, A. Ray, and E. Keller, "Symbolic time series analysis of ultrasonic data for early detection of fatigue damage," *Mechanical Systems and Signal Processing*, vol. 21, no. 2, pp. 866–884, 2007.

[58] S. Chin, A. Ray, and V. Rajagopalan, "Symbolic time series analysis for anomaly detection: A comparative evaluation," vol. 85, no. 9, pp. 1859–1868, September 2005.

[59] S. Sarkar, M. Yasar, S. Gupta, A. Ray, and K. Mukherjee, "Fault detection and isolation in aircraft gas turbine engines: Part ii - validation on a simulation test bed," *Proceedings of the I Mech E Part G: Journal of Aerospace Engineering*, vol. 222, no. 3, pp. 319–330, May 2008.

[60] S. Chakraborty, S. Sarkar, S. Gupta, and A. Ray, "Damage monitoring of refractory wall in a generic entrained-bed slagging gasification system," *Proceedings of the I Mech E Part A: Journal of Power and Energy*, vol. 222, no. 8, pp. 791–807, October 2008.

[61] S. Chakraborty, A. Ray, A. Subbu, and E. Keller, "Analytic signal space partitioning and symbolic dynamic filtering for degradation monitoring of electric motors," *Signal, Image, and Video Processing, in press.*

[62] G. Mallapragada, I. Chattopadhyay, and A. Ray, "Automated behavior recognition in mobile robots using symbolic dynamic filtering," *Proceedings of the I Mech E Part I: Journal of Systems and Control Engineering*, vol. 222, no. 6, pp. 409–424, 2008.

[63] M. Buhl and M. Kennel, "Statistically relaxing to generating partitions for observed time-series data," *Physical Review E*, vol. 71, no. 4, p. 046213, 2005.

[64] X. Jin, S. Gupta, K. Mukherjee, and A. Ray, "Wavelet-based feature extraction using probabilistic finite state automata for pattern classification," *Pattern Recognition*, vol. 44, no. 7, pp. 1343–1356, 2011.

[65] D. Gabor, "Theory of communications," *J. Inst. Electrical Engineering*, vol. 93, p. 429457, 1946.

[66] L. Cohen, *Time-Frequency Analysis*. Prentice Hall PTR, 1995.

[67] A. Lohmann, D. Mendlovic, and Z. Zalevsky, "Fractional hilbert transform," *Opt. Lett.*, vol. 21, no. 4, p. 281283, 1996.

[68] S.-C. Pei and M.-H. Yeh, "Discrete fractional hilbert transform," *IEEE Transactions on Circuits and SystemsII: Analog and Digital Signal Processing*, vol. 47, no. 11, 2000.

[69] Y. Luo, S. Al-dossary, M. Mahroon, and M. Alfaraj, "Generalized hilbert transform and its applications in geophysics," *The Leading Edge*, March 2003.

[70] S. Sarkar, K. Mukherjee, X. Jin, and A. Ray, "Optimization of time-series data partitioning for parameter identification in nonlinear dynamical systems," in *ASME Dynamical Systems and Control Conference*, (Boston, MA), September 2010.

[71] D. S. Singh, S. Sarkar, S. Gupta, and A. Ray, "Optimal partitioning of ultrasonic data for fatigue damage detection," in *in Proceedings of American Control Conference, San Francisco, CA, USA*, 2011.

[72] R. Steuer, L. Molgedey, W. Ebeling, and M. Jimenez-Montano, "Entropy and optimal partition for data analysis," *The European Physical Journal B*, vol. 19, pp. 265–269, 2001.

[73] J. Li, X.-B. Ning, W. Wu, and X.-F. Ma, "Detecting dynamical complexity changes in time series using based-scale entropy," *Chinese Physics Letter*, vol. 14, no. 12, pp. 2428–2432, December 2005.

[74] H. Choset, "Coverage for robotics - a survey of recent results," *Annals of Mathematics and Artificial Intelligence*, vol. 31, no. 1-4, pp. 113–126, 2001.

[75] J. Thompson and H. Stewart, *Nonlinear Dynamics and Chaos*. Wiley, Chichester, UK, 1986.

[76] G. J. Mclachlan, *Discriminant Analysis and Statistical Pattern Recognition*. (Wiley Series in Probability and Statistics) Wiley-Interscience, 2004.

[77] E. Choi and C. Lee, "Feature extraction based on the bhattacharyya distance," *Pattern Recognition*, vol. 36, pp. 1703 – 1709, August 2003.

