PROBLEMS IN DECENTRALIZED DECISION MAKING AND COMPUTATION

by

John Nikolaos Tsitsiklis

B.S., Massachusetts Institute of Technology
(1980)

S.M., Massachusetts Institute of Technology
(1981)

SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE
DEGREE OF

DOCTOR OF PHILOSOPHY

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

November 1984

© Massachusetts Institute of Technology 1984

Signature of the Author .....
Department of Electrical Engineering and Computer
Science, November, 1984

Certified by .....
Michael Athans
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Departmental Graduate Committee

PROBLEMS IN DECENTRALIZED DECISION MAKING AND COMPUTATION

by

John Nikolaos Tsitsiklis

Submitted to the Department of Electrical Engineering and Computer Science on November 20, 1984 in partial fulfillment of the requirements for the Degree of Doctor of Philosophy.

ABSTRACT

We investigate certain fundamental problems in decentralized decision making and computation. We study the problem of whether a set of decision makers (or processors) with different (but related) information may make compatible decisions without communication and we characterize the computational complexity of this problem. We also analyze the complexity of other basic problems of decentralized decision making, such as decentralized detection (hypothesis testing).

We then consider a scheme whereby a set of decision makers (processors) exchange and update tentative decisions which minimize a common cost function, given the information they possess; we show that they are guaranteed to converge to consensus.

Finally, we consider a broad class of asynchronous distributed (deterministic and stochastic) iterative optimization algorithms tolerating communication delays. We associate communication requirements with such algorithms and show that they converge appropriately under certain conditions which are no more severe than those required for their centralized counterparts.

Several applications in human organizations, parallel computation and distributed signal processing are indicated.

Thesis Supervisor: Dr. Michael Athans
Title: Professor of Systems Science and Engineering

ACKNOWLEDGMENTS

I greatly acknowledge the invaluable guidance and constant support of my thesis supervisor, Professor Michael Athans, whose energy and enthusiasm have helped to shape this thesis.

I am also greatly thankful to my thesis reader, Professor Dimitri Bertsekas whose ideas had an important role in a major part of this thesis and who was also an invaluable source of support.

Professor Robert Tenney and Dr. Hans Witsenhausen who were also thesis readers, deserve special thanks for many discussions and their insights. Let me also thank Professor Richard Dudley for suggesting a proof of Lemma 4.2.1.

Part of Chapter 3 of this thesis was shaped through collaboration with Professor Christos Papadimitriou, but his support and cooperation has extended far beyond it and he deserves special thanks.

Throughout my years at MIT I have had the pleasure of interacting with several other people who have thus had a direct influence on my education and I would like to thank them all: Dr. David Castanon, Dr. Stanley Gershwin and Professors Bernard Levy, Sanjoy Mitter, Fred Schweppe, George Verghese and John Wyatt.

I am deeply thankful to Ms Fifa Monserrate for her excellent typing as well as her patience when it came to revising the manuscript. Thanks are also due to Mr. Arthur Giordani for his fine art work.

TABLE OF CONTENTS

		PAGE
1.	INTRODUCTION	1
	1.1 Problem Definition	1
	1.2 Outline and Overview	4
	1.3 Contributions of this Report	7
2.	A DISCUSSION OF DECENTRALIZED SYSTEMS	10
	2.1 Decentralized Systems and their Design	10
	2.2 On Real World Organizations	21
	2.3 On Communications and Associated Problems	23
	2.4 Literature Survey	24
3.	DISCRETE DECENTRALIZED DECISION PROBLEMS	34
	3.1 Introduction and Motivation	34
	3.2 A Problem of Silent Coordination	36
	3.3 Decentralized Detection	53
	3.4 Related Problems	62
	3.5 On Designing Communications Protocols	64
	3.6 Conclusions	68
4.	CONVERGENCE AND ASYMPTOTIC AGREEMENT OF PROCESSORS INTERESTED IN A COMMON DECISION	71
	4.1 Introduction and Motivation	71
	4.2 Model Formulation	77
	4.3 Convergence and Agreement Results	85
	4.4 The Linear Quadratic Gaussian (LQG) Model	96
	4.5 A Model Involving a Coordinator	110
	4.6 Processors with Different Models	113
	4.7 Conclusions	117
5.	DECENTRALIZED DETERMINISTIC AND STOCHASTIC ITERATIVE ALGORITHMS	118
	5.1 Introduction and Overview	118
	5.2 A Model of Decentralized Computation	121
	5.3 Decentralized Gradient-like Algorithms	140
	5.4 Decentralized Stochastic Algorithms with Correlated Noise	171
	5.5 Applications in System Identification	207
	5.6 Decentralized Gradient Algorithm for an Additive Cost Function	224

PAGE

5.7 Towards Organizational Design 234
5.8 Other Possible Applications 238
5.9 Summary of Problems for Future Research 241
5.10 Summary and Conclusions 243

6. A GLOBAL VIEW 245

6.1 Overview 245
6.2 An Organizational View 246
6.3 General Areas for Future Research 247

APPENDIX 248

A. Proof of Lemma 5.2.1 248

REFERENCES 256

LIST OF FIGURES

		PAGE
FIGURE 2.1.1	A Decentralized System with Two Control Modules	12
3.2.1	An Information Structure with $D_1=D_2=2$	46
3.3.1	A Structure for Decentralized Detection	53a
4.4.1	Mean Square Errors of Decomposition Algorithms, for Covariance Σ_w; Dashed Line Indicates the Optimal Mean Square Error.	108
4.4.2	Mean Square Errors of Decomposition Algorithms, for Covariance Σ_w+I; Dashed Line Indicates the Optimal Mean Square Error.	109
5.5.1	Two Processors Identifying Identical ARMA Systems	209
5.5.2	Two Processors Identifying the same Moving Average System	210
5.5.3	Two Systems Driven by a Common Colored Noise.	221
5.6.1	The Graph Associated to an Additive Cost Function.	227
A.1	Reduction to the zero delay case.	250

CHAPTER 1: INTRODUCTION

1.1 PROBLEM DEFINITION

The subject of this report is the investigation of certain central problems in decentralized (distributed)* decision making and computation.

Classical (centralized) decision making theory deals with the situation in which a single decision maker (DM) (man or machine) possesses all available knowledge and information related to a certain system and has to make a decision (or a sequence of decisions) so as to achieve a certain objective (minimize a cost criterion, for example).

Many real world systems (power systems, public or business organizations, large manufacturing systems, etc.), however, are too large for the classical model of decision making to be applicable. There may be a multitude of decision makers (or processors), none of which possesses all relevant knowledge. In addition, there may be limitations on the amount of communications allowed between distinct decision makers, so that it is impractical to exchange all available information and convert the problem to a centralized one. In fact, even if there exists infinite capacity for communications, still centralization may be inadvisable, because no decision maker may have the capability of tackling the overall problem by himself. The above reasons make decentralization necessary, whereby several decision makers make their own decisions, based on partial information, possibly by solving a problem related to the original one. This raises the need to structure the decentralized decision process so that the outcome of the joint effort of the various decision makers achieves, in some sense, the goal of the overall system (organization).

From the point of view of the theory of computation, "centralized" theory deals

*The two terms are assumed in this thesis to have the same meaning and will be used interchangeably. The term "decentralized" seems preferable, in general, because "distributed" is often associated to systems governed by partial differential equations.

with the case in which a single serial processor is to execute a sequence of instructions, in order to evaluate a desired result (value). In decentralized computation*, the same goal is achieved by a set of processors operating in parallel and exchanging partial results. Parallel computation is advantageous in many situations, because the desired final result may be evaluated much faster, or because the input data of the computation are physically distributed (for example, if the computation consists of statistical processing of data acquired by physically distinct sensors). Interesting problems arise in this context because good parallel algorithms can be very different from simple adaptations of good serial algorithms.

Concerning decision making problems, especially in human organizations, it is often the case that distinct decision makers have different objectives, which are also different from the objective of the organization. This may lead to conflict and to situations best addressed by game theory. We will restrict, however, to situations in which:

a) There is a well-defined organizational objective.

b) The individual decision makers are either physical processors (so that no interest may be ascribed to them) or they may be treated - for the purpose of analysis - as if they were processors with predictable behavior. For example, a human decision maker may make decisions motivated by his perceived self-interest; however, if an analyst knows the perceived self-interest of that decision maker, he may be able to predict his behavior. From that point on, self-interests become irrelevant: as far as analysis is concerned, the human decision maker may be modelled as a processor, operating in a specific way.

*In this context, the terms "parallel" or "distributed" computation are often used.

Given such a setting, this report aims at the better understanding of the limitations and capabilities of decentralized systems. Clearly, any issues pertinent to centralized decision problems may arise in decentralized problems as well. Our goal is, however, to focus on issues which are unique to decentralized systems, the predominant ones being the effects of communications, or of the impossibility of communications.

Our work does not center around any particular application. The selection of the problems to be addressed is guided, however, by applications in:

a) Decentralized (parallel) computation.

b) Human Organizations.

c) Decentralized Signal Processing.

In some respects, there is little in common between so diverse applications. On the other hand, they all involve decentralization. If an abstract study may find applications in more than one of the above areas, it is legitimate to claim that a common denominator behind a variety of decentralized systems has been addressed. This is not an elusive goal: for example, Arrow and Hurwicz [1960] have indicated common features of decentralized computation and the operation of economic markets.

We close this section with a remark on terminology. Depending on the context, the "entities" in a decentralized system which perform computations or make decisions, are often called processors, decision makers, modules or agents. Each of the above terms often carries certain connotations, which we wish to avoid. In particular, using the term "decision maker" often implies the existence of an individual interest, on the basis of which decisions are being made. If, however, we have a human who simply does exactly what he has been told to do (as if he were programmed), the term

"agent" seems to be more appropriate. The latter term, however, sounds unnatural in a situation involving only computing machines. For these reasons, we prefer to use the neutral terms "processor" or "module", with the understanding that, in some occasions, they may refer to humans who process information and apply decisions in a prescribed manner.

1.2 OUTLINE AND OVERVIEW

In this section we outline the contents of this report, with the main goal of highlighting the unity and continuity of the chapters that follow.

Chapter 2 addresses certain conceptual issues related to decentralized systems. Starting with a general definition of decentralized systems, we indicate some typical features that may be present. We then discuss the nature of associated design problems, pointing to the possibility of having a decentralized system designing another decentralized system.

We then continue with an abstract discussion and a schematic history of actual organizations, which serves as a motivation for some of the problems to be studied in the sequel, together with a brief discussion of the types of mathematical problems raised by the possibility of communications, motivating again the work that follows. Chapter 2 ends with a survey of some of the literature related to decentralized systems.

Chapter 3 deals with the simplest but fundamental problems of decentralized decision making, exploring the effect of communication constraints.

The first problem we raise is the following: given a set of processors with partial information, is it possible that they make satisfactory decisions (in a

certain sense), without communicating? The next problem, which follows logically from the preceding one, is: if it turns out that they have to communicate, what is the least amount of communications required? We pose these problems in a simple setting in which the sets of possible observations and decisions are finite. We show, however, that these are hard combinatorial problems. We also investigate a variety of special cases and versions of the basic problems, with the goal of determining the boundary between easy and hard problems. Special attention is paid to the well-known problem of decentralized hypothesis testing (detection).

The results of Chapter 3 lead to the conclusion that the optimal design of a decentralized system, even in the absence of any dynamics, is very hard computationally. The problem "who should communicate to whom, what, when, etc." cannot be addressed, for all practical purposes, with the goal of optimizing within a most general class of communication protocols. Rather, we have to restrict to smaller, more tractable, classes of communication protocols. To which particular class of protocols one chooses to restrict, unescapably contains a certain degree of arbitrariness.

With this point of view, we consider, in Chapter 4, a particular protocol concerning a set of processors with identical prior information (Bayesian models), different posterior (on-line) observations and identical cost functions. These processors are assumed to exchange (asynchronously) tentative decisions (that is, decisions which are optimal given the information they possess). We show that the decisions of the processors will asymptotically converge to consensus even if they have limited memory and forget some of their past information, in the course of the process of exchanging tentative decisions. We show that this scheme leads to a decomposition algorithm for static linear estimation problems. We also discuss briefly what could happen if the decision makers had different cost functions and/or models (prior probabilities).

The scheme of Chapter 4 will be often hard to implement, because at each stage each processor has to evaluate an optimal (tentative) decision, given its information, which may be a hard non-linear problem. For this reason, we take in Chapter 5 a more realistic approach, more applicable to human decision makers with bounded rationality (cognitive limitations) or computing machines with limited capabilities: each processor makes tentative decisions which are communicated to other processors; concurrently with the process of exchanging messages, each processor updates its tentative decision. In contrast with Chapter 4, however, these updates are not optimal in a Bayesian sense; rather, they are small updates in a direction which is expected to improve performance.

The scheme of Chapter 5, may be viewed from several different perspectives: as a decentralized algorithm for solving an optimization problem, in which case tentative decisions are local states of computation; alternatively, as a process of adjustment of human decision makers in a divisionalized organization. Of course, the different perspectives concern the interpretation of the results, not the mathematical analysis.

We now outline the contents of Chapter 5, in more technical terms. We first develop a suitable model of decentralized computation and then proceed to prove convergence results for a broad class of asynchronous decentralized (deterministic and stochastic) optimization algorithms, tolerating communication delays. These include constant step-size algorithms (e.g. gradient-type deterministic optimization algorithms), as well as decreasing step-size algorithms (e.g. stochastic approximation algorithms).

The conditions for convergence typically contain requirements on the frequency of communications between processors. These conditions may be interpreted as

guidelines for designing the communication flows in a decentralized system (be it a human organization or a parallel computer) in a way that guarantees smooth operation of the system. We also apply our results in algorithms for decentralized identification of dynamical systems.

The last chapter contains an overview of our study, some conclusions and suggests future research directions.

1.3 CONTRIBUTIONS OF THIS REPORT

In this Section we list the contributions of Chapters 3,4 and 5 which contain our technical results.

Chapter 3

° We show that the discrete versions of the basic (static) problems of decentralized decision making (including the well-known team-decision problem [Marschak and Radner, 1972]) are algorithmically hard (NP-complete, or worse), even though the corresponding centralized problem is trivial. We also obtain complexity results for several special cases.

° We show that the problem of decentralized hypothesis testing [Tenney and Sandell, 1981] is algorithmically hard, if a certain simplifying assumption of Tenney and Sandell is removed.

° We show that the problem of designing an optimal communications protocol is algorithmically hard, even in the simplest setting.

Chapter 4:

° We show that a set of processors (with identical models and cost function, but different on-line information) who exchange optimal (given their information) decisions will asymptotically converge to consensus, under certain assumptions. This is true even if the processors have limited memory, so that they may forget some of the information they had acquired during the operation of this scheme. These results generalize significantly earlier results of Aumann [1976], Geanakopoulos and Polemarchakis [1978], Borkar and Varaiya [1982].

° We obtain a new decomposition algorithm for solving linear estimation problems and prove that it converges exponentially.

Chapter 5:

° We develop a model for asynchronous decentralized computation which generalizes an earlier model of Bertsekas [1982,1983]. This model allows different processors either to specialize in updating a component assigned to them; or, they may "overlap" so that many of them update the same component of a decision vector. A novel feature of this model is that it allows us to associate with the decentralized algorithm an aggregate state of computation.

° We prove convergence (to the centralized optimal) of asynchronous decentralized versions of a broad class of deterministic and stochastic optimization algorithms (with either constant or decreasing step-size) under conditions which are not significantly stronger than those required for the centralized counterparts of our results [Poljak and Tsypkin, 1973].

° We prove convergence results for decentralized stochastic algorithms driven by correlated noise, analyzed via the ODE approach [Ljung, 1977a].

° We study gradient-type deterministic optimization algorithms for an additive cost function in which a different processor is assigned to a different term of the cost function. We show that convergence is guaranteed if the frequency of communications between two processors is proportional to the degree of coupling between their subproblems.

° We apply our results and prove convergence of certain decentralized algorithms for identification of dynamical systems.

° Leaving technical details aside, a main contribution of Chapter 5 is that a novel way has been found to associate communication requirements with the smooth operation of a class of decentralized systems.

CHAPTER 2: A DISCUSSION OF DECENTRALIZED SYSTEMS

In this Chapter we discuss certain general issues concerning decentralized systems. Our discussion has to be limited, however, due to the fact that there is a great variety of real world decentralized systems. For example, a decentralized controller for a large power system, a large company or a special purpose multiprocessor, have enough differences so as to render a unified analysis impossible. On the other hand, very different systems may have substantial similarities, so that an abstract viewpoint may be useful. (See [Arrow and Hurwicz, 1960], for example.)

In Section 2.1 we start with a definition of a decentralized system (in order to fix the terminology) and discuss some qualitative features that may be present. We comment on the nature of the problem of designing a decentralized system and suggest that the design may be carried out by another decentralized system which is distinct from the first.

Section 2.2 takes a brief and schematic look at existing organizations, so as to identify some types of problems that may be addressed.

Section 2.3 focuses on the role of communications, since these are the cause of the main differences between centralized and decentralized systems. This Section, together with the preceding one, serves as an abstract motivation for the Chapters that follow.

Section 2.4 surveys some of the literature related to decentralized systems.

2.1 DECENTRALIZED SYSTEMS AND THEIR DESIGN

What is a Decentralized System.

We provide below a fairly general definition of what constitutes a (discrete-time) decentralized system. This definition is intended primarily to provide a

language for the discussion that follows. For this reason, we avoid the discussion of continuous-time decentralized systems which may lead to non-trivial well-posedness questions, which need not concern us here.

From an abstract mathematical perspective, a decentralized system is simply an interconnected system in which we give certain particular interpretations to the interconnection variables involved. Formally, we have (see Figure 2.1.1):

1. An environment module M_0 and a set $V = \{M_1,\ldots,M_N\}$ of control modules.
2. A directed graph $G=(V,E)$ whose nodes are the control modules. The arcs in this graph indicate the allowed directions of communications.
3. For any $(i,j) \in E$, let M_{ij} be a channel module, through which messages from M_i to M_j are being transmitted.
4. Let $x_i(t)$, $x_{ij}(t)$ denote the state of module M_i, M_{ij}, respectively, at time t, (t belongs to a discrete index set T) assumed to lie in some set (state space) X_i, X_{ij}, respectively.
5. Let $u_i(t)$ denote the control (decision) applied by module M_i, $i \neq 0$, on the environment, at time t, assumed to lie in some set U_i.
6. Let $y_i(t)$ denote the observation (measurement) on the environment, obtained at time t by module M_i, $i=0$, assumed to lie in some set Y_i.
7. Let w_i, w_{ij} be disturbances which affect the operation of modules M_i, M_{ij}, respectively, belonging to sets W_i, W_{ij}, respectively. Here w_o captures the uncertainty in the environment; w_i, $i \neq 0$, captures the uncertainty in the inner operation of control module M_i; w_{ij} captures the uncertainty in the communication process (message distortions and delays). (Following the conventions of

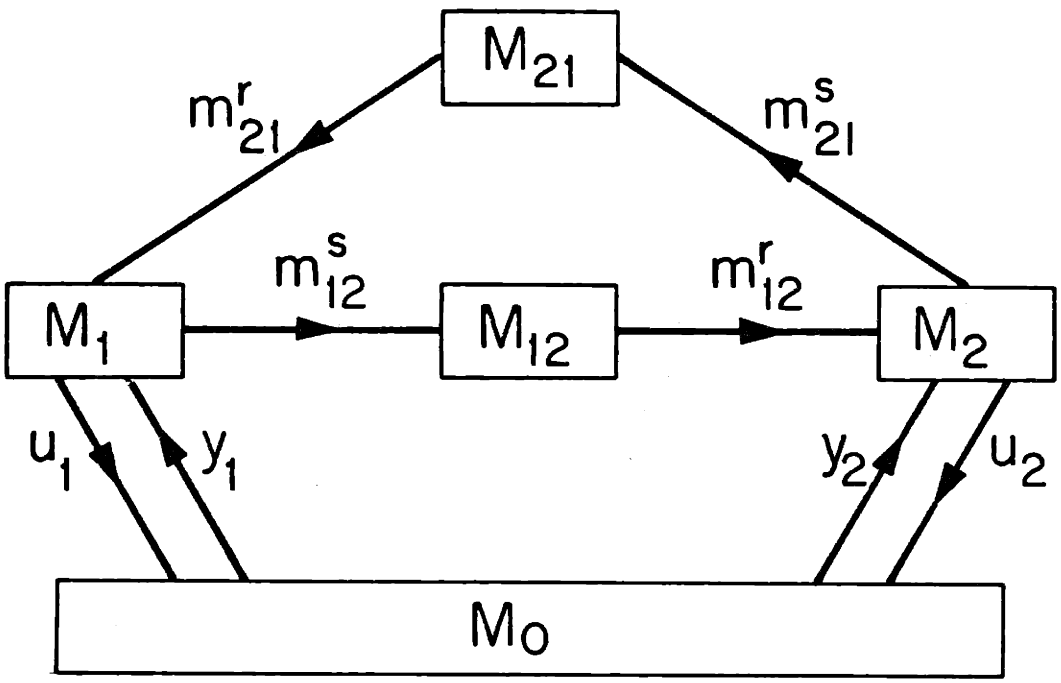

Figure 2.1.1: A Decentralized System with Two Control Modules.

probability theory, the disturbances are not indexed by time; so the sets W_i, W_{ij} are most likely to be sets of sample paths. Of course, no probabilistic assumptions are made at this point.)

8. Let m^s_{ij}, m^r_{ij}, $(i,j)\in E$, denote the messages sent from M_i to M_j at time t, and received by M_j from M_i, at time t, respectively. These are assumed to belong to sets M_{ij}.

9. Finally, we have certain relations between the above introduced variables which specify the evolution of the internal states of each module as well as the interactions between modules:

$$x_o(t+1) = \Phi_o(t,x_o(t),u_1(t),\ldots,u_N(t),w_o) \tag{2.1.1}$$

$$x_i(t+1) = \Phi_i(t,x_i(t),u_i(t),y_i(t),m^r_{1i}(t),\ldots,m^r_{Ni}(t),w_i),\quad i\neq 0, \tag{2.1.2}$$

$$x_{ij}(t+1) = \Phi_{ij}(t,x_{ij}(t),m^s_{ij}(t),w_{ij}),\quad i,j\neq 0, \tag{2.1.3}$$

$$u_i(t) = \psi_{io}(t,x_i(t),w_i),\quad i\neq 0, \tag{2.1.4}$$

$$y_i(t) = \psi_{oi}(t,x_o(t),u_i(t),w_o),\quad i\neq 0, \tag{2.1.5}$$

$$m^s_{ij}(t) = \psi^s_{ij}(t,x_i(t),y_i(t),u_i(t)),\quad i,j\neq 0, \tag{2.1.6}$$

$$m^r_{ij}(t) = \psi^r_{ij}(t,x_{ij}(t),m^s_{ij}(t),w_{ij}),\quad i,j\neq 0, \tag{2.1.7}$$

Equations (2.1.1)-(2.1.7) implicitly assume the following sequence of events at time t:

(i) Controls are applied.

(ii) Observations are obtained.

(iii) Messages are transmitted.

(iv) Messages are received.

(v) Next states are formed.

(Of course, the above assumed sequence of events is a matter of convention.)

General Features of Decentralized Systems

We now discuss briefly some features which may (but need not) be present in a decentralized system.

1. The environment may be modelled as an interconnected system, or it may be completely absent. If it is an interconnected system, its topology may coincide or may be different than the topology of the interconnections of the control modules. The case of coinciding topologies creates some additional structure, which may be mathematically exploited, and has been emphasized in the literature; however, such a coincidence is not a logical necessity.

2. The controls applied by the control modules may influence the evolution of the state of the environment. They may also influence the observations directly, via equation (2.1.5). This allows us to capture actions of a special type which initiate a measurement process, without affecting the environment. In other words, the action of a control module may determine whether a measurement will be obtained or not, and of what quality.

3. Disturbances w_i on control modules may reflect processor failures, rounding-off errors or bounded rationality, when the modules represent humans.

4. If at time t no observation is made by M_i, we may let $y_i(t)$ be equal to a special symbol indicating this absence. So, it is not required that an observation be made at each time instance. The same convention may be applied concerning message transmissions and receptions.

5. Observations obtained or messages received by module M_i may be remembered or forgotten. (Such effects can be captured by (2.1.2).)

6. The channel modules M_{ij} are often fairly simple. For example, a time delay together with some uncertain distortion (channel noise).

7. Equations (2.1.4)-(2.1.7) have been written so that - if there are no communication delays and no channel noise - any observation obtained by module M_i may become immediately available to any other module M_j. In that case, all modules may make the next decision on the basis of common information.

8. A message $m_{ij}^r(t)$ may influence $x_j(t+1)$ which in turn may influence $m_{ij}^s(t+1)$. This allows us to model a situation in which a message contains a request for certain pieces of information.

9. A control module may have certain special state variables which remain constant (equal to their initial values) but nevertheless influence the evolution of the remaining state variables. Such special variables may be viewed as parametrizations, "set-points" of a controller or, for modules representing humans, they may capture the capabilities, beliefs or prior knowledge of human decision makers.

10. The local disturbances w_i to control module M_i may influence the time at which certain messages are transmitted or certain actions are undertaken. This allow us to model systems which operate without any strict synchronization.

11. The evolution functions Φ_i of either the environment or the control modules may be dynamic (next state depends on current one) or not.

Classification of Decentralized Systems

We saw above that a variety of qualitative phenomena may arise in a decentralized system. If all of them are concurrently present, no meaningful analysis seems possible. For this reason, it may be useful to classify decentralized systems in a qualitative way. A general classification is possible along the following lines, starting from a higher and proceeding to a lower level:

(i) According to the topology of the modules.

(ii) According to the presence or absence of certain variables.

(iii) According to which variables explicitly influence other variables.

(iv) According to the nature of these influences.

The Performance of a System

We assume that a system exists in order to accomplish some task, perform some function or control the environment in a specific way; that is, there exists a well-defined organizational objective. Otherwise, no design-oriented analysis would be meaningful. This does not exclude the possibility that certain control modules have their own interests, or that they have incorrect models of the other modules. Nevertheless, no matter what each module "believes" to be true, the system's analyst has to postulate the existence of a true, objective model which describes the evolution of the system. Moreover, this evolution is to be compared to a "desired" one, to see whether certain performance criteria are met. What may be desirable from the analyst's perspective may be quite distinct from what is desirable by the modules.

There is an abundance of qualitatively different ways of measuring the performance of a system. For example:

1. The performance criterion may depend on the temporal evolution of the state variables of the environment and/or control modules. In particular, it may depend on the entire history of the state variables (e.g. control theory with additive costs), it may be asymptotic (e.g. the stability requirement in adaptive control) or it may depend only on a final state (e.g. the performance of a parallel algorithm may be judged in terms of the termination time).
2. A cost may be incorporated which is related to the size of the state spaces of the control modules (so as to reflect the cost of memory) or to the complexity of the evolution equations of the control modules (to reflect computation costs during implementation).
3. There may be costs depending on the values of the controls being applied (e.g. a penalty on energy being used), or costs of obtaining measurements.
4. Finally, there may be costs associated to the process of communications. For example, each message transmitted could result to a penalty, possibly depending on the identities of the transmitter and the receiver, as well as on the length of the message.

Note that the above mentioned aspects 1,2,3 of the cost criterion are also relevant to systems with a single control module. So, it is the fourth aspect which should be expected to lead to qualitatively new issues.

Typically, not all of the design issues are included in the performance criterion. Rather, a design is performed in terms of a narrow criterion and, then, the designer has to check whether certain side-concerns are adequately handled. Some common side-issues are related to robustness, sensitivity and adaptivity to small and/or structural unmodelled variations.

On Design Problems

We start with the premise that we are not just interested in analyzing the performance of a given, fixed decentralized system; rather, we want to design one. We must therefore distinguish those elements of the system which are given from those which are amenable to design.

Formally, a system S may be viewed as an ordered pair (α,β), where α,β represent the fixed and variable elements of the system, respectively. We assume that, for any α, we are given a set $B(\alpha)$ of admissible choices of the manipulable parts of the system. It is often helpful to think about many systems simultaneously, which motivates the next definition:

Definition 2.1.1: A decentralized scheme is a collection $\{(\alpha,\beta(\alpha)):\alpha\in A\}$ of decentralized systems, where β is some function $\beta:A \to B(\alpha)$.

Essentially, β yields an admissible design for any value of the fixed elements of the system. For example, if A is the set of linear systems, let the map β assign a stabilizing compensator to each $\alpha\in A$, according to a well-defined rule.

Note that, typically, the separation of a system into α and β-parameters follows a certain pattern:

(i) The environment dynamics, the effects of the controls, the environment disturbances and the properties of the channels are usually fixed.

(ii) The controller dynamics and the mechanisms generating messages and controls are usually free to be designed.

Let us now discuss design problems. Suppose that we are given a set A of parameters; for each $\alpha \in A$, a set $B(\alpha) \subset B$ of admissible designs; a performance criterion $J: A \times B \to R$. The following questions may be raised:

(i) Given α, β, what is the value of $J(\alpha,\beta)$?

(ii) For $\alpha=\alpha_o$, find $\beta^* \in B(\alpha_o)$ which minimizes $J(\alpha_o,\beta)$, over all $\beta \in B(\alpha_o)$.

(iii) Find $\beta^*: A \to B$ such that for any fixed $\alpha \in A, \beta^*(\alpha)$ solves problem (ii).

Let us now focus on questions (ii) and (iii). They are different in the same way that the following two problems are different: (a) Minimize $(\beta-2)^2$; (b) Minimize $(\beta-\alpha)^2$. Question (ii) corresponds to a case study: designing a specific system; question (iii) corresponds to designing a scheme. Although applications always concern specific systems, theoretical analysis, in order to be general and useful, must focus on questions of type (iii). We are thus led into the next issue: what constitutes an answer to question (iii)? A closed form representation or a listing of the values of β^* in a table is usually impossible. The only alternative left is for the answer to consist of a list of instructions that may be followed to construct $\beta^*(\alpha)$ from α; that is, an algorithm which evaluates β^*. The instructions of any such algorithm will consist of some elementary instructions (or operations) which are assumed to be readily implementable, say in the form of a subroutine call. (What we call elementary instructions here may be at a very high level when compared to the elementary instructions considered in the theory of computation.) For example,

the dynamic programming algorithm uses the instruction $u := \arg\min_u g(u)$ and the gradient algorithm for unconstrained optimization uses the instruction $\lambda := (\partial g/\partial u)(u)$. (So, the instruction assumed to be available for the solution of one problem may be the problem itself in another setting). Accordingly, what constitutes an answer to question (iii) depends strongly on the set of elementary instructions.

We argued above that a decentralized system (or scheme) is to be designed by an algorithm, possibly a distributed one. Notice that such algorithms are special cases of (decentralized) systems, under our definition. We obtain, therefore, a hierarchy:

(i) A system being implemented. (Lower level.)

(ii) A system which yields as final output the system to be implemented. (Higher Level.)

1. The two systems, at the different levels, are distinct and should not be confused, even though they are both related to the same situation.

2. Each of the systems in (i) and (ii) above may be either centralized or decentralized, thus leading to four different combinations.

3. It makes a large difference whether the lower level is centralized and the higher decentralized, or the converse is true. (In the context of linear control, the first case has been called "decomposition" and the second "decentralization" by Sandell [1976].)

4. If the systems at both levels are decentralized, the two topologies may coincide, although this is not necessary. (In most of the literature on hierarchical control [Mahmoud, 1977], the two topologies are assumed to be identical. This may

lead to some conceptual confusion, as it may be unclear which of the two decentralized systems is being referred to. Findeisen [1982] clarifies this distinction by talking about the "programming" and "execution" phases.)

5. The lower and higher level systems are interrelated: a) If we change the design problem for the lower level system, a different high level system (more complex or less complex) could be used. Typically, one is willing to sacrifice some performance for the lower level system, if this results to a less complex design algorithm (higher level system). However, such tradeoffs are often extremely hard to quantify. b) Once the higher level system has terminated its operation, its output must be transmitted to certain locations (the control modules of the lower level system). These transmissions may entail a certain cost which is not pertinent to the higher or lower level system, but rather to the fact that the second must start from where the first has stopped. c) It is also conceivable that the two systems referred to above, operate concurrently. For example, an organization may change its structure while operating.

2.2 ON REAL WORLD ORGANIZATIONS

There are many types of organizations but all of them are decentralized systems in our terminology. We choose to restrict to those organizations in which there are certain people who have a direct interest in the performance of the organization and also have substantial authority for making changes. In our terminology, they are faced with the problem of designing a decentralized system. This problem being often intractable, very few organizations have been structured according to a grand-design

which started from scratch. Rather, the design process was itself a decentralized system, possibly operating concurrently with the organization itself. Such a design is often incremental in two ways:

(i) Changes in the mode of operation of each module are incremental.

(ii) Modules are added to the organization incrementally. (An organization typically grows from smaller to larger).

Due to this incrementalism, the current shape of an organization can only be understood in terms of its past history. (Organizational design problems may have many local optima; the final design will therefore depend on the starting point and the steps that were followed). The following abstract history of evolution may be hypothesized [Galbraith, 1977]:

1. In the beginning, we have a small organization in which each module operates in a fixed manner, which may be assumed to be close to optimal. Many actions are undertaken without communicating but whenever a module needs some information from other modules, it may obtain it by communicating.

2. As the organization grows, it ceases being optimal and some modules start changing their mode of operation so as to improve performance. As their operation changes, they need -and do - inform those modules who should be concerned about such changes.

3. For an even larger organization, it becomes hard for each module to know who is concerned about what. So, higher levels are introduced who do not know details about the lower level modules, but have a global picture and know who should be concerned about what. They act as coordinators by setting up the necessary

information flows. Also, in case that the interests of the lower level modules divert themselves from the interests of the organization, the higher level designs a reward scheme so as to bring these interests as much in line as possible.

The above procedure does not need to lead to an optimal design. However, the global problem being very hard, the above incremental procedure has to be accepted. What remains to be done, from a normative perspective, is to ensure that each step of the above procedure is carried out in a rational way, as best as possible. The above outline may, therefore, serve as a general guide on what kinds of problems should be addressed.

2.3 ON COMMUNICATIONS AND ASSOCIATED PROBLEMS

Many of the aspects of decentralized systems are also present in centralized ones; therefore, they are not the prime subject of a theory of decentralized systems. The main aspect which is unique to decentralized systems is the distribution of information and communications and these should be the focus of theoretical investigations.

Communications are unimportant if they are instantaneous, unconstrained, not penalized and not corrupted by noise: in such a case, an optimal centralized design coincides with an optimal decentralized design. Communications become important only if one of the above assumptions is violated. Formally, communications add just another component to the cost function and some new constraints. Difficulties arise, however, because typically these new costs and constraints are qualitatively different from the costs and constraints associated to the operation of the rest of the system. This may make decentralized versions of otherwise easy problems particularly hard (see Chapter 3).