[78] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[79] M. Loog, R. Duin, and R. Haeb-Umbach, "Multiclass linear dimension reduction by weighted pairwise fisher criteria," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 7, p. 762766, 2001.

[80] A. Biem, S. Katagiri, and B. Juang, "Pattern recognition using discriminative feature extraction," *IEEE Trans. Signal Process.*, vol. 45, no. 2, p. 500504, 1997.

[81] R. Kohavi and F. Provost, "Glossary of terms," *Machine Learning*, vol. 30, no. 2/3, pp. 271–274, 1998.

[82] R. Alaiz-Rodrguez, A. Guerrero-Curieses, and J. Cid-Sueiro, "Minimax classifiers based on neural networks," *Pattern Recognition*, vol. 38, no. 1, pp. 29 – 39, 2005.

[83] X. Jin, S. Sarkar, K. Mukherjee, and A. Ray, "Suboptimal partitioning of time-series data for anomaly detection," in *Proceedings of 48th IEEE Conference on Decision and Control*, (Shanghai, China), pp. 1020–1025, December 2009.

[84] S. Sarkar, C. Rao, and A. Ray, "Statistical estimation of multiple faults in aircraft gas turbine engines," *Proceedings of the I Mech E Part G: Journal of Aerospace Engineering*, vol. 223, no. 4, pp. 415–424, 2009.

[85] D. L. Simon and S. Garg, "Optimal tuner selection for kalman filter-based aircraft engine performance estimation," *Journal of Engineering for Gas Turbines and Power*, vol. 132, no. 3, p. 031601, 2010.

[86] P. Lu, M. Zhang, T. Hsu, and J. Zhang, "An evaluation of engine faults diagnostics using artificial neural networks," *Journal of Engineering for Gas Turbines and Power*, vol. 123, no. 2, pp. 340–346, 2001.

[87] J. C. Newman, "Fastran-ii - a fatigue crack growth structural analysis program," *NASA Technical Memorandum 104159, Langley Research Center*, 1992.

[88] S. Sarkar, D. S. Singh, A. Srivastav, and A. Ray, "Semantic sensor fusion for fault diagnosis in aircraft gas turbine engines," in *American Control Conference*, (San Francisco, California, USA), 2011.

[89] O. Basir and X. Yuan, "Engine fault diagnosis based on multi-sensor information fusion using dempstershafer evidence theory," *Information Fusion*, vol. 8, pp. 379–386, 2007.

[90] C. Romessis and K. Mathioudakis, "Bayesian network approach for gas path fault diagnosis," *J. Eng. Gas Turbines Power*, vol. 128, pp. 64–72, 2006.

[91] E. Blasch, I. Kadar, K. Hintz, J. Biermann, C. Chong, and S. Das, "Resource Managemanet Coordination with Level 2/3 Fusion Issues and Challenges," *IEEE A& E Sytems Magazine*, pp. 32–46, March 2008.

[92] J. Armstrong, "Users guide for the transient test case generator," September 2009. NASA GRC Internal Report.

[93] R. Nelson, *An Introduction to Copulas, 2nd Edition*. Springer, New York, NY, USA, 2006.

[94] K. Huang, *Statistical Mechanics*. New York, NY, USA: John Wiley & Sons, 2nd ed., 1987.

[95] S. Durlauf, "How can statistical mechanics contribute to social science?," *Proc. Natl. Acad. Sci. USA*, vol. 96, p. 1058210584, September 1999.

[96] M. Millonas, "Swarms, phase transitions, and collective intelligence (paper 1); and a nonequilibrium statistical field theory of swarms and other spatially extended complex systems (paper 2)," Working Papers 93-06-039, Santa Fe Institute, June 1993.

[97] M. Schulz, *Statistical Physics and Economics: Concepts, Tools and Applications Springer Tracts in Modern Physics, Vol. 184*. Berlin, Germany: Springer, 2003.

[98] B. Krishnamachari, S. Wicker, and R. Bejar, "Phase transition phenomena in wireless ad-hoc networks," in *Global Telecommunications Conference, IEEE*, 2001.

[99] A. Srivastav, A. Ray, and S. Phoha, "Adaptive sensor activity scheduling in distributed sensor networks: A statistical mechanics approach," *International Journal of Distributed Sensor Networks*, vol. 5, no. 3, pp. 242–261, 2009.

[100] S. Sarkar, K. Mukherjee, A. Srivastav, and A. Ray, "Understanding phase transition in communication networks to enable robust and resilient control," in *American Control Conference*, (St. Louis, MO), 2009.