For these reasons we need some understanding of basic decentralized problems in which communications enter in a simple way. Such simple problems may yield some insight for more complex ones, or may be used as building blocks. From the mathematical point of view, it is important to discover new ways of incorporating communications costs and constraints into decentralized versions of centralized problems, in a sufficiently simple fashion so that analysis does not become impossible. Then, we may proceed to address fundamental questions such as: "who should communicate what, to whom and when, so as to guarantee that a decentralized system performs as desired?"

2.4 LITERATURE SURVEY

There are many disciplines relevant to the analysis or design of decentralized systems: for example, decision theory, game theory, control theory, mathematical programming, organization theory and computer science. The associated bibliography runs in the thousands and a comprehensive survey is impossible. In this Section, we discuss some representative lines of research, focusing on those areas which bear some relation to our work. More specific references may be also found in the main body of Chapters 3,4 and 5.

Bayesian Team Decision Theory and Decentralized Control

Team decision theory, in its original - static - version [Radner, 1962; Marschak and Radner, 1972] deals with the following problem: a set of processors obtain measurements of a stochastic environment; then, each processor makes a decision, according to a decision rule, based on its own set of measurements only, i.e. without communicating. Then a cost is incurred, depending on the decisions of each processor

and the state of the environment. The problem consists of designing the decision rule of each processor, so as to minimize the expected cost. A solution may be easily obtained for certain special cases in which we may restrict to a set of decision rules admitting simple parametrizations: for example, linear quadratic gaussian (LQG) problems [Marschak and Radner, 1972] or linear gaussian problems with an exponential cost criterion [Krainak, Speyer and Marcus, 1982a, 1982b; Speyer, Marcus and Krainak, 1980]. In both of the above cases, optimal decision rules turn out to be affine functions of the measurements of each processor. Very little, however, can be said for the general version of the static team problem. (See Section 3.4 for some results on the complexity of this problem.) An interesting result has been obtained by Arrow and Radner [1979] who have shown that the law of large numbers may be exploited to yield a simple approximate solution for a class of team problems involving a very large number of processors. Also, Witsenhausen [1981] has derived a lower bound for a class of team problems.

The static problem considered thus far admits a multi-stage (or continuous time) extension [Ho and Chu, 1972; Singh, 1981]: the dynamic team or optimal decentralized control problem. Most studies have focused on the LQG case [Ho and Chu, 1972; Chu, 1972; Ho and Chang, 1980; Ho, 1980], since the general case is much harder, even for static problems. The main results are in a sense negative: a solution may be easily found only in the presence of "partially nested information structures", that is when a condition of the following type is satisfied: If the decision of processor A at time t influences the measurements of processor B at time T (T>t), then the information of B at time T contains the information of processor A, at time t. In the absence of such a condition, Witsenhausen [1968] has constructed a simple two-stage

example showing that optimal decision rules may be nonlinear functions of the information. Even if linearity of decision rules is imposed as a constraint, the optimal decision rules may correspond to infinite-dimensional compensators [Barta, 1978]. Certain special information structures are easier to study. For example, the control-sharing information pattern [Sandell and Athans, 1974] and the one-step delay information pattern [Sandell and Athans, 1974; Kurtaran and Sivan, 1974; Bagchi and Basar, 1980; Varaiya and Walrand, 1978; Krainak et.al., 1982c] for which simple solutions can be found.

The difficulties in the dynamic team problem may be understood in different ways. We first have the "second quessing" effect: each processor makes decisions by guessing the decisions of other processors, hence guess their guesses and so on, ad infinitum, so that infinite dimensional compensators are obtained for finite dimensional systems. Second, there is the possibility of using the system being controlled as a channel for communicating information, by judicious choice of the control variables (signalling) [Sandell and Athans, 1974; Ho, Kastner and Wong, 1978]. From a more mathematical viewpoint, the root of the difficulty lies in the fact that the distribution of information and the prohibition of communications corresponds to a special type of constraint on admissible decision rules, which is hard to handle [Ho and Chang, 1980.]

A way out of these difficulties has been taken by restricting the admissible compensators (decision rules) to have a fixed structure [Chong and Athans, 1971; Looze et.al., 1978]. For the infinite horizon problem, we obtain a nonlinear parametric optimization problem which can be solved numerically [Looze et.al., 1978; Geromel and Bernussou, 1979]; however, it is a non-convex problem that may possess many local minima.

A higher level problem, which has been little studied, consists of designing the information structure, subject to certain constraints [Chu, 1978; Ho and Papadopoulos, 1979; Papavassilopoulos, 1983].

Once the pursuit of optimality is abandoned, we may pose the problem of designing a decentralized compensator which stabilizes a given system. Several, and fairly complete results have been obtained for this problem [Wang and Davison,1973; Saeks, 1979; Anderson and Clements, 1981; Davison and Ozguner, 1983; Sezer and Siljak, 1981], some of them surprising [Anderson and Moore, 1981]. Finally, if the model of the system to be controlled is itself unknown, one may consider the possibility of decentralized adaptive control, on which some preliminary results have been reported [Ioannou and Kokotovic, 1983].

More references in this general area can be found in the survey papers of Sandell et.al. [1978] and Siljak [1983].

Decentralized Estimation and Hypothesis Testing

Suppose that a set of sensors obtains measurements (as in the team problem) and each one of them tries to estimate some random vector or perform a hypothesis test. Suppose also that the cost criterion couples the estimates (or decisions) of different sensors by penalizing, for example, positive correlation of their errors. The problem of designing the estimators (decision rules) of each sensor is then a special case of the team problem; the main difference is that, now, the estimates (decisions) of one sensor do not affect the measurements of other sensors, thus avoiding the possibility of signalling. Barta [1978] has shown that the best estimates for linear Gaussian dynamical systems can be obtained by finite-dimensional filters (resembling to the Kalman filter), although of a much larger dimension.

A two-sensor decentralized hypothesis testing problem has been studied by Tenney and Sandell [1981] under the assumption that the measurements of the two sensors are statistically independent (conditioned on either hypothesis). It was shown that, similarly to the centralized case, likelihood ratios provide sufficient statistics; the optimal thresholds for the two sensors are coupled through nonlinear algebraic equations. Most interestingly, it was shown that asymmetric thresholds may be optimal for perfectly symmetric problems, reflecting hedging behavior. Qualitatively similar results have been also obtained for the decentralized quickest detection problem [Teneketzis, 1982], the decentralized sequential hypothesis testing (Wald) problem [Teneketzis, 1983], as well as problems involving communication of zero-one messages from certain sensors to others [Ekchian and Tenney, 1982]. However, almost all available results depend heavily on the conditional independence assumption. Section 3.3 shows that if this assumption is removed, the problem becomes very hard.

The above described research allows either no communications at all, or the communication of a few zero-one messages. Consequently, the final estimates are, in general, worse than the estimates that would be obtained if the processors were to exchange their detailed information. At the other extreme, we have schemes in which sufficient communications are allowed, so that optimal centralized estimates are obtained. Several architectures for decentralized Kalman filtering (or smoothing) have been suggested [Speyer, 1979; Hassan et.al., 1978; Chang, 1980; Chong, 1979; Willsky et.al., 1982; Levy et.al., 1983]. These results typically boil down to the following: the centralized optimal Kalman filter estimate is a linear function of the measurements of the different sensors and the proposed schemes are different ways of applying the superposition principle.

Finally, we have some intermediate schemes in which real numbers are being communicated, but without necessarily attaining the centralized optimal performance. The scheme of Borkar and Varaiya [1982] is an example. (These results are significantly extended in Chapter 4.) In the scheme of Sanders et.al. [1974, 1978] each sensor produces estimates of only some of the components of an unknown state vector, corresponding to a particular subsystem. Each sensor takes interactions from other subsystems into account using either noisy observations of these interactions, or messages - possibly corrupted by noise - from other sensors. The loss of optimality is compensated by the fact that simpler filters, requiring fewer computations may be used.

Decentralized (Parallel) Computation

In decentralized computation, several processors do a sequence of computations (in parallel) and exchange partial results until they obtain a desired final result. The main advantage is that parallelism may reduce the time required for obtaining the final result (even though the total number of operations involved may be larger).

Arrow and Hurwicz [1960] have shown that decentralized computation is closely related to decentralized decision making. Their inspiration came from the observation that market mechanisms (which can be viewed as decentralized systems) do, under certain assumptions, minimize a social cost function. Moreover, they have indicated the importance of the compatibility of the distribution of computation with the distribution of information. The analogy between optimization algorithms and models of rational decision making has been carried further, especially in the context of

resource allocation in divisionalized organizations. [Moore, 1979; Burton and Obel, 1980]. Section 5.7 of this report also proceeds along these lines.

There has been a significant volume of research on decentralized algorithms for several classical problems, in which each processor only needs to know a subset of the input data. Such algorithms may be broadly classified into synchronous and asynchronous ones.

Synchronous algorithms may be viewed as alternative implementations of centralized (serial) algorithms. In general, however, it is not true that a decentralized implementation of a good serial algorithm is also a good decentralized algorithm. This leads to new and interesting theoretical questions concerning mainly the trade-offs between the number of processors used, the total computation time required and the number of messages that have to be exchanged. Some representative decentralized synchronous algorithms have been developed for traditional numerical analysis problems [Miranker, 1971], for optimal routing in data communication networks [Gallager, 1977], as well as for many combinatorial problems [Borodin and Hopcroft, 1982; Gallager, 1983; Ullman, 1984]. Trade-offs have been extensively studied by computer scientists, because they have consequences on the practical feasibility of solving certain types of problems by VLSI multiprocessors [Yao, 1981].

From a more theoretical perspective, the amount of communications required to solve a discrete (combinatorial) problem by a decentralized algorithm is another complexity measure, similar to the "time" and "space" complexity measures for serial computation. Certain general results have been obtained by Papadimitrou and Sipser [1982]. (See also [Aho, Ullman, Yannakakis, 1983]). We will see, however, in Section 3.5 that it is very hard to determine the minimum number of communications required for solving a given problem.

In many applications, asynchronous decentralized algorithms are desirable, because of weaker requirements on the timing of computations and communications [Kung, 1976]. Such algorithms raise theoretical questions as to how much asynchronism may be tolerated, while maintaining correctness (convergence) of the algorithm. Bertsekas [1982,1983] has obtained general results for the successive approximations algorithm for dynamic programming and the computation of fixed points. Earlier results may be found in [Baudet, 1978; Chazan and Miranker, 1969]. Chapter 5 contains several results on asynchronous decentralized optimization algorithms.

Hierarchical Decision Making and Control

Mesarovic, Mako and Takahara [1970] have suggested a general and formal approach to hierarchical decision making. The environment is assumed to be an interconnected system and a processor is assigned to each subsystem. Each such processor solves a relatively small optimization problem derived from a global optimization problem; a higher level processor (coordinator) affects, through some parameter, the structure of the lower level problems. The main task of the coordinator is to find a value for the parameter so that the decisions evaluated by the lower level processors are optimal for the global problem. The original theory was very abstract [Varaiya, 1972] but it was followed by much research [Mahmoud, 1977] with particular emphasis on infinite dimensional (e.g. optimal control) problems. It seems that this theory is mostly applicable to the steady-state control of dynamic systems. For stochastic and dynamic optimization problems, the hierarchical decision making schemes that have been proposed are, in general, not optimal because they either incorporate some sort of open-loop feedback or because they restrict the set of admissible control

laws in an ad-hoc fashion [Chong and Athans, 1976; Forestier and Varaiya, 1978]. This is not necessarily undesirable, because real-world decentralized systems are not supposed to achieve the centralized optimum, but only to approximate it. However, relatively little progress has been made in obtaining systematically approximately optimal decision rules which are easier to find than the truly optimal ones.

Spatial and Time-Scale Separation

Many large scale systems consist of subsystems which are weakly coupled or which evolve in different time scales. Such a structure may be exploited for designing decentralized controllers based on the theory of singular or non-singular perturbations.

A non-singularly perturbed system is one composed of weakly coupled subsystems (spatial separation). An approximate description can be obtained by neglecting the interactions, during the first stage of the analysis [Simon and Ando, 1961; Aoki, 1968]. Singular perturbations correspond to time scale separation, are related to a relatively rich class of phenomena and need much more sophisticated mathematics to be analyzed [Papanicolaou, Stroock and Varadhan, 1977]. Many results have been obtained with applications in filtering [Haddad, 1976] and control [Teneketzis and Sandell, 1977; Kokotovic, O'Malley and Sannuti, 1976] of linear systems, control of Markov chains [Delebecque and Quadrat, 1981; Phillips and Kokotovic, 1981] and multimodelling in large scale system [Khalil and Kokotovic, 1978].

Game Theory and the Design of Incentives

Game theory aims at characterizing rational behavior in the presence of conflict [Von Neumann and Margenstern, 1953; Luce and Raiffa, 1957; Vorobev, 1977; Roth, 1979]. A particular class of games is related to incentive system design. Here the higher level of an organization chooses a reward scheme and the lower levels make decisions trying to maximize their individual reward. The problem consists of choosing a reward scheme so that the lower level decision makers end up minimizing an organizational cost function. In the language of Section 2.1, the incentive design problem is essentially a problem of designing a decentralized system when the available control modules are "selfish" individuals which aim at maximizing their individual rewards, rather than optimizing the performance of the organization. It should be emphasized that the nature of results obtained for this problem depends heavily on the set of admissible reward schemes; the larger this set, the easier it becomes to force the low-level decision makers to behave in a desired way [Groves, 1973; Groves and Loeb, 1979; Jennergren, 1980; Ho, Luh and Muralidharan, 1981].

CHAPTER 3: DISCRETE DECENTRALIZED DECISION PROBLEMS

3.1 INTRODUCTION AND MOTIVATION

In this Chapter we formulate and study certain simple decentralized problems. Our goal is to formulate problems which reflect the inherent difficulties of decentralization; that is, any difficulty in this class of problems is distinct from the difficulty of corresponding centralized problems. This is accomplished by formulating decentralized problems whose centralized counterparts are either trivial or vacuous.

One of our goals is to determine a boundary between "easy" and "hard" decentralized problems. Our results will indicate that the set of "easy" problems is relatively small.

All problems to be studied are imbedded in a discrete framework; the criteria we use for deciding whether a problem is difficult or not come from complexity theory [Garey and Johnson, 1979; Papadimitriou and Steiglitz, 1982]: following the tradition of complexity theory, problems that may be solved by a polynomial algorithm are considered easy; NP-complete, or worse, problems are considered hard.* However, an NP-completeness result should not be thought as a result that closes a subject, but rather as a result which can redirect research efforts to heuristic and approximate algorithms or possibly easier special cases. It should be also stressed here that NP-completeness of a discrete problem can be an indication that the corresponding continuous problem does not admit an analytical solution nor an efficient numerical solution [Papadimitriou and Tsitsiklis, 1984].

The main issue of interest in decentralized systems may be loosely phrased as "who should communicate to whom, what, how often," etc. From a purely logical point of view, the first question that has to be raised is "are there any communications

*One way of viewing NP-complete problems, is to say that they are effectively equivalent to the Traveling Salesman problem, which is well-known to be algorithmically hard.

necessary?" Any further questions deserve to be studied only if we come to the conclusion that communications are indeed necessary.

The subject of Section 3.2 is to characterize the inherent difficulty of the problem of deciding whether any communications are necessary, for a given situation. We adopt the following approach: a decentralized system exists in order to accomplish a certain goal which is externally specified and well-known. A set of processors obtain (possibly conflicting) observations on the state of the environment. Each processor has to make a decision, based on its own observation. However, for each state of the environment, only certain decisions accomplish the desired goal. The question "are there any communications necessary?" may be then reformulated as "can the goal be accomplished, with certainty, without any communications?" We show that this problem is, in general, a hard one.

We then impose some more structure on the problem, by assuming that the observations of different processors are related in a particular way. The main issue that we address is "how much structure is required so that the problem is an easy one?" and we determine the boundary between easy and hard problems.

In Section 3.3 we study a particular (more structured) decentralized problem - the problem of decentralized hypothesis testing - on which there has been some interest recently, and characterize its difficulty.

In Section 3.4 we formulate a few problems which are related to the basic problem of Section 3.2 and discuss their complexity.

Suppose that it has been found that communications are necessary. The next question of interest is "what is the least amount of communications needed?" This

problem (Section 3.5) is essentially the problem of designing an optimal communications protocol; it is again a hard one and we discuss some related issues.

In Section 3.6 we present our conclusions and discuss the conceptual significance of our results. These conclusions may be summarized by saying that:

a) Even the simplest problems of decentralized decision making are hard.

b) Allowing some redundancy in communications, may greatly facilitate the (off-line) problem of designing a decentralized system.

c) Practical communications protocols should not be expected to be optimal, as far as minimization of the amount of communications is concerned.

The results of this Chapter appear in [Papadimitriou and Tsitsiklis, 1982; Tsitsiklis and Athans, 1985].

3.2 A PROBLEM OF SILENT COORDINATION

In this Section we formulate and study the problem whether a set of processors with different information may accomplish a given goal -with certainty- without any communications.

Let $\{1,\ldots,M\}$ be a set of processors. Each processor, say processor i, obtains an observation y_i which comes from a finite set Y_i of possible observations. Then, processor i makes a decision u_i which belongs to a finite set U_i of possible decisions, according to a rule

$$u_i = \gamma_i(y_i) \tag{3.2.1}$$

where γ_i is some function from Y_i into U_i. The M-tuple $(y_1,\ldots,y_M)$ is the total information available; so it may be viewed as the "state of the environment."