[101] T. Ohira and R. Sawatari, "Phase transition in computer network traffic model," *Physical Review E*, vol. 58, pp. 193–195, 1998.

[102] A. Lawniczak and X. Tang, "Network traffic behaviour near phase transition point," *The European Physical Journal B - Condensed Matter and Complex Systems*, vol. 50, no. 1-2, pp. 231–236.

[103] R. Guimera, A. Arenas, A. Diaz-Guilera, and F. Giralt, "Dynamical properties of model communication networks," *Physical Review E*, vol. 66, p. 026704, 2002.

[104] F. Baccelli, D. R. McDonald, and J. Reynier, "A mean-field model for multiple tcp connections through a buffer implementing red," *Performance Evaluation*, vol. 49, no. 1-4, pp. 77–97, 2002.

[105] S. Sarkar, K. Mukherjee, A. Srivastav, and A. Ray, "Critical phenomena and finite-size scaling in communication networks," in *American Control Conference*, (Baltimore, MD), 2010.

[106] M. Plischke and B. Bergersen, *Equilibrium Statistical Physics, 3rd Edition*. Singapore: World Scientific, 2005.

[107] D. Landau and K. Binder, *A Guide to Monte Carlo Simulations in Statistical Physics, 2nd Edition*. Cambridge University Press, 2005.

[108] L. Fichter, "Binary eutectic phase diagram," tech. rep., Department of Geology and Environmental Science, James Madison University, Harrisonburg, Virginia 22807, 2000.

[109] M. Ilyas and I. Mahgoub, *Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems*. CRC, July 2004.

[110] J. S. Baras and T. Jiang, *Cooperation, Trust and Games in Wireless Networks*. Advances in Control, Communication Networks, and Transportation Systems, Birkhuser Boston, 2005.

[111] M. E. Yildiz, A. Scaglione, and A. Ozdaglar, "Asymmetric information diffusion via gossiping on static and dynamic networks," in *in Proceedings of 49th IEEE Conference on Decision and Control, Atlanta, GA, USA*, pp. 7467–7472, 2010.

[112] C. Castellano, S. Fortunato, and V. Loreto, "Statistical physics of social dynamics," *Rev. Mod. Phys.*, vol. 81, pp. 591–646, May 2009.

[113] D. J. Stilwell, E. M. Bollt, and D. G. Roberson, "Sufficient conditions for fast switching synchronization in time-varying network topologies," *SIAM Journal on Applied Dynamical Systems*, vol. 5, pp. 140–156, 2006.

[114] Z. Toroczkai and H. Guclu, "Proximity networks and epidemics," *Physica A: Statistical Mechanics and its Applications*, vol. 378, no. 1, pp. 68 – 75, 2007.

[115] J. D. Skufca and E. M. Bollt, "Communication and synchronization in disconnected networks with dynamic topology: Moving neighborhood networks," *Mathematical Biosciences and Engineering*, vol. 1, no. 2, July 2004.

[116] J. P. Sethna, K. Dahmen, S. Kartha, J. A. Krumhansl, B. W. Roberts, and J. D. Shore, "Hysteresis and hierarchies: Dynamics of disorder-driven first-order phase transformations," *Phys. Rev. Lett.*, vol. 70, pp. 3347–3350, May 1993.

[117] S. Solomon, G. Weisbuch, L. de Arcangelis, N. Jan, and D. Stauffer, "Social percolation models," *Physica A*, vol. 277, pp. 239–247, 2000.

[118] D. J. Watts, *Small Worlds: The Dynamics of Networks Between Order and Randomness*. Princeton Univ. Press, Princeton, 1999.

[119] I. Chattopadhyay and A. Ray, "Renormalized measure of regular languages," *International Journal of Control*, vol. 79, no. 9, pp. 1107–11117, 2006.

[120] A. Tahbaz-Salehi and A. Jadbabaie, "Consensus over ergodic stationary graph processes," *IEEE Transactions on Automatic Control*, vol. 55, no. 1, pp. 225–230, 2010.

[121] S. N. Dorogovtsev, A. V. Goltsev, and J. F. F. Mendes, "Critical phenomena in complex networks," *Rev. Mod. Phys.*, vol. 80, pp. 1275–1335, Oct 2008.

[122] J. Liu, V. Yadav, H. Sehgal, J. M. Olson, H. Liu, and N. Elia, "Phase transitions on fixed connected graphs and random graphs in the presence of noise," *IEEE Transactions on Automatic Control*, vol. 53, pp. 1817–1825, Sept 2008.

[123] D. J. Watts, "A simple model of global cascades on random networks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 99, pp. 5766–5771, April 2002.

[124] S. Sarkar, K. Mukherjee, A. Srivastav, and A. Ray, "Distributed decision propagation in mobile agent networks," in *Proceedings of Conference on Decision and Control*, (Atlanta, GA, USA), 2010.

[125] M. C. Gonzleza, P. G. Linda, and H. J. Herrmann, "Networks based on collisions among mobile agents," *Physica D: Nonlinear Phenomena*, vol. 224, no. 1-2, pp. 137 – 148, 2006.

[126] J. Choi, S. Oh, and R. Horowitz, "Cooperatively learning mobile agents for gradient climbing," in *Proceedings of the 46th IEEE Conference on Decision and Control*, (New Orleans, Louisiana USA), December 2007.

[127] S. Patterson, B. Bamieh, and A. El Abbadi, "Convergence rates of distributed average consensus with stochastic link failures," *IEEE Transactions on Automatic Control*, vol. 55, no. 4, pp. 880–892, 2010.

[128] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Gossip algorithms: Design, analysis and applications," in *in Proceedings of IEEE INFOCOM*, pp. 1653–1664, 2005.

[129] Y. Wen, A. Ray, I. Chattopadhyay, and S. Phoha, "Vector space formulation of probabilistic finite state automata," in *in Proceedings of American Control Conference, San Francisco, CA, USA*, 2011.

[130] Y. Wen, A. Ray, I. Chattopadhyay, and S. Phoha, "Modeling of symbolic systems: Part ii - hilbert space construction for model identification and order reduction," in *in Proceedings of American Control Conference, San Francisco, CA, USA*, 2011.

[131] B. Nabet, N. E. Leonard, I. D. Couzin, and S. A. Levin, "Dynamics of decision-making in animal group motion," *Journal of Nonlinear Science*, vol. 19, pp. 399–435, 2009.

[132] T. Kobayashi and D. L. Simon, "A hybrid neural network-genetic algorithm technique for aircraft engine performance diagnostics," in *37th Joint Propulsion Conference and Exhibit cosponsored by the AIAA, ASME, SAE, and ASEE*, (Salt Lake City, Utah), 2001.

[133] P. M. Koushanfar, F. and A. Sangiovanni-Vincentelli, "On-line fault detection of sensor measurements," *IEEE Sensors*, pp. 974–980, 2003.

[134] C. Tian, W. Ding, R. Cao, and S. Jiang, "Extensive epidemic spreading model based on multi-agent system framework," in *ICCS '07: Proceedings of the 7th international conference on Computational Science, Part IV*, (Berlin, Heidelberg), pp. 129–133, Springer-Verlag, 2007.

[135] L. Bayinder and E. Sahin, "A review of studies in swarm robotics," *Turk Journal of Electric Engineering*, vol. 15, pp. 115–147, 2007.

[136] R. Chen, D. Chen, and S. Lin, "Actam: Cooperative multi-agent system architecture for urban traffic signal control," *IEICE transaction Information and Systems*, vol. E88D, pp. 119–126, January 2005.

[137] N. Vlassis, *A Coincise Introduction to Multiagent Systems and Distributed Artificial Intelligence*. Morgan and Claypool Publishers, 2007.

[138] W. Spears, D. Spears, J. Hamann, and R. Heil, "Distributed, physics-based control of swarms of vehicles," *Autonomous Robots*.

[139] E. Sahin and O. Soysal, "Probabilistic aggregation strategies in swarm robotic systems," in *Proc. of the IEEE Swarm Intelligence Symposium*, (Pasadena, California, USA), 2005.

[140] D. Payton, R. Estkowski, and M. Howard, "Pheromone robotics and the logic of virtual pheromones,"

[141] S. Nouyan, "Path formation and goal search in swarm robotics," dea thesis, University Libre de Bruxelles, Belgium, September 2004.

[142] E. Bahceci and E. Sahin, "Evolving aggregation behaviors for swarm robotic systems: A systematic case study," in *Proc. of the IEEE Swarm Intelligence Symposium*, (Pasadena, California, USA), 2005.

[143] A. Hayes and P. Dormiani-Tabatabaei, "Self-organized flocking with agent failure: Off-line optimization and demonstration with real robots," in *Proc. of the 2002 IEEE Int. Conf. on Robotics and Automation*, (Washington DC, USA), pp. 3900–3905, 2002.