For each state of the environment, we assume that only certain M-tuples $(u_1,\ldots,u_M)$ of decisions accomplish a given, externally specified, goal. More precisely, for each $(y_1,\ldots,y_M)\in Y_1 \times\ldots\times Y_M$, we are given a set $S(y_1,\ldots,y_M)\subset U_1 \times\ldots\times U_M$ of <u>satisficing</u> decisions. (So, S may be viewed as a function from $Y_1 \times Y_2 \times\ldots\times Y_M$ into $2^{U_1 \times\ldots\times U_M}$).

The problem to be studied, which we call the "distributed satisficing problem" (after the term introduced by H. Simon [1980]) may be described formally as follows:

<u>Distributed Satisficing (DS)</u>: Given finite sets $Y_1,\ldots,Y_M$, $U_1,\ldots,U_M$ and a function $S: Y_1 \times\ldots\times Y_M \to 2^{U_1\times\ldots\times U_M}$, are there functions $\gamma_i: Y_i \to U_i$, $i=1,2,\ldots,M$, such that

$$(\gamma_1(y_1),\ldots,\gamma_M(y_M))\in S(y_1,\ldots,y_M),\quad \forall(y_1,\ldots,y_M)\in Y_1\times\ldots\times Y_M? \qquad (3.2.2)$$

<u>Remarks</u>:

1. We are assuming that the function S is "easily computable"; for example, it may be given in the form of a table.

2. The centralized counterpart of DS would be to allow the decision u_i of each agent depend on the entire set $(y_1,\ldots,y_M)$ of observations; so, γ_i would be a function from $Y_1 \times\ldots\times Y_M$ into U_i. (This corresponds to a situation in which all processors share all information). Clearly, then, there exist satisfactory (satisficing) functions $\gamma_i: Y_1 \times\ldots\times Y_M \to U_i$, if and only if $S(y_1,\ldots,y_M)\neq\phi$, $\forall(y_1,\ldots,y_M)\in Y_1 \times\ldots\times Y_M$. Since S is an "easily computable" set as a function of its arguments, we can see that the centralized counterpart of DS is a trivial problem. So, any difficulty inherent in DS is only caused by the fact that information is decentralized.

3. A "solution" for the problem DS cannot be a closed-form formula which gives an answer 0 (no) or 1 (yes). Rather, it has to be an algorithm, a sequence of instructions, which starts with the data of the problem $(Y_1,\ldots,Y_M, U_1,\ldots,U_M,S)$ and eventually provides the correct answer. Accordingly, the difficulty of the problem DS may be characterized by determining the place held by DS in the complexity hierarchy. For definitions related to computational complexity and the methods typically used, the reader is referred to [Garey and Johnson, 1979; Papadimitriou and Steiglitz, 1982].

4. If, for some i, the set U_i is a singleton, processor i has no choice, regarding its decision and, consequently, the problem is equivalent to a problem in which processor i is absent. Hence, without loss of generality, we only need to study instances of DS in which $|U_i| \geq 2, \forall i$*.

5. We believe that the problem DS captures the essence of coordinated decision making with decentralized information and without communications (silent coordination).

Some initial results on DS are given by the following:

<u>Theorem 3.2.1</u>:

a) The problem DS with two processors (M=2) and restricted to instances for which the cardinality of the decision sets is 2 ($|U_i|=2$, i=1,2) may be solved in polynomial time.

b) The problem DS with two processors (M=2) is NP-complete, even if we restrict to instances for which $|U_1|=2$, $|U_2|=3$.

c) The problem DS with three (or more) processors ($M \geq 3$) is NP-complete, even if we restrict to instances for which $|U_i|=2$, $\forall i$.

<u>Proof</u>: We will only prove here part (c); the first two parts are corollaries of Theorem 3.2.2 which is a stronger result.

---

* For any finite set A, we let $|A|$ denote its cardinality.

Our starting point is the satisfiability problem for propositional calculus with three literals per clause (3SAT) which is a known NP-complete problem [Garey and Johnson, 1979]. We will reduce 3SAT to DS with three processors (M=3) and binary decision sets. Given an instance of 3SAT, let V be the set of literals and C the set of clauses. We construct an instance of DS as follows:

Let $Y_i=\{1,2,\ldots,|V|\}, U_i=\{0,1\}$, $i=1,2,3$. Let $S(k,k,k)=\{(0,0,0),(1,1,1)\}$, $k=1,\ldots,|V|$. Finally, we interpret each clause in C as stating that a certain triple is not in the satisficing set. (For example, the clause* $(x_i \vee \bar{x}_j \vee x_k)$, where $1\leq i<j<k\leq|V|$, translates to the statement that the triple of decisions (0,1,0) does not belong to $S(i,j,k)$.

If there exists a satisfying assignment for the instance of 3SAT, let us define decision rules $\gamma_i(j)=0$, $i=1,2,3$, (respectively $\gamma_i(j)=1$, $i=1,2,3$) if the satisfying assignment sets the j-th literal to zero (respectively 1). It then follows that there exist satisficing decision rules.

Conversely, if there exist satisficing decision rules $(\gamma_1,\gamma_2,\gamma_3)$, we must have $\gamma_1(j) = \gamma_2(j) = \gamma_3(j)$, $\forall j$; we assign this common value of $\gamma_i(j)$ to the j-th literal in V, to obtain an assignment that satisfies the clauses in C. ■

Theorem 3.2.1 states that the problem DS is, in general, a hard combinatorial problem, except for the special case in which there are only two processors and each one has to make a binary decision. It should be noted that the difficulty is not caused by an attempt to optimize with respect to a cost function, because no cost function has been introduced. In game theoretic language, we are faced with a "game of kind", rather than a "game of degree".

---

* Throughout this Chapter an overbar stands for negation. Also, $\wedge$ stands for logical "and" and $\vee$ stands for logical "or".

We will now consider some special cases (which reflect the structure of typical practical problems) and examine their computational complexity, trying to determine the dividing line between easy and hard problems. From now on we restrict our attention to the case in which there are only two processors. Clearly, if a problem with two processors is hard, the corresponding problem with three or more processors cannot be easier.

We have formulated above the problem DS so that all pairs $(y_1,y_2) \in Y_1 x Y_2$ are likely to occur. So, the information of different processors is completely unrelated; their coupling is caused only by the structure of the satisficing sets $S(y_1,y_2)$. In most practical situations, however, information is not completely unstructured: when processor 1 observes y_1, it is often able to make certain inferences about the value of the observation y_2 of the other processor and exclude certain values. We now formalize these ideas:

<u>Definition</u>: An <u>Information Structure</u> I is a subset of $Y_1 x Y_2$. We say that an information structure I has degree (D_1,D_2), (D_1,D_2 are positive integers) if:

(i) For each $y_1 \in Y_1$ there exist at most D_1 distinct elements of Y_2 such that $(y_1,y_2) \in I$.

(ii) For each $y_2 \in Y_2$ there exist at most D_2 distinct elements of Y_1 such that $(y_1,y_2) \in I$.

(iii) D_1,D_2 are the smallest integers satisfying (i), (ii).

We now interpret this definition: The information structure I is the set of pairs (y_1,y_2) of observations that may occur together. If I has degree (D_1,D_2) processor 1 may use its own observation to decide which elements of Y_2 may have been observed by

processor 2. In particular, it may exclude all elements except for (at most) D_1 of them. The situation faced by processor 2 is symmetrical.

If $D_1=1$ and processor 1 observes y_1, there is only one possible value for y_2. So, processor 1 knows the observation of processor 2. (The converse is true when $D_2=1$). We could call this a nested information structure because the information of one processor contains the information of the other.

When $D_1=D_2=1$, each processor knows the observation of the other; so, their information is essentially shared.

Since pairs (y_1,y_2) not in I cannot occur, there is no meaning in requiring the processors to make compatible decisions if (y_1,y_2) were to be observed. This leads to the following version of the problem DS:

<u>DSI</u>: Given finite sets $Y_1,Y_2,U_1,U_2,I \subset Y_1 x Y_2$ and a function $S:I \to 2^{U_1 x U_2}$, are there functions $\gamma_i: Y_i \to U_i$, i=1,2, such that

$$(\gamma_1(y_1),\gamma_2(y_2)) \in S(y_1,y_2), \quad \forall (y_1,y_2) \in I? \tag{3.2.3}$$

Note that any instance of DSI is equivalent to an instance of DS in which $S(y_1,y_2)=U_1 x U_2$, $\forall (y_1,y_2) \notin I$. That is, no compatibility restrictions are placed on the decisions of the two processors, for those (y_1,y_2) that cannot occur.

We now proceed to the main result of this Section:

<u>Theorem 3.2.2</u>:

a) The problem DSI restricted to instances satisfying any of the following:

(i) One or more of $|U_1|$, $|U_2|$, D_1,D_2 is equal to 1,

(ii) $|U_1|=|U_2|= 2$,

(iii) $D_1=D_2=2$,

(iv) $D_1 = |U_2|=2$, (or $D_2=|U_1|=2$),

may be solved in polynomial time.

b) The problem DSI is NP-complete even if we restrict to instances for which

$$|U_1|= D_1=3,\ |U_2|= D_2=2 \ .$$

Proof: In order to study the complexity of the distributed satisficing problem, we shall first point out the close connection between this problem and a family of restricted versions of the satisfiability problem for propositional calculus, which we call RSAT. A formula of RSAT has a set of variables $F=F_1 \cup F_2 \cup F_3$, where

$$F_1 = \{y_{ij}: i=1,\ldots,\ell;\ j=1,2,3\},$$

$$F_2 = \{z_i: i=1,\ldots,m\},\quad F_3 = \{x_i : i=1,\ldots,n\},$$

and ℓ,m,n, are non-negative integers.

The set C of clauses of an instance of RSAT are the following:

(i) One clause for each i between 1 and ℓ, stating that exactly one of the variables y_{i1},y_{i2},y_{i3} is true,

(ii) An arbitrary number of clauses of the form $(\bar{y}_{ij} v x_q)$ or $(\bar{y}_{ij} v\ x_q)$, and

(iii) An arbitrary numbers of clauses of the form $(z_i v x_j)$, $(\bar{z}_i v x_j)$, $(z_i v\ \bar{x}_j)$, $(\bar{z}_i v\ \bar{x}_j)$.

For any fixed i, we will say that the variables y_{i1},y_{i2},y_{i3} belong to the same group. Each variable in F_2 or F_3 will be thought as forming a separate group. A

clause of type (ii) or (iii) is said to <u>connect</u> two groups if the literals of that clause belong to these groups. Finally, we will say that an instance of RSAT has degree (D_1,D_2) if each group of variables in F_1 or F_2 is connected to at most D_1 other groups and each group of variables in F_3 is connected to at most D_2 other groups.

<u>Lemma 3.2.1</u>:

a) RSAT restricted to instances of degree (D_1,D_2) is equivalent to DSI restricted to instances of degree (D_1,D_2), with $|U_1|=3$, $|U_2|=2$.

b) RSAT restricted to instances for which $F_1=\phi$ is equivalent to DSI restricted to instances for which $|U_1|=|U_2|=2$.

<u>Proof</u>: Let $Y_1=\{1,\ldots,\ell+m\}$, $Y_2=\{1,\ldots,n\}$. Think of y_{ij} as stating that, if processor 1 observes i (with $1\leq i\leq \ell$) then it decides j; similarly, if processor 1 observes $\ell+i$ (with $1\leq i\leq m$) it decides 1 (respectively 2) if $z_i=1$ (respectively 0). Finally, if processor 2 observes i ($1\leq i\leq n$), it decides 1 (respectively 2) if $x_i=1$ (respectively 0).

We may then interpret the clauses of RSAT of type (ii) and (iii) as stating that certain pairs of decisions are not in the satisficing set. (For example, the statement $(3,2)\notin S(i,j)$, for some i,j, $1\leq i\leq \ell$, $1\leq j\leq n$, is equivalent to the clause $(\bar{y}_{i3}\vee x_j)$.)

Note that processor 1 may choose between three decisions and processor 2 between two decisions, so that $|U_1|=3$, $|U_2|=2$. (In fact, if processor 1 observes i, with $\ell+1\leq i\leq \ell+m$, it may only decide 1 or 2. But this is the same as if it could also decide 3 but the satisficing sets are such that

$(3,k)\notin S(i,j)$, $\forall k\in\{1,2\}$, $\forall i\in\{\ell+1,\ldots,\ell+m\}$, $\forall j\in\{1,\ldots,n\}$.)

In the case where $F_1=\phi$, processor 1 has only two choices for each observation, and this corresponds to DSI with $|U_1|=|U_2|=2$.

Clearly, each "group" of variables corresponds to a possible observation. If two groups are not connected by a clause, then no constraint is placed on the compatibility of decisions for the corresponding pair of observations, and conversely. But placing no compatibility constraint is equivalent to assuming that this pair of observations may never occur together. This shows that the degree of an instance of RSAT is the same as the degree of the information structure I for the corresponding instance of DSI.

Clearly, the above construction may proceed both ways which shows the equivalence of the two problems. This concludes the proof of Lemma 3.2.1. ■

a) (i) If $U_1=1$ or $U_2=1$, the problem is trivial. If $D_2=1$, a satisficing decision rule exists if and only if

$$\bigcap_{\{y_2:(y_1,y_2)\in I\}} \pi_1(S(y_1,y_2))\neq\phi, \quad \forall\, y_1\in Y_1 ,$$

where $\pi_1(u_1,u_2)\triangleq u_1$. The above condition can be clearly tested in polynomial time.

a) (ii) By Lemma 3.2.1(b), DSI with $|U_1|=|U_2|=2$ is equivalent to RSAT restricted to instances for which $F_1=\phi$; the latter is a special case of 2SAT (the satisfiability problem of propositional calculus with two literals per clause) which may be solved in time which is a linear function of the number of variables and clauses. So, a $O(|Y_1|\cdot|Y_2|)$ solution is possible.

a) (iii) Let $D_1=D_2$. Possibly by renaming, assume that Y_1, Y_2 are disjoint sets. Consider the graph $G=(Y_1 \cup Y_2, I)$. (Here $Y_1 \cup Y_2$ is the set of nodes, I the set of undirected edges). Each node of this graph has degree at most 2. Therefore, the connected components of G are either isolated nodes, chains or cycles.* Each connected component of G defines a subproblem and these subproblems are decoupled. So, without loss of generality, we may assume that G consists of a single connected component. We complete the proof under the assumption that G is cycle. (If G is a chain, the proof is similar but simpler; in fact we may introduce an additional edge and make G is a cycle; this will not make the problem any easier, because a new edge simply allows the presence of more constraints. If G is an isolated node the problem is trivial).

Let us assume that G is a cycle. Then Y_1 and Y_2 must have the same number n of elements. In particular, the elements of Y_1, Y_2 may be numbered so that (see Fig. 3.2.1). $I=\{(i,i): i=1,\ldots,n\} \cup \{(i,i-1): i=2,\ldots,n\} \cup \{(1,n)\}$.

Let us define

$$S'(1,n-1) = \Big\{(u_1,u'_{n-1}) \in U_1 x U_2 : \exists (u_n,u'_n) \in U_1 x U_2 \text{ such that } (u_n,u'_n) \in S(n,n),\ (u_n,u'_{n-1}) \in S(n,n-1),\ (u_1,u'_n) \in S(1,n)\Big\}$$

and note that $S'(1,n-1)$ may be evaluated in $O\left[|U_1|^2 |U_2|^2\right]$ time. We now have:

---

* A graph with n nodes is a <u>chain</u>, if its nodes may be numbered so that $E=\{(i,i+1): i=1,\ldots,n-1\}$; it is a <u>cycle</u> if its nodes may be numbered so that $E=\{(i,i+1): i=1,\ldots,n-1\} \cup \{(n,1)\}$. (Here E stands for the set of edges.)

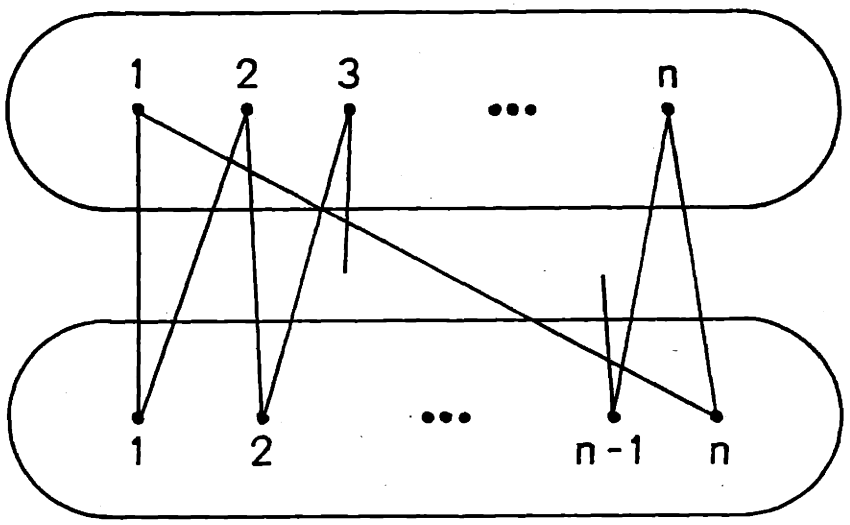

Figure 3.2.1: An Information Structure with $D_1=D_2=2$.

An instance of DSI is a "YES" instance $\Longleftrightarrow$

$$\exists(u_1,u_1',\ldots,u_n,u_n')\Big[(u_i,u_i')\in S(i,i),\ i=1,\ldots,n \text{ and } (u_i,u_{i-1}')\in S(i,i-1),\ i=2,\ldots,n \text{ and } (u_1,u_n')\in S(1,n)\Big] \Longleftrightarrow$$

$$\exists(u_1,u_1',\ldots,u_{n-1},u_{n-1}')\Big[(u_i,u_i')\in S(i,i),\ i=1,\ldots,n-1 \text{ and } (u_i,u_{i-1}')\in S(i,i-1),\ i=2,\ldots,n-1 \text{ and } (u_1,u_{n-1}')\in S'(1,n-1)\Big] .$$

This last expression corresponds to a new instance of DSI in which n has been replaced by n-1. Proceeding in the same way, the problem will be solved after at most n similar stages. This is essentially an algorithm of the dynamic programming type which solves the problem in time $O(|Y_1||U_1|^2|U_2|^2)$.

<u>Remark:</u> In fact an $O(|Y_1||U_1||U_2|(|U_1|+|U_2|))$ solution may be obtained, if at each stage of the dynamic programming algorithm we only eliminate one, rather than two, variables. If G is a chain, an $O(|Y_1||U_1||U_2|)$ algorithm is obtained along the same lines.

a) (iv) We now suppose that $D_1 = |U_2|=2$. Let $Y_1 = \{1,\ldots,m\}$, $Y_2 = \{1,\ldots,n\}$ and assume, without loss of generality, that for each $k\in Y_1$ there exist exactly two distinct elements i, j of Y_2 such that $(k,i)\in I$, $(k,j)\in I$.

Note that we have a "YES" instance of DSI if and only if

$$\exists u_1,\ldots,u_n\in U_2\ \ \exists v_1,\ldots,v_m\in U_1\ \ \forall(k,i)\in I\ [(v_k,u_i)\in S(k,i)] . \qquad (3.2.4)$$

Consider also the statement

$$\exists u_1,\ldots,u_n \in U_2 \ \forall (i,j)\in Y_2 x Y_2 \ \forall k \in Y_1 \Big[[(k,i)\in I \ \Lambda (k,j)\in I \ \Lambda \ i\neq j] \Rightarrow \exists v_k \in U_1 [(v_k,u_i)\in S(k,i) \Lambda \ (v_k,u_j)\in S(k,j)]\Big] \tag{3.2.5}$$

which is equivalent to

$$\exists u_1,\ldots,u_n \in U_2 \ \forall (i,j)\in Y_2 x Y_2 [(u_i,u_j)\in S'(i,j)], \tag{3.2.6}$$

where

$$S'(i,j) = \Big\{(u,u')\in U_2 x U_2 : \forall k \in Y_1 \ \Big[[(k,i)\in I \Lambda \ (k,j)\in I \Lambda \ i\neq j] \Rightarrow \exists v_k \in U_1 [(v_k,u)\in S(k,i) \ \Lambda \ (v_k,u')\in S(k,j)]\Big]\Big\}. \tag{3.2.7}$$

If (3.2.4) holds, then it is easy to see that (3.2.5) holds, as well, with the same assignment of values to u_i, v_k. Conversely, assume that (3.2.5) holds. For each k, there exists only one pair (i,j) such that the condition $[(k,i)\in I]\Lambda[(k,j)\in I]\Lambda[(i \neq j)]$ holds. Accordingly, for each k, the clause $[(v_k,u_i)\in S(k,i) \ \Lambda (v_k,u_j)\in S(k,j)]$ needs to be checked only for one pair (i,j). Therefore, for each k, a value of v_k which makes (3.2.5) true can be chosen regardless of (i,j) and this value makes (3.2.4) true as well.

Therefore, we only need to show that the truth of (3.2.5) can be decided in polynomial time. Note that, for each (i,j), the set S'(i,j) defined by (3.2.7) may be constructed in time $O(|Y_1||U_1|)$. Moreover there are at most $\min\{|Y_2|^2,|Y_1|\}$ pairs to be considered; so the sets S'(i,j) may be constructed in time $O(|Y_1||U_1|\min\{|Y_2|^2,|Y_1|\})$. Once S'(i,j) is constructed, the statement $(u_i,u_j)\in S'(i,j)$ may be expressed as a set of clauses with two literals per clause (the literals are the boolean variables u_i,u_j; this is similar to the proof of part (a) (ii)). Therefore, deciding the truth of (3.2.5) is a special case of the satisfiability problem of propositional calculus with two literals per clause (2SAT), which can be solved in linear time. This concludes the proof of part (a).

<u>Lemma 3.2.2:</u> RSAT is NP-complete, even if $F_2=\phi$.

<u>Proof:</u> The proof consists of reducing to RSAT (with $F_2=\phi$) the problem of satisfiability of propositional formulae with three literals per clause (3SAT). Given such a propositional formula, we shall construct an equivalent RSAT formula F, as follows:

For each variable of the original formula, we have in F an x-variable. For each pair of variables a and b of the original formula, we add to F two <u>triples</u> of y-variables y_{abj} and y'_{abj}, j=1,2,3, and the corresponding "exactly one is true" clause, for each triple. (For example, the clause "exactly one of y_{ab1}, y_{ab2}, y_{ab3} is true," may be written as

$$(y_{ab1} \vee y_{ab2} \vee y_{ab3}) \wedge (\bar{y}_{ab1} \vee \bar{y}_{ab2}) \wedge (\bar{y}_{ab2} \vee \bar{y}_{ab3}) \wedge (\bar{y}_{ab3} \vee \bar{y}_{ab1})) \ .$$

Also, we add to F the x-variable z_{ab} and the following ten clauses:

$$(\bar{y}_{ab1} \vee a), (\bar{y}_{ab1} \vee b), (\bar{y}_{ab2} \vee a), (\bar{y}_{ab2} \vee \bar{b}),$$
$$(\bar{y}'_{ab1} \vee \bar{a}), (\bar{y}'_{ab1} \vee b), (\bar{y}'_{ab2} \vee \bar{a}), (\bar{y}'_{ab2} \vee \bar{b}),$$
$$(\bar{y}_{ab3} \vee z_{ab}), (\bar{y}'_{ab3} \vee \bar{z}_{ab}) \ .$$

It is easy to verify that the above ten clauses force the variables $y_{ab1}, y_{ab2}, y'_{ab1}, y'_{ab2}$ to always take the same values as the expression $(a \wedge b), (a \wedge \bar{b}), (\bar{a} \wedge b), (\bar{a} \wedge \bar{b})$, respectively. Using this observation, we can rewrite any three literal clause of our original formula as a two-literal clause of F. (For example, the clause $(a \vee \bar{b} \vee c)$ is equivalent to $(a \vee \bar{b} \vee c) = \overline{(\bar{a} \wedge b)} \vee c = (\bar{y}'_{ab1} \vee c)$ which is of the type of clause allowed in RSAT). This completes the proof of Lemma 3.2.2. □

The proof of part (b) of Theorem is completed by combining part (a) of Lemma 3.2.1 with the following:

<u>Lemma 3.2.3</u>: RSAT is NP-complete, even if it is restricted to instances for which $D_1=3$, $D_2=2$.

<u>Proof</u>: The idea of the proof is to create multiple copies of each variable so that, instead of connecting a "group" of variables to many other groups, we connect each time a different copy of the same group.

Suppose that we are given an instance (F_1,F_2,F_3,C) of RSAT with $F_2=\phi$. We will construct an equivalent instance (F_1',F_2',F_3',C') of RSAT, for which $D_1=3$, $D_2=2$. We let

$$F_1' = \{y_{ij}^p:\ 1\le i\le \ell;\ j=1,2,3;\ p=1,\ldots,|C|\}\cup F_1,$$

$$F_2' = \{\hat{x}_i^p:\ 1\le i\le n;\ 1\le p\le |C|\},$$

$$F_3' = \{\hat{y}_{ij}^p:\ 1\le i\le \ell;\ j=1,2,3;\ 1\le p\le |C|\}\cup\{x_i^p,\hat{\hat{x}}_i^p:\ 1\le i\le n;\ 1\le p\le |C|\}\cup F_3\ .$$

(Here $y_{ij}^p,\hat{y}_{ij}^p$ (respectively $x_i^p,\hat{x}_i^p,\hat{\hat{x}}_i^p$) are meant to be copies of y_{ij} (respectively x_i).)

The clauses in C' are the following:

(i) For each i, a clause stating that exactly one of y_{i1},y_{i2},y_{i3} is true

(ii) For each i,j,p, clauses stating that:

$$\hat{y}_{ij}^1 = y_{ij},\ \hat{y}_{ij}^p = y_{ij}^{p-1},\quad y_{ij}^p = \hat{y}_{ij}^p$$

$$\hat{x}_i^1 = x_i,\ \hat{x}_i^p = x_i^{p-1},\ x_i^p = \hat{x}_i^p$$

$$\hat{\hat{x}}_i^p = \hat{x}_i^p$$

(Note that an equality a=b may be written as $(a \vee \bar{b}) \wedge (\bar{a} \vee b)$.)

(iii) Finally, for each clause of C (say the k-th clause) introduce an equivalent clause in C' connecting the k-th copies of the variables involved. For example, if the k-th clause is $(\bar{y}_{ij} \vee \bar{x}_q)$ it would become $(\overline{y_{ij}^k} \vee \bar{\tilde{x}}_q^k)$. It is easy to check that this new instance of RSAT has degree (3,2) and is equivalent to the original instance. This concludes the proof of the Theorem.■

The result concerning the case $D_1=1$ or $D_2=1$ is not surprising. It is well-known that nested information structures may be exploited to solve otherwise difficult decentralized problems. But except for the case $D_1=D_2=2$ (which is sort of a boundary) the absence of nestedness makes decentralized problems computationally hard. Our result gives a precise meaning to the statement that non-nested information structures are much more difficult to handle than nested ones.

Theorem 3.2.2 shows that even if D_1, D_2 are held constant, the problem DSI is, in general, NP-complete. There is, however, a special case of DSI, with D_1, D_2 constant, for which an efficient algorithm of the dynamic programming type is possible.

<u>Theorem 3.2.3</u>: Let $Y_1=Y_2=\{1,2,\ldots,n\}$. Let D be a positive integer constant. Consider those instances of DSI for which $(i,j)\in I$ implies either $|i-j|\leq D$ or $|i-j|\geq n-D$. Then DSI may be solved in time which is polynomial in n.

A condition of the type $|i-j|\leq D$, $\forall (i,j)\in I$ is fairly natural in certain applications. For example, suppose that the observations y_1 and y_2 are noisy measurements of an unknown variable x ($y_i=x+w_i$) where the noises w_i are bounded: $|w_i|<D/2$. Similarly, the condition $|i-j|\leq D$ or $|i-j|\geq n-D$, $\forall (i,j)\in I$, arises if the observations y_1, y_2 are noisy measurements of an unknown quantized angle: $y_i = \theta+w_i \pmod{2\pi}$, where the noises w_i are again bounded by D/2.

<u>Proof</u>: This proof is effectively a generalization of the dynamic programming argument in the proof of part (a) (iii) of Theorem 3.2.2. Let us assume that $n \geq 2D$. For any k such that $2D \leq k \leq n$, let

$$\Gamma(k) = \Big\{(u_1,v_1,u_2,v_2,\ldots,u_D,v_D,u_{k-D+1},v_{k-D+1},\ldots,u_k,v_k) \in (U_1 x U_2)^{2D}:$$

$$\exists (u_{D+1},v_{D+1},\ldots,u_{k-D},v_{k-D}) \in (U_1 x U_2)^{k-2D} \text{ such that}$$

$$(u_i,v_j) \in S(i,j),\ \forall (i,j) \in \{1,\ldots,k\}^2 \cap I\Big\} \quad .$$

Note that $\Gamma(k)$ is of size at most $|U_1|^{2D}|U_2|^{2D}$. Now assume that $2D \leq k \leq n-1$ and let

$$\hat{\Gamma}(k+1) = \Big\{(u_1,v_1,u_2,v_2,\ldots,u_D,v_D,u_{k-D+1},v_{k-D+1},\ldots,u_{k+1},v_{k+1}) \in (U_1 x U_2)^{2D+2}:$$

$$\exists (u_{D+1},v_{D+1},\ldots,u_{k-D},v_{k-D}) \in (U_1 x U_2)^{k-2D} \text{ such that}$$

$$(u_i,v_j) \in S(i,j),\ \forall (i,j) \in \{1,\ldots,k+1\}^2 \cap I\Big\}$$

Using the assumption $|i-j| \leq D$, or $|i-j| \geq n-D$, $\forall (i,j) \in I$, we can see that

$$\{1,\ldots,k+1\}^2 \cap I \subset [\{1,\ldots,k\}^2 \cap I] \cup A_{k+1}$$

where

$$A_{k+1} = \{1,\ldots,D,k-D+1,\ldots,k+1\}^2 \cap I.$$

With this observation, $\hat{\Gamma}(k+1)$ may be rewritten as

$$\hat{\Gamma}(k+1) =$$

$$\Big\{(u_1,v_1,u_2,v_2,\ldots,u_D,v_D,u_{k-D+1},v_{k-D+1},\ldots,u_{k+1},v_{k+1}) \in (U_1 x U_2)^{2D+2}:$$

$$(u_1,v_1,u_2,v_2,\ldots,u_D,v_D,u_{k-D+1},v_{k-D+1},\ldots,u_k,v_K) \in \Gamma(k) \text{ and}$$

$$(u_i,v_j) \in S(i,j),\ \forall (i,j) \in A_{k+1}\Big\} \ .$$

Assuming that the set $\Gamma(k)$ has been computed, we may use it to evaluate $\hat{\Gamma}(k+1)$ as follows: for each element of $\Gamma(k)$ (at most $|U_1|^{2D}|U_2|^{2D}$ elements) try each pair $(u_{k+1},v_{k+1})\in U_1 x U_2$ ($|U_1||U_2|$ pairs) and check for each $(i,j)\in A_{k+1}$ (A_{k+1} has at most $4D^2$ elements) whether $(u_i,v_j)\in S(i,j)$ holds. Therefore, given $\Gamma(k)$, we may obtain $\hat{\Gamma}(k+1)$ in time $O(D^2|U_1|^{2D+1}|U_2|^{2D+1})$. Finally, from $\hat{\Gamma}(k+1)$ we may easily obtain $\Gamma(k+1)$, by taking a projection so as to eliminate u_{k-D+1},v_{k-D+1}. This process may be repeated (for no more than n stages) to compute $\Gamma(n)$, in time $O(nD^2|U_1|^{2D+1}|U_2|^{2D+1})$. Then note that we have a YES instance of DSI if and only if $\Gamma(n)\neq\phi$. ■

Remark: The algorithm in the proof of Theorem 3.2.3 does not find a satisficing decision rule; it only determines whether one exists. However, satisficing decision rules may be computed by keeping in the memory some of the intermediate results produced by the algorithm.

3.3 DECENTRALIZED DETECTION

A basic problem in decentralized signal processing, which has attracted a fair amount of attention recently, is the problem of decentralized detection (hypothesis testing) [Tenney and Sandell, 1981; Ekchian, 1982; Ekchian and Tenney, 1982; Kushner and Pacut, 1982; Lauer and Sandell, 1983]. In this section we consider a simple (discrete) version of this problem involving only two processors and two hypotheses.

Two processors S_1 and S_2 receive observations $y_1\in Y_1$, $y_2\in Y_2$, respectively, where Y_i is the set of all possible observations of processor i. (Figure 3.3.1). There are two hypotheses H_o and H_1 on the state of the environment, with prior probabilities

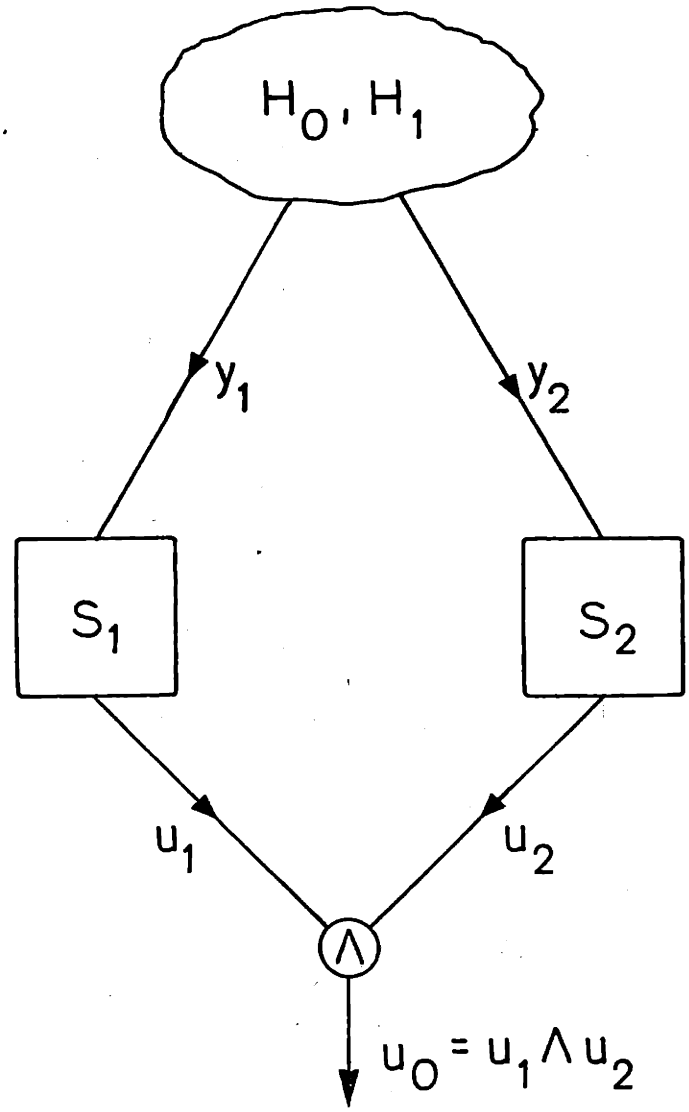

Figure 3.3.1: A Structure for Decentralized Detection.

p_o and p_1, respectively. For each hypothesis H_i, we are also given the joint probability distribution $P(y_1,y_2|H_i)$ of the observations, conditioned on the event that H_i is true. Upon receipt of y_i, processor S_i evaluates a binary message $u_i \in \{0,1\}$ according to a rule $u_i = \gamma_i(y_i)$, where $\gamma_i: Y_i \to \{0,1\}$. Then, u_1 and u_2 are transmitted to a central processor (fusion center) which evaluates $u_o = u_1 u_2$ and declares hypothesis H_o to be true if $u_o=0, H_1$ if $u_o=1$. (So, we essentially have a voting scheme). The problem is to select the functions γ_1, γ_2 so as to minimize the probability of accepting the wrong hypothesis. (More general performance criteria may be also considered).