[144] A. Ilachinski, *Cellular Automata: A Discrete Universe*. World Scientific Publishing, Singapore, 2001.

[145] R. Brooks, "A robust layered control system for a mobile robot," *IEEE Journal of Robotics and Automation*, vol. 2, no. 1, pp. 14–23, 1986.

[146] O. Katai, K. Toda, and H. Kawakami, "Decentralized control of multi-agent systems based on modal logics and extended higher order petri nets," in *AMT '01: Proceedings of the 6th International Computer Science Conference on Active Media Technology*, (London, UK), pp. 182–190, Springer-Verlag, 2001.

[147] M. Hsieh, V. Kumar, and L. Chaimowicz, "Decentralized controllers for shape generation with robotic swarms," *Robotica*, vol. 26, pp. 691–701, 2008.

[148] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*. New York, USA: Oxford Unversity Press, 1999.

[149] T. Horiguchi, H. Takahashi, K. Hayashi, and C. Yamaguchi, "Ising model for packet routing control," *Physics Letters A*, vol. 330, pp. 192–197, sep 2004.

[150] E. Alpaydin, *Introduction to Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2004.

[151] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*. The MIT Press, 1998.

[152] C. Watkins, "Learning from delayed rewards," thesis, University of Cambidge, England, 1989.

[153] K. Ito and A. Gofuku, "Hybrid autonomous control for heterogeneous multiagent system," in *Proceedings of the IEEWRSJ Intl. Conference on Intelligent Robots and Systems*, (Las Vegas, Nevada), 2003.

[154] T. Alpcan, P. Wang, P. G. Mehta, and T. Basar, "A non-equilibrium analysis and control framework for active queue management," *Automatica*, vol. 44, no. 10, pp. 2474–2486, 2008.

[155] R. Montemanni and L. Gambardella, "Swarm approach for a connectivity problem in wireless networks," in *Proceedings of the IEEE Swarm Intelligence Symposium*, (Pasadena, U.S.A.), pp. 265–272, 2005.

[156] M. Mataric, "Reinforcement learning in the multi-robot domain," *Autonomous Robots*, vol. 4, pp. 73–83, 1997.

[157] V. Trianni, R. Grob, T. Labella, and M. Dorigo, "Evolving aggregation behaviors in a swarm of robots," in *Proceedings of the Seventh European Conference on Artificial Life, volume 2801 of Lecture Notes in Artificial Intelligence*, pp. 865–874, Springer Verlag, 2003.

[158] I. Chattopadhyay and A. Ray, "Language-measure-theoretic optimal control of probabilistic finite-state systems," *International Journal of Control*, vol. 80, no. 8, pp. 1271–1290, 2007.

[159] V. Garg, "Probabilistic languages for modeling of deds," in *Proceedings of IEEE Conference on Information and Sciences*, (Princeton, NJ, USA), pp. 198–203, March 1992.

[160] W. Rudin, *Real and Complex Analysis*. 3rd ed., McGraw Hill, New York, 1988.

[161] V. Garg, "An algebraic approach to modeling probabilistic discrete event systems," in *Proceedings of IEEE Conference on Decision and Control*, (Tucson, AZ, USA), p. 23482353, December 1992.

Vita

Soumik Sarkar

Soumik Sarkar was born in West Bengal, India on November 21, 1983. He received his Bachelor of Engineering Degree in Mechanical Engineering in 2006 from Jadavpur University, Kolkata, India. He spent summer of 2004 and 2005 as a Research Fellow in IISc, Bangalore and JNCASR, Bangalore respectively. He received M.S. in Mechanical Engineering and M.A. in Mathematics in 2009 from Penn State University, USA. He continued his doctoral research in Mechanical Engineering simultaneously under the guidance of Professor Asok Ray and successfully defended his thesis on May 20, 2011. His research interests include Cyber-physical systems, Signal Processing and Pattern Recognition, Sensor Fusion, Fault Detection in Aerospace and Energy systems, Control of Complex Networks and Analysis of Critical Phenomena in Complex Engineering Systems. At Penn State, he successfully participated in multi-disciplinary teams for multiple projects sponsored by NASA, ONR, ARO and DOE. He coauthored more than ten journal and more than fifteen refereed conference papers. He received Two Best Session Paper Presentation Awards in 2009 American Control Conference. He co-instructed three multi-disciplinary graduate level courses at Penn State. He has also served as a reviewer for several technical journals and conferences.