Most available results assume that

$$P(y_1,y_2|H_i) = P(y_1|H_i)P(y_2|H_i), \quad i=1,2 \tag{3.3.1}$$

which states that the observations of the two processors are <u>independent</u>, when conditioned on either hypothesis.* In particular, it has been shown [Tenney and Sandell, 1981] that if (3.3.1) holds then the optimal decision rules γ_i are given in terms of thresholds for the likelihood ratio $\dfrac{P(H_o|y_i)}{P(H_1|y_i)}$. The optimal thresholds for the two sensors are coupled through a system of equations which gives necessary conditions of optimality. (These equations are just the person-by-person optimality conditions). Few analytical results are available when the conditional independence assumption is removed [Lauer and Sandell, 1983]. The purpose of this section is precisely to explain this status of affairs.

Suppose that (3.3.1) holds and let N_i denote the cardinality of Y_i. Given the results of [Tenney and Sandell [1981]] there are only $N_i + 1$ decision rules γ_i which are candidates for being optimal. We may evaluate the cost associated to each pair of candidate decision rules and select a pair with least cost. This corresponds to a

*Such an assumption is reasonable in problems of detection of a known signal in independent noise, but is typically violated in problems of detection of an unknown signal.

polynomial algorithm and shows that under condition (3.3.1) decentralized detection is an easy problem. Without the conditional independence assumption (3.3.1), however, there is no guarantee that optimal decision rules can be defined in terms of thresholds for the likelihood ratio. Accordingly, a solution by exhaustive enumeration could require the examination of as many as $2^{N_1+N_2}$ pairs of decision rules. One might expect that a substantially faster (i.e. polynomial) algorithm is possible. However, the main result of this section (Theorem 3.3.1 below) states that decentralized detection is NP-complete even if we restrict to instances for which perfect detection (zero probability of error) is possible for the corresponding centralized detection problem.

We now present formally a suitable version of the problem:

Decentralized Detection (DD): We are given finite sets Y_1, Y_2; a rational number K; a rational probability mass function $p: Y_1 x Y_2 \to Q$; a partition $\{A_o, A_1\}$ of $Y_1 x Y_2$. Do there exist $\gamma_i: Y_i \to \{0,1\}$, i=1,2, such that $J(\gamma_1,\gamma_2) \leq K$? Here

$$J(\gamma_1,\gamma_2) = \sum_{(y_1,y_2)\in A_o} p(y_1,y_2)\gamma_1(y_1)\gamma_2(y_2) + \sum_{(y_1,y_2)\in A_1} p(y_1,y_2)[1-\gamma_1(y_1)\gamma_2(y_2)]. \tag{3.3.2}$$

Remarks:

1. In the above definition of DD, think of H_i as being the hypothesis that $(y_1,y_2)\in A_i$. Then it is easy to see that $J(\gamma_1,\gamma_2)$ corresponds to the probability of error associated to the decision rules γ_1,γ_2. Note that if a single processor knew both y_1 and y_2 (centralized information) it could make the correct decision with certainty. Consequently, the above defined problem corresponds to the special case of decentralized detection problems for which perfect centralized detection is possible.

2. If we let K=0, then DD is a special case of problem DS with $|U_1|=|U_2|=2$ and is therefore polynomially solvable.

Theorem 3.3.1: DD is NP-complete

Proof: Consider the following problem of propositional calculus, which we call P:

Problem P: We are given two sets $X = \{x_1,\ldots,x_m\}$, $Z = \{z_1,\ldots,z_n\}$ of boolean variables; a set D of (distinct) clauses of the form $x_i \wedge z_j$ or $\overline{(x_i \wedge z_j)}$ (we assume that for any pair (i,j) at most one of the above clauses is in D); a collection $\{q_{ij}: i\in\{1,\ldots,m\},\ j\in\{1,\ldots,n\}\}$ of non-negative integers and an integer K. Is there a truth assignment for X and Z such that $J\leq K$, where

$$J \triangleq \sum_{\substack{x_i\wedge z_j=0\\(i,j)\in A_1}} q_{ij} + \sum_{\substack{x_i\wedge z_j=1\\(i,j)\in A_o}} q_{ij}\ , \tag{3.3.3}$$

$A_o = \{(i,j)$: the clause $\overline{(x_i\wedge z_j)}$ is in D$\}$,
$A_1 = \{(i,j)$: the clause $(x_i\wedge z_j)$ is in D$\}$.*

Lemma 3.3.1: Problem P is equivalent to DD.

Proof: Think of X, Z as being the sets of observations of processors S_1, S_2, respectively. A truth assignment to X,Z corresponds to a choice as to what binary message to transmit to the fusion center, given each processor's observation. Let H_o (respectively H_1) be the hypothesis that $(i,j)\in A_o$ (respectively A_1). Finally, view q_{ij}, as the (unnormalized) probability that the pair (i,j) of observations is obtained by the two processors. Pairs (i,j) that belong to neither A_o nor A_1 may be viewed as having zero probability and are, therefore, of no concern. Then, it is easy to verify that J as defined by (3.3.3) is precisely the (unnormalized) probability of error. □

*So, J is the sum of the weights q_{ij} of the clauses that are not satisfied.

In order to complete the proof of the Theorem, we need to show that P is NP-complete. This will be accomplished by reducing to P the following (Maximum 2-Satisfiability) problem of propositional calculus which is known to be NP-complete [Garey and Johnson, 1979].

<u>MAX-2-SAT</u>: Given a set U of boolean variables, a collection C of (distinct) clauses over U, such that each clause $c \in C$ has exactly two variables and an integer $K \leq |C|$, is there a truth assignment for U which simultaneously satisfies at least K of the clauses in C? (Without loss of generality, we assume that if a clause is in C, then its negation is not in C).

Suppose that we are given an instance (U,C,K) of MAX-2-SAT. We construct an instance of P as follows: Suppose that $U = \{u_1,\ldots,u_n\}$. Then, let $X = \{x_{i1},x_{i2},x_{i3}: i=1,\ldots,n\}$ and $Z = \{z_{i1},z_{i2},z_{i3}: 1,\ldots,n\}$. For each $i \in \{1,\ldots,n\}$ introduce the set D_i of clauses:

$$(\overline{x_{i1} \wedge z_{i2}}),\quad (x_{i2} \wedge z_{i2}),\quad (\overline{x_{i2} \wedge z_{i1}}),\quad (x_{i3} \wedge z_{i2}),$$

$$(\overline{x_{i2} \wedge z_{i3}}),\quad (x_{i3} \wedge z_{i1}),\quad (x_{i1} \wedge z_{i3}),\quad (x_{i3} \wedge z_{i3}) .$$

To these clauses we assign the weights (L is a large integer to be determined later):

$$q_{i1,i2}=30L,\quad q_{i2,i2}=15L,\quad q_{i2,i1}=4L,\quad q_{i3,i2}=20L,$$

$$q_{i2,i3}=8L,\quad q_{i3,i1}=2L,\quad q_{i1,i3}=25L,\quad q_{i3,i3}=100L .$$

Next, for each clause $(u_i \wedge u_j)$, $(\bar{u}_i \wedge u_j)$, $(\bar{u}_i \wedge \bar{u}_j)$, $(u_i \vee u_j)$, $(\bar{u}_i \vee u_j)$, $(\bar{u}_i \vee \bar{u}_j)$ in C (with $i<j$), introduce clauses $(x_{i1} \wedge z_{j1})$, $(x_{i2} \wedge z_{j1})$, $(x_{i2} \wedge z_{j2})$, $(\overline{x_{i2} \wedge z_{j2}})$, $(\overline{x_{i1} \wedge z_{j2}})$, $(\overline{x_{i1} \wedge z_{j1}})$, respectively. Denote this last set of clauses by D_o, and assign to each

one unit weight. We now let $D = \bigcup_{\ell=0}^{n} D_\ell$ and observe that $X,Z,D,\{q_{ij}\}$, K define an instance of P.

Note that for any assignment for X,Z, the corresponding cost (equation (3.3.3)) may be decomposed as

$$J = J_o + \sum_{\ell=1}^{n} J_\ell \ . \tag{3.3.4}$$

where J_ℓ, $\ell\epsilon\{0,1,\ldots,n\}$ is the sum of the weights q_{ij} of the clauses in D_ℓ which are not satisfied .

<u>Lemma 3.3.2</u>: For any $i\epsilon\{1,\ldots,n\}$, we have $J_i=35L$ if and only if either of the following is true:

(i) $x_{i1}=z_{i1}=x_{i3}=z_{i3}=1, \quad x_{i2}=z_{i2}=0,$

(ii) $x_{i2}=z_{i2}=x_{i3}=z_{i3}=1, \quad x_{i1}=z_{i1}=0$.

For any assignment to $\{x_{ij},z_{ij}: i=1,2,3\}$ other than the two assignments above $J_i \geq 37L$.

<u>Proof</u>: By direct evaluation of the costs of each possible assignment. (See Table 3.3.1). Note that in the calculations in Table 3.3.1, we only consider assignments such that $x_{i3}=z_{i3}=1$. This is because the clause $(x_{i3}\wedge z_{i3})$ carries large enough weight $(q_{i3,i3}=100L)$, so that the possibility of letting $(x_{i3}\wedge z_{i3})=0$ may be immediately discarded. □

TABLE 3.3.1

The cost of each assignment to $\{x_{i1}, x_{i2}, x_{i3}, z_{i1}, z_{i2}, z_{i3}\}$. (The factor L is omitted.)

Clause				$\overline{x_{i1} \wedge z_{i2}}$	$x_{i2} \wedge z_{i2}$	$\overline{x_{i2} \wedge z_{i1}}$	$x_{i3} \wedge z_{i2}$	$\overline{x_{i2} \wedge z_{i3}}$	$x_{i3} \wedge z_{i1}$	$x_{i1} \wedge z_{i3}$	
Weight				30	15	4	20	8	2	25	TOTAL
x_{i1}	z_{i2}	x_{i2}	z_{i1}								
0	0	0	0		15		20		2	25	62
0	0	0	1		15		20			25	60
0	0	1	0		15		20	8	2	25	70
0	0	1	1		15	4	20	8		25	72
0	1	0	0		15				2	25	42
0	1	0	1		15					25	40
0	1	1	0					8	2	25	35
0	1	1	1			4		8		25	37
1	0	0	0		15		20		2		37
1	0	0	1		15		20				35
1	0	1	0		15		20	8	2		45
1	0	1	1		15	4	20	8			47
1	1	0	0	30	15				2		47
1	1	0	1	30	15						45
1	1	1	0	30				8	2		40
1	1	1	1	30		4		8			42

In view of Lemma 3.3.2, the clauses in D_i and their associated weights have the following interpretation: the variable x_{i1} may be freely assigned, but the remaining variables must be assigned so that $x_{i2}=z_{i2}=\overline{z_{i1}}=\overline{x_{i1}}$. For this reason the clauses in D_o are effectively the same as the original set C of clauses.

Lemma 3.3.3: Let L be large enough so that $|C|<L$. Then, there exists a truth assignment for U for which at least K clauses in C are satisfied, if and only if there exists a truth assignment for X,Z such that the resulting cost J is less or equal than $35nL + |C|-K$.

Proof: (i) Given an assignment for U, with at least K clauses satisfied, assign the variables in X,Z as follows:

$$x_{i1}=z_{i1}=u_1, \quad x_{i2}=z_{i2}=\overline{u_1}, \quad x_{i3}=z_{i3}=1 .$$

Using Lemma 3.3.2 and the identity (3.3.4), the resulting cost in 35nL (i.e. 35L from each collection D_i, $i=1,\ldots,n$) plus the number of clauses in D_o which are not satisfied (since these carry unit weight). The latter number is identical to the number of clauses in C which are not satisfied, which is less or equal than $|C|-K$.

(ii) Conversely, given an assignment for X,Z such that $J\leq 35nL + |C|-K$, suppose that for some $i\in\{1,\ldots,n\}$, $J_i \geq 37L$. Using Lemma 3.3.2 and the inequality $|C|<L$, we obtain

$$J \geq \sum_{i=1}^{n} J_i \geq 35nL + 2L > 35nL + |C|-K$$

which is a contradiction and shows that $J_i = 35L$, $\forall i$. Consequently, $\{x_{i1},x_{i2},z_{i1},z_{i2}\}$ have been assigned values in one of the two ways suggested by Lemma 3.3.2. We now

assign truth values for U, by setting $u_i = x_{i1}$. Then J_o is the number of clauses in C which are not satisfied. Moreover, since $J_i = 35L$, $i \in \{1,\ldots,n\}$, it follows that $J_o \leq |C|-K$, which implies that at least K clauses in C are satisfied. This completes the proof of Lemma 3.3.3.□

It is easy to see that the above reduction of MAX-2-SAT to P is polynomial. Therefore, P is NP-complete and so is DD, thus completing the proof of the Theorem.■

It should be pointed out that Theorem 3.3.1 remains valid if the problem is modified so that the fusion center uses some other rule for combining the messages it receives (e.g. $u_o = u_1(1-u_2)$), or if the combining rule is left free and the fusion center is supposed to find and use an optimal such rule.

Let us now interpret Theorem 3.3.1. Although it is, in a sense, a negative result, it can be useful in suggesting meaningful directions for future research: instead of looking for efficient exact algorithms, the focus should be on approximate ones. (In fact, it is an interesting research question whether polynomial approximate algorithms for DD exist). Theorem 3.3.1 also shows that any necessary conditions to be developed for problem DD will be deficient in one of two respects:

a) Either there will be a very large number of decision rules satisfying these conditions,

b) Or, it will be hard to find decision rules satisfying these conditions.

Another consequence of Theorem 3.3.1 is that optimal decision rules are not given, in general, in terms of thresholds for the likelihood ratio, because in that case an efficient algorithm could be obtained. Of course, this fact can be also verified directly by constructing appropriate examples. When the condition (3.3.1) holds and decision rules are given in terms of thresholds, the decision rule of a processor can

be viewed as a tentative local decision, submitted to the fusion center. In general, however, optimal decision rules are not threshold rules and this interpretation is no more valid. Rather DD should be viewed as a problem of optimal quantization of the observation of each processor. In that respect, Theorem 3.3.1 is reminiscent of the result of Garey, Johnson and Witsenhausen [1982], namely that the general problem of minimum distortion quantization is NP-complete.

3.4 RELATED PROBLEMS

The best known static decentralized problem is the team decision problem [Marschak and Radner, 1972] which admits an easy and elegant solution under linear quadratic assumptions. Its discrete version is the following:

Team Decision Problem (TDP): Given finite sets Y_1, Y_2, U_1, U_2, a rational probability mass function $p: Y_1 x Y_2 \to Q$ and an integer cost function $c: Y_1 x Y_2 x U_1 x U_2 \to N$, find decision rules $\gamma_i: Y_i \to U_i$, $i=1,2$ which minimize the expected cost

$$J(\gamma_1,\gamma_2) = \sum_{y_1 \in Y_1} \sum_{y_2 \in Y_2} c(y_1,y_2,\gamma_1(y_1),\gamma_2(y_2))p(y_1,y_2)$$

Another problem related to DS is the following:

Maximize Probability of Satisficing (MPS): Given finite sets Y_1, Y_2, U_1, U_2 a probability mass function $p: Y_1 x Y_2 \to Q$ and a function

$S: Y_1 x Y_2 \to 2^{U_1 x U_2}$, find decision rules $\gamma_i: Y_i \to U_i$, $i=1,2$, which maximize

$$J(\gamma_1,\gamma_2) = P\left[(\gamma_1(y_1),\gamma_2(y_2)) \in S(y_1,y_2)\right]$$

(which is the probability of making a satisfactory decision).

Given an instance of TDP, let

$$S(y_1,y_2) = \{(u_1,u_2): c(y_1,y_2,u_1,u_2)=0\}.$$

If we solve TDP, we also effectively answer the question whether $J(\gamma_1,\gamma_2)=0$. But this is equivalent to solving the instance of DS associated to the above definition of $S(y_1,y_2)$. Therefore, TDP cannot be easier than DS. The same argument is also valid for MPS. It then follows from Theorem 3.2.1 that TDP and MPS are NP-hard (that is, NP-complete or worse) even if we restrict to instances for which $|U_1|=2, |U_2|=3$. However, even more is true: it suffices to notice that the problem DD of the previous section is a special case of both TDP and MPS, with $|U_1|=|U_2|=2$. Using Theorem 3.3.1, we obtain:

<u>Corollary 3.4.1</u>: TDP and MPS are NP-hard even if we restrict to instances for which $|U_1|=|U_2|=2$. This is true even if the cost function c associated to TDP is restricted to take only the values 0 and 1.

We could also define dynamic versions of DS or of the team problem, in a straightforward way [Tenney, 1983]. Since dynamic problems cannot be easier than static ones, they are automatically NP-hard.

Corollary 3.4.1 states that unlike the linear quadratic case, the team decision problem is, in general, a hard combinatorial problem. The reason for this difference is the following: in the linear quadratic problem, Radner's theorem [Radner, 1962] guarantees that, as a consequence of the convex structure of the problem, a person-by-person optimal decision rule is also team optimal. This is no longer true for nonconvex (for example, discrete) team problems. While it may be argued that finding person-by-person optimal decision rules is relatively easy, there is no simple criterion

for deciding whether a given person-by-person optimal decision rule is also team optimal and this is really the source of the difficulty. Let us also stress here that difficulties arising from the possibility of multiple person-by-person optima are equally relevant to team decision problems formulated on continuous decision and probability spaces, as is commonly done. In other words, the difficulties do not arise because we discretize an otherwise easy problem, but they are of a more fundamental nature [Papadimitriou and Tsitsiklis, 1984].

3.5 ON DESIGNING COMMUNICATIONS PROTOCOLS

Suppose that we are given an instance of the distributed satisficing problem (DS) and that it was concluded that unless the processors communicate, satisficing cannot be guaranteed for all possible observations. Assuming that communications are allowed (but are costly), we have to consider the problem of designing a communications protocol: what should each processor communicate to the other, and at what order? Moreover, since communications are costly, we are interested in a protocol which minimizes the total number of binary messages (bits) that have to be communicated in the worst case.

Before proceeding, we must make more precise the notion of a communication protocol and of the number of bits that guarantee satisficing.

Given an instance $\mathcal{D}=(Y_1,Y_2,U_1,U_2,I,S)$ of the problem DSI we will say that:

There is a protocol which guarantees satisficing with 0 bits of communications, if $\mathcal{D}$ is a YES instance of the problem DSI. (That is, if there exist satisficing decision rules, involving no communications.)

We then proceed inductively:

There is a protocol which guarantees satisficing with K bits of communications ($K \in N$), if for some $i \in \{i,2\}$ (say, $i=1$) there is a function $m: Y_1 \to \{0,1\}$, such that for each of the instances $\mathcal{D}'=(Y_1 \cap m^{-1}(0), Y_2, U_1, U_2, I \cap [(Y_1 \cap m^{-1}(0)) x Y_2], S)$ and $\mathcal{D}''=(Y_1 \cap m^{-1}(1), Y_2, U_1, U_2, I \cap [(Y_1 \cap m^{-1}(1)) x Y_2], S)$ there is a protocol which guarantees satisficing with not more than K-1 bits of communications. (Here $m^{-1}(i)=\{y_1 \in Y_1 : m(y_1)=i\}$).

The envisaged sequence of events behind this definition is the following: Each processor observes its measurement $y_i \in Y_i$, $i=1,2$. Then, one of the processors, say processor 1, transmits a message $i=m(y_1)$, with a single bit to the other processor. From that point on, it has become common knowledge that $y_1 \in Y_1 \cap m^{-1}(i)$; therefore, the remaining elements of Y_1 may be ignored.

We can now state formally the problem of interest:

<u>MBS (Minimum bits to satisfice)</u>: Given an instance $\mathcal{D}$ of DSI and $K \in N$, is there a a protocol which guarantees satisficing with not more than K bits of communications?

By definition, MBS with K=0 is identical to the problem DSI. Moreover, MBS with K arbitrary cannot be easier than MBS with K=0 (which is a special case). Therefore, MBS is, in general NP-hard. Differently said, problems involving communications are at least as hard as problems involving no communications.

We have seen in Section 3.2 that when $|U_1|=|U_2|=2$, DSI may be solved in polynomial time. Therefore, MBS with K=0, $|U_1|=2$, $|U_2|=2$ is polynomially solvable. However, for arbitrary K, this is no longer true:

<u>Theorem 3.5.1</u>: MBS is NP-complete, even if $|U_1|=|U_2|=\{0,1\}$ and even if we restrict to instances for which, for any $(y_1,y_2) \in I$, either $S(y_1,y_2)=\{(0,0)\}$ or $S(y_1,y_2)=\{(1,1)\}$.

The above theorem proves a conjecture of A. Yao [Yao, 1979]. The proof was mainly constructed by C. Papadimitriou and may be found in [Papadimitriou and Tsitsiklis, 1982].

We should point out that the special case referred to in Theorem 3.5.1 concerns the problem of distributed function evaluation: we are given a Boolean function $f: Y_1 x Y_2 \rightarrow \{0,1\}$ and we require that both processors eventually determine the value of the function (given the observation - input (y_1,y_2)), by exchanging a minimum number of bits. In our formalism, $S(y_1,y_2)=\{(0,0)\}$ if $f(y_1,y_2)=0$ and $S(y_1,y_2)=\{(1,1)\}$ if $f(y_1,y_2)=1$.

In Section 3.2 we had investigated the complexity of DSI by restricting to instances for which the set I had constant degree (D_1,D_2). This may be done, in principle, for MBS, as well, but no results are available, except for the simple case in which $D_1=D_2=2$.

In fact, when $D_1=D_2=2$ each processor may transmit its information to the other processor by communicating a single binary message and, for this reason, we have:

<u>Proposition 3.5.1</u>: MBS restricted to instances for which $D_1=D_2=2$ may be solved in polynomial time. Moreover, an optimal protocol requires transmission of at most two binary messages, one from each processor.

When (D_1,D_2) is larger than (2,2), there is not much we can say about optimal protocols. However, it is easy to verify that there exist fairly simple non-optimal protocols (which may be calculated in polynomial time) which involve relatively small amounts of communication. This is because:

<u>Proposition 3.5.2</u>: Suppose that I has degree (D_1,D_2) and that $S(y_1,y_2)=\phi$, $\forall(y_1,y_2)\in I$. Then information may be centralized (and therefore satisficing is guaranteed) by

means of a protocol requiring communication of at most $[\log_2(D_1D_2)]$ binary messages by each processor. Moreover, such a protocol may be constructed in time $O((|Y_1|\cdot|Y_2|)(|Y_1|+|Y_2|))$. (Here $[x], x\in R$, stands for the smallest integer larger than x.)

Remark: It might be tempting to guess that processor 1 (respectively 2) needs to communicate only $[\log_2 D_2]$ (respectively $[\log_2 D_1]$) bits, but this is not true, as can be seen from fairly simple examples.*

Proof: Consider the (undirected) graph $G=(Y_1\cup Y_2, I)$. Here $Y_1\cup Y_2$ is the set of nodes (we assume that, by possibly renaming elements, $Y_1\cap Y_2=\phi$) and I is the set of edges. Note that G, is a bipartite graph. Each $y\in Y_i$ is connected to at most D_i other nodes.

We will show that we may colour the nodes of G so that, if $(y_1,y_2)\in I$ and $(y_1,y_2')\in I$, then y_2 and y_2' have different colours; similarly, if $(y_1,y_2)\in I$, $(y_1',y_2)\in I$, then y_1,y_1' will have different colours. Moreover, at most D_1D_2 colours will be used. Then, each processor i, may transmit to the other the colour of his own observation ($[\log_2(D_1D_2)]$ binary messages) and the other processor will be able to infer the exact value of the observation of the first one.

The above mentioned colouring may be accomplished as follows: We first colour the elements of Y_1. Suppose that the first k elements $y_1^{(1)},\ldots,y_1^{(k)}$ of Y_1 have been coloured. Consider the k+1 element $y_1^{(k+1)}$. If for some $i\leq k$, there exists $y_2\in Y_2$

*Nevertheless, there exist protocols for centralizing information which are more efficient than the one we construct here (A. El Gamal, private communication). See also [Witsenhausen, 1976] for a related problem.

such that $(y_1^{(k+1)}, y_2) \in I$ and $(y_1^{(i)}, y_2) \in I$, then $y_1^{(k+1)}$ must be colored differently from $y_1^{(i)}$. Now, $y_1^{(k+1)}$ is connected to at most D_1 elements of Y_2 and each such element of Y_2 is connected to at most D_2-1 elements of Y_1, other than $y_1^{(k+1)}$. This means that at most $D_1(D_2-1) \leq D_1D_2-1$ colours are prohibited for $y_1^{(k+1)}$. Therefore, if D_1D_2 colours are available, $y_1^{(k+1)}$ may be coloured in the desired way, and so on. Nodes in Y_2 may be also coloured in the same way.

With this algorithm, for each $y_1^{(k+1)}$ to be coloured we must examine $|Y_1| \cdot |Y_2|$ elements $(y_1^{(i)}, y_2)$ to check whether $(y_1^{(k+1)}, y_2) \in I$ and $(y_1^{(i)}, y_2) \in I$. So, the set Y_1 is coloured in time $|Y_1|^2 |Y_2|$ and the construction of the desired protocol takes time $O(|Y_1| \cdot |Y_2| (|Y_1| + |Y_2|))$. ■

3.6 CONCLUSIONS

We summarize here the main conclusions from the investigations of this Chapter.

Even if a set of processors have complete knowledge of the structure of a decentralized decision making problem and the desired goal; even if the corresponding centralized problem is trivial; even if all relevant sets are finite, a satisficing decision rule that involves no on-line communications may be very hard to find, the corresponding problem being, in general, NP-complete. There are many objections to the idea that NP-completeness is an unequivocal measure of the difficulty of a problem, because it

is based on a worst case analysis, whereas the average performance of an algorithm might be a more adequate measure; moreover NP-hard optimization problems may have very simple approximate algorithms. However, NP-complete problems are often characterized by the property that any known algorithm is very close to systematic exhaustive search; they do not possess any structure to be exploited. Furthermore, NP-completeness of a discrete problem is an indication that the corresponding continuous problem is very likely to be hard as well.

Concerning the problem DS, and its variations, we may reach the following specific conclusions: No simple algorithm could solve DS. Given that communications would be certainly required for those instances of DS that possess no satisficing decision rules, it would not be a great loss if we allowed the processors to communicate even for some instances of DS for which this would not be necessary. Even if these extra communications-being redundant - do not lead to better decisions, they may greatly facilitate the decision process and -from a practical point of view - remove some load from the computing machines employed.

Concerning the problem of distributed detection, we have shown that it becomes hard, once a simplifying assumption of conditional independence is removed. This explains why no substantial progress on this problem had followed the work of Tenney and Sandell [1982].

From a more general perspective, we are in a position to say that the basic (and the simplest) problems of decentralized decision making are hard, in a precise mathematical sense. Moreover, their difficulty does not only arise when one is interested in optimality. Difficulties persist even if optimality is replaced by satisficing. As a consequence, further research should focus on special cases and easily solvable problems as well as an approximate versions of the original problems.

In cases where communications are necessary (but costly) there arises naturally the problem of designing a protocol of communications. Unfortunately, if this problem is approached with the intention to minimize the amount of communications that will guarantee the accomplishment of a given goal, we are again led to intractable combinatorial problems. Therefore, practical communications protocols can only be designed on a "good" heuristic or ad-hoc basis, and they should not be expected to be optimal; approximate optimality is probably a more meaningful goal. Again, allowing some redundancy in on-line communications may lead to substantial savings in off-line computations.

CHAPTER 4: CONVERGENCE AND ASYMPTOTIC AGREEMENT OF PROCESSORS INTERESTED IN A COMMON DECISION

4.1 INTRODUCTION AND MOTIVATION

There are many situations in which several processors (decision makers) with different on-line information (input) have to cooperate, combine their information and arrive at a common decision. The distributed function evaluation problem of Section 3.5 is an example. There are two issues involved here: first, the processors must reach consensus and, second, their final decision must be a desirable one, in some sense. Concerning the issues of what is a desirable decision, we abandon the "satisficing" point of view of Chapter 3 and we introduce a cost function. We will take, however, a more pragmatic approach than in Chapter 3: we will not insist that final decisions be as desirable as possible (that is, optimal). We concentrate on the requirement that consensus must be reached and, we require, as a secondary issue, that the final decision takes into account the cost function involved and that it is better than the decision that each processor would make if it were to rely exclusively on its own information.

To motivate our approach, we are interested in situations in which

(i) There are rigid time limits within which preliminary or final decisions must be made.

(ii) There are communications limitations, restricting the number and the nature of the messages that can be exchanged.

(iii) Conflict resolution procedures involving higher levels of the decision making hierarchy, are undesirable because they are likely to result in delays and tend to overload these higher levels.

We will be looking for a scheme which leads to consensus, while taking into consideration, directly or indirectly, the above requirements.

First and foremost, any scheme that would involve the centralization of all data (by communicating them to a predetermined processor) should be considered undesirable for the following reasons: it is often the case that too many data are available, which would overload communications channels; moreover good decisions can often be made on the basis of aggregations of the initial data; furthermore, if a processor is a model of a human, the plethora of data would saturate his short-term memory. This implies that it is preferable to communicate a few aggregate data. Determining an optimal way to "aggregate" is not a well-posed problem. If constraints are placed on the number of bits to be transmitted, the problem becomes computationally intractable, even if there is only a finite number of possible events and decisions (Sections 3.5). On the other hand, if a message is allowed to be any real number, all data can be coded in a single message. Also, for any fixed aggregation protocol, a processor could slightly change its message and code all information in the least significant bits of the message. (This is reminiscent of decentralized control problems in which a processor may observe the decisions of other processors - the so-called control sharing pattern [Aoki, 1973; Sandell and Athans, 1974]). Any such trick is very sensitive to noise in the channel and is effectively just a more complicated way of centralizing information. Since centralization was deemed undesirable in the first place, any indirect way of centralizing should be also undesirable and explicitly prohibited.

The above discussion, as well as the conclusions in Section 3.6, imply that a particular aggregation of the data should be chosen by means of some ad-hoc rule that guarantees that certain desirable characteristics are present. Unless some particular structure on the problem is assumed, the optimal decision (given a

processor's information) seems to be a very natural message that a processor could transmit and this is precisely the protocol that we study. Of course, a demonstration that our scheme leads to (approximate) consensus with fewer communications and/or computations than direct centralization has to rely on numerical experimentation and the answer will depend on the specific situation.

Problem Description and Overview

We consider the following situation: A set $\{1,\ldots,N\}$ of N processors possessing a common model of the world (same prior probabilities) and having the same cost function (common objective) want to make an optimal decision. Each processor bases its decision on a set of observations it has obtained and we allow these observations to be different for each processor. Given this setting, the decisions of the processors will be generally different. Aumann [1976] has shown, however, that agreement is guaranteed in the following particular case: If the decision to be made is the evaluation of the posterior probability of some event and if all processors' posteriors are common knowledge, then all processors agree. (In Aumann's terminology, common knowledge of an event means that all processors know it, all processors know that all processors know it, and so on, ad infinitum.)

The situation where each processor's posterior is common knowledge is very unlikely, in general. On the other hand, if agreement is to be guaranteed, posteriors have to be common knowledge. The problem then becomes how to reach a state of agreement where decisions are common knowledge, starting from an initial state of disagreement.

Geanakopoulos and Polemarchakis [1978] and Borkar and Varaiya [1982] gave the following natural solution to the above problem: Namely, processors start communicating to each other their tentative posteriors (or, in the formulation of [Borkar and Varaiya, 1982] the conditional expectation of a fixed random variable) and then update their own posterior, taking into account the new information they have received. In the limit, each processor's posterior converges (by the martingale convergence theorem) and assuming that "enough" communications have taken place, they all have to converge to a common limit.

The above results hold even when each processor obtains additional raw observations during the adjustment process and when the history of communications is itself random. Similar results were also proved for a detection problem [Borkar and Varaiya, 1982].

A related - and much more general situation is studied in this chapter; we assume that the processors are not just interested in obtaining an optimal estimate or a likelihood ratio, but their objective is to try to minimize some common cost function, given the available information. (Clearly, if each processor has a different cost function no agreement is possible even if each processor had identical information). In this setting, we assume that processors communicate to each other tentative decisions (which initially will be different). That is, at any time, a processor computes an optimal decision given the information it possesses and communicates it to other processors. Whenever a processor receives such a message from another processor its information essentially increases and it will, in general, update its own tentative decision, and so on. We prove that the qualitative results obtained in [Genakopoulos and Polemarchakis, 1978; Borkar and Varaiya, 1982] for the estimation

problem (convergence and asymptotic agreement) are also valid for the decision making problem for several, quite general, choices of the structure of the cost function. However, tentative decisions do not form a martingale sequence and a substantially different mathematical approach is required for the proofs. We point out that estimation problems are a special case of the decision problems studied in this Chapter, being equivalent to the minimization of the mean square error.

A drawback of the above setting is that each processor is assumed to have an infinite memory. We have implicitly assumed that the knowledge of a processor can only increase with time and, therefore, it has to remember the entire sequence of messages it has received in the past. There is also the implicit assumption that if a processor receives additional raw data from the environment, while the communication process is going on, these data are remembered forever. These assumptions are undesirable, especially if the processors are supposed to model humans, because limited memory is a fundamental component of the bounded rationality behavior of human decision makers [Simon, 1980]. We will therefore relax the infinite memory assumption and allow the processors to forget any portion of their past knowledge. We only constrain them to remember their most recent decision and the most recent message (tentative decision) coming from another processor. (For a particular class of communication protocols, we even allow them to forget their most recent decision). We then obtain convergence results similar to those obtained for the unbounded memory model, although in a slightly weaker sense.

A particular problem of interest is one in which all random variables are jointly Gaussian and the cost is a quadratic function of an unknown state of the world and the decision. It was demonstrated in [Borkar and Varaiya, 1982] that

the common limit to which decisions converge (for the estimation problem) is actually the centralized estimate, i.e. the estimate that would be obtained if all processors were to communicate their detailed observations. We prove (Section 4.4) that the same is true in the presence of memory limitations, provided that each processor never forget its own raw observations. (That is, it may only forget past tentative decisions sent to it by other processors). We indicate that, for linear quadratic Gaussian (LQG) problems, our scheme is essentially a decomposition algorithm for solving static linear estimation problems. As we point out in Section 4.4, this scheme has certain appealing features: there is significant parallelism in the computations which matches nicely with the assumed distribution of the data; also, in the course of the algorithm, acceptable estimates are obtained much earlier than the time that would be needed to compute the optimal estimate by centralizing the information. These tentative estimates can be very useful whenever there are strict time limits within which certain decisions have to be made.

We also consider (Section 4.5) a slightly different scheme in which each processor transmits its tentative decision to a coordinator. The latter evaluates a weighted average of the tentative decisions it has received and sends it back to all processors. We show that our results remain valid for this scheme as well and suggest an economic interpretation in which the coordinator can be viewed as some sort of market mechanism. We also show that making optimal tentative decisions corresponds to Nash strategies for a certain sequential game.

A weak point of the model is that, not only each processor has the same prior information and knows the statistics of the other processors' observations, but also

has the same model of the probabilistic mechanism that generates inter-processor communications. In particular, if this is a deterministic mechanism, a processor must know the precise history of communications between any pair of other processors, a strong requirement. If it is a stochastic mechanism, then there are two possibilities: either the history of communications becomes commonly known on-line (at the expense of additional communications) or each processor will have to make probabilistic inferences about the communications between all other processors. These weaknesses disappear, however, if every tentative decision is broadcast simultaneously to all other processors, at each stage. In that case the history of communications is simple, commonly known and easy to remember. (This will be the case, for example, if a set of experts with the same objective teleconfer and take turns into suggesting what they believe to be the optimal decision).

In Section 4.2 we define the model and the scheme to be studied, as well as a few special cases of particular interest. Section 4.3 contains the main results of this Chapter. Section 4.4. specializes to linear quadratic gaussian (static linear estimation) problems. Section 4.5 considers the scheme involving a coordinator to which we referred above. Section 4.6 discusses the case of processors with different models and Section 4.7 states the main conclusions of this Chapter.

Most of the results in this chapter appear in [Tsitsiklis and Athans, 1984].

4.2 MODEL FORMULATION

In this Section we present a mathematical formulation of the model informally described in Section 4.1. We start with the general assumptions and later proceed to the development of alternative specialized models to be consider (e.g. memory

limitations, particular forms of the cost function etc.). As far as the description of the sequence of communications and updates goes, we basically adopt the model of Borkar and Varaiya [1982] except that time is considered to be discrete. As in [Borkar and Varaiya, 1982], events are timed with respect to a common, absolute clock. As far as notation is concerned, we will use subscripts to denote time and superscripts to denote processors.

We assume that we are given a set $\{1,\ldots,N\}$ of N processors, an underlying probability space $(\Omega,\mathcal{F},\mathcal{P})$ and a real valued cost function $c:\Omega \times U \to R$, where U is the set of admissible values of the decision variable. It will be useful in the sequel to distinguish between elements of U and U-valued random variables. The letter v will be used to denote elements of U whereas u, w will be used to denote U-valued random variables (measurable functions from Ω to U).

<u>Assumption 4.2.1</u>: Either

(4.2.1.1): U is a finite set, or

(4.2.1.2): $U = R^n$, for some n.

<u>Assumption 4.2.2</u>: The cost function c is nonnegative and jointly measurable in (ω,v). Moreover, $E[c(v)] < \infty$, $\forall v \in U$. When Assumption (4.2.1.2) holds, we assume that there exists a positive and measurable function $A:\Omega \to R$ such that

$$A(\omega)||v_1-v_2||^2 \leq \frac{1}{2}[c(\omega,v_1)+c(\omega,v_2)] - c\left(\omega, \frac{v_1+v_2}{2}\right), \quad \omega\in\Omega, \quad v_1,v_2\in U \tag{4.2.1}$$

(Remark: If we fix v_1, v_2, $v_1 \neq v_2$ and take expectations of both sides of (4.2.1), it follows that A is integrable.)

Inequality 4.2.1 implies that c is a strictly convex function of v and strict convexity holds in a uniform way, for any fixed $\omega\in\Omega$. It also follows that $c(\omega,v)$ is continuous for any $\omega\in\Omega$. This assumption is satisfied, in particular, if c is twice continuously differentiable in v and its Hessian is positive definite, uniformly in v, for any fixed $\omega\in\Omega$.

We may use the function A, defined in Assumption 4.2.2, to define a new measure μ on $(\Omega,\mathcal{F})$ by

$$\mu(B) = \int_B A(\omega)\,d\mathcal{P}(\omega), \quad B\in\mathcal{F} . \tag{4.2.2}$$

This measure will be used in Section 4.3.

We now consider the generic situation facing processor i at some time n. Let $\mathcal{F}_n^i \subset \mathcal{F}$ be a σ-field of events describing the information possessed by processor i at time n. Because of Assumption 4.2.2, the conditional expectation $E[c(v)|\mathcal{F}_n^i]$ exists (is finite), is $\mathcal{F}_n^i$-measurable and is uniquely determined up to a set of measure zero, for any fixed $v\in U$. Processor i then computes a <u>tentative decision</u> u_n^i that minimizes $E[c(v)|\mathcal{F}_n^i]$. The following Lemma states that u_n^i is well-defined and $\mathcal{F}_n^i$-measurable.

<u>Lemma 4.2.1:</u> Under Assumptions 4.2.1.2, 4.2.2, there exists a $\mathcal{F}_n^i$-measurable random variable u_n^i, which is unique up to a set of measure zero, such that

$$E[c(u_n^i)|\mathcal{F}_n^i] \leq E[c(w)|\mathcal{F}_n^i] , \qquad \text{almost surely,} \tag{4.2.3}$$

for any U-valued, $\mathcal{F}_n^i$-measurable random variable w. The same results are true, (except for uniqueness) under Assumption 4.2.1.1, 4.2.2.

Proof: Let, for notational convenience, $g(\omega,v) = E[c(v)|F_n^i]$. Under Assumptions 4.2.1.2 and 4.2.2, $g(\omega,v)$ can be chosen so that it is jointly measurable, strictly convex in v and converges to infinity as v converges to infinity, for any $\omega\in\Omega$. Hence, $\forall\omega\in\Omega$, the infimum of $g(\omega,v)$ is attained by some $u_n^i(\omega)$ which is unique, because of strict convexity.

Let $Q = \left\{q_k\right\}_{k=1}^{\infty}$ be a countable dense subset of U. Then, by continuity of g, $\inf_{v\in Q} g(\omega,v) = \inf_{v\in U} g(\omega,v)$, $\forall\omega\in\Omega$. Moreover, $\inf_{v\in Q} g(\omega,v)$ is F_n^i-measurable. Let $\phi_m(\omega) = q_k$, where k is the smallest index such that

$$g(\omega,q_k) \leq \inf_{v\in U} g(\omega,v) + \frac{1}{m} .$$

Then ϕ_m is F_n^i-measurable and converges, for each ω, to u_n^i. Hence, u_n^i if F_n^i-measurable. Inequality (4.2.3) now follows from the definition of u_n^i and uniqueness is a consequence of strict convexity. The measurability of u_n^i under Assumption 4.2.1.1 is trivial. ■

We continue with a description of the process of communications between processors. When, at time n, processor i computes its tentative optimal decision u_n^i, it may communicate its realized value (say v_n^i) to any other processor. (If v_n^i is not unique, a particular minimizing v_n^i is selected according to some commonly known rule.) Whether, when and to which processors v_n^i is to be sent is a random event whose statistics are described by (Ω,F,P). In particular, it may depend on the data possessed by i at time n. So, we implicitly allow the processors to influence the process of communications, although we do not require this influence to be optimal in any sense. This allows the possibility of signalling additional information, beyond that contained

in v_n^i, by appropriately choosing when and to which processors to communicate. We allow the communication delays to be random but finite. We also assume that when a processor receives a message it knows the identity of the processor who sent it.

We now impose conditions on the number of messages to be communicated in the long run; these conditions are necessary for agreement to be guaranteed. Namely, we require that there is an indirect communication link from any processor to any other processor which is used an infinite number of times. This can be made precise as follows:

Let A(i) be the set of all processors that send an infinite number of messages to processor i, with probability 1. Then, we make the following assumption:

<u>Assumption 4.2.3</u>: There is a sequence $m_1,\ldots,m_{k+1}=m_1$ of not necessarily distinct processors such that $m_i \in A(m_{i+1})$, $i=1,2,\ldots,k$. Each processor appears at least once in this sequence.

The main consequence of Assumption 4.2.3, which will be repeatedly used, is the following: If $\{h^i:1,\ldots,N\}$ is a set of numbers such that $h^i \leq h^j$, $\forall j \in A(i)$, $\forall i$ then $h^i=h^j$, $\forall i,j$.

We continue with a more detailed specification of the operation of the processors. We introduce assumptions on the knowledge F_n^i which are directly related to the properties of the memory of processor i. A processor may receive (at any time) observations on the state of the world or receive tentative decisions (messages) of other processors. The knowledge of a processor at some time will be a subset (depending on the properties of its memory) of the total information it has received up to that time. We consider four alternative models of memory, formalized with the four assumptions that follow.

Let w_n^i be any message received by processor i at time n. Our most general assumption requires that w_n^i and u_{n-1}^i are remembered at time n:

Assumption 4.2.4: (Imperfect Memory) For all n, the σ-field F_n^i is such that u_{n-1}^i and w_n^i are F_n^i-measurable.

Assumption 4.2.4 can be further weakened if some restrictions are imposed on the communications protocol:

Assumption 4.2.5: (Imperfect Memory) For each n there exists a set I(n) of processors such that:

a) u_{n-1}^i is F_n^j-measurable, $\forall i \in I(n-1)$, $\forall j \in I(n)$

b) $F_n^i = F_{n-1}^i$, $\forall i$ not in I(n).

Intuitively, I(n) is the set of processors that update their decision at time n. Assumption 4.2.5 is satisfied by the following two common communication protocols, provided that processor i may obtain additional observations only at times such that $i \in I(n)$:

Ring Protocol: $I(n) = \{k\}$, where k is the unique integer such that $1 \leq k \leq N$ and $k+mN=n$, for some integer m. Here, exactly one processor updates at any time instance and communicates its tentative decision to the next processor and so on.

Star Protocol: $I(n) = \{1,\ldots,N-1\}$, if n is odd; $I(n) = \{N\}$, if n is even. Here all processors but the last one update simultaneously, communicate to the last processor who updates and communicates to all other processors and so on.

Assumption 4.2.6: (Own Data Remembered) Let G_n^i be the subfield of F describing all information that has been observed by processor i up to time n, except for the messages of other processors. We assume that $G_n^i \subset F_n^i$.

With Assumption 4.2.6, we allow the processors to forget the messages they received in the past, but they are restricted to remember all their past observations. In this case the total information available to all processors is preserved.

Assumption 4.2.7: (Perfect Memory) We let Assumptions 4.2.4 and 4.2.6 hold and assume that $\mathcal{F}_n^i \subset \mathcal{F}_{n+1}^i$, $\forall i,n$.

Whenever Assumption 4.2.7 holds, we will denote by $\mathcal{F}_\infty^i$ the smallest σ-field containing $\mathcal{F}_n^i$, for all n.

We now define a few special cases of particular interest:

(i) Estimation Problem: We are given a R^n-valued random vector x on $(\Omega,\mathcal{F},\mathcal{P})$. The objective is to minimize the mean square error. Hence, the cost function is $c(v) = (x-v)^T(x-v)$, where T denotes transpose. It is easy to see that this is a particular case of a strictly convex function covered by Assumption 4.2.2, with $A(\omega)$ being a constant.

(ii) Static Linear Quadratic Gaussian Decision Problem (LQG): Let x be an unknown random vector. Let the sequence of transmission and reception times be deterministic. We assume that the random variables observed by the processors are zero mean and, together with x, jointly normally distributed. We allow the total number of observations to be infinite. Let $U=R^n$. The objective is to fix v so as to minimize the expectation of the quadratic cost function $c(v) = v^TRv+x^TQv$, with $R>0$. If follows that the optimal tentative decision of processor i at time n is

$u_n^i = GE[x|\mathcal{F}_n^i] = E[Gx|\mathcal{F}_n^i]$, where G is a precomputable matrix. If we redefine the unknown vector x to be equal to Gx instead of x, we conclude that we may restrict to estimation problems, without loss of generality.

(iii) Finite Probability Spaces: Here we let Ω be a finite set. Then, there exist finitely many σ-fields of subsets of Ω. Strict convexity implies that for each σ-field $F_o \subset F$ and any $\omega \in \Omega$ of positive probability there exists a unique optimal tentative decision. This implies in turn that tentative decisions take values in some finite subset of U, with probability 1. We will therefore assume, without loss of generality, that U is a finite set.

We conclude this section by presenting a simple example that illustrates our scheme, under imperfect memory assumptions, where each processor forgets everything, except for its last decision and the most recent message it has received.

Suppose that we only have two processors who communicate to each other their tentative decisions at each instant of time. Let x, z_n^1, z_n^2, $n=1,2,\ldots$ be independent random variables, with known probability distributions. Let $y_n^i = h(x,z_n^i)$ be the observation of processor i at time n. Let $c(x,v)$ be a cost function, satisfying Assumption 4.2.2.

The tentative decisions are defined inductively as follows: Suppose that y_n^i, u_{n-1}^1, u_{n-1}^2 are known by processor i at time n. It then computes, for each v, the conditional cost $E[c(x,v)|y_n^i, u_{n-1}^1, u_{n-1}^2]$, which is equal to $g_n^i(y_n^i,u_{n-1}^1,u_{n-1}^2,v)$, for some Borel function g_n^i. Finally, it chooses a minimizing v, which is a function of y_n^i, u_{n-1}^1, u_{n-1}^2 and this is its tentative decision at time n. Hence, for appropriate functions f_n^1, f_n^2, we have

$$u_n^i = f_n^i(y_n^i,u_{n-1}^1,u_{n-1}^2), \qquad i=1,2$$

If we now define (Ω,F,P) to be the product of the probability spaces on which x, z_n^i are defined, and let F_n^i be the σ-field generated by y_n^i, u_{n-1}^1, u_{n-1}^2. Assumptions 4.2.3 and 4.2.4 are satisfied and the asymptotic properties of the above recursions can be analyzed within our general framework.

4.3 CONVERGENCE AND AGREEMENT RESULTS

In this Section we state and discuss our main results. Assumptions 4.2.2 and 4.2.3 will be assumed throughout the rest of this chapter and will not be explicitly mentioned in the statement of each theorem. Before proceeding to our results, we prove a Lemma to be used later.

Lemma 4.3.1: Let $\{u_n\},\{w_n\}$ be two sequences of U-valued random variables such that

$$\lim_{n\to\infty} E\left[c\left(\frac{u_n+w_n}{2}\right)\right] = \lim_{n\to\infty} E[c(u_n)] = \lim_{n\to\infty} E[c(w_n)] \ . \tag{4.3.1}$$

If Assumptions 4.2.1.2 and 4.2.2 hold, then $\lim_{n\to\infty}(u_n-w_n)=0$ in $L_2(\Omega,\mathcal{F},\mu)$* and in probability.

Proof: By Assumptions 4.2.1.2, 4.2.2 and equation (4.3.1)

$$\lim_{n\to\infty} E[A(\omega)||u_n-w_n||^2] \leq \lim_{n\to\infty} E\left[\frac{c(u_n)+c(w_n)}{2} - c\left(\frac{u_n+w_n}{2}\right)\right]=0 \tag{4.3.2}$$

which shows that (u_n-w_n) converges to zero in $L_2(\Omega,\mathcal{F},\mu)$. Therefore, it also converges in measure with respect to μ.

Recall that $A(\omega)>0$ and $\mu(B) = \int_B A(\omega)\,d\mathcal{P}(\omega)$, $\forall B\in\mathcal{F}$. Therefore, $\mu(B)=0$ implies $\mathcal{P}(B)=0$ and $\mathcal{P}$ is absolutely continuous with respect to μ. Let $B_n^\varepsilon = \{|u_n-w_n|\geq\varepsilon\}$. Since (u_n-w_n) converges to zero in measure μ, for any $\varepsilon>0$, we have $\lim_{n\to\infty}\mu(B_n^\varepsilon)=0$ and, by absolute continuity, $\lim_{n\to\infty}\mathcal{P}(B_n^\varepsilon)=0$, which shows that we have convergence in probability. □

Our first theorem holds under least restrictive assumptions on memory:

*That is, in the mean square with respect to the measure μ.

Theorem 4.3.1: We assume that transmissions and receptions are deterministic, that communication delays are bounded and that the time between two consecutive transmissions from processor j to processor i (with $j \in A(i)$) is bounded. Then, under Assumptions 4.2.1.2 (convex costs) and either Assumption 4.2.4 or 4.2.5 (imperfect memory):

a) $\lim_{n\to\infty}(u^i_{n+1}-u^i_n)=0$, in probability and in $L_2(\Omega,F,\mu)$.

b) $\lim_{n\to\infty}(u^i_n-u^j_n)=0$, $\forall i,j$, in probability and in $L_2(\Omega,F,\mu)$.

Proof: We start with the proof under Assumption 4.2.4. Since u^i_n is F^i_{n+1}-measurable, we have (by the minimizing property of u^i_{n+1}) $E[c(u^i_{n+1})]\leq E[c(u^i_n)]$. Since c is non-negative, $E[c(u^i_n)]$ converges to some constant g^i. We also note that $(u^i_{n+1}+u^i_n)/2$ is F^i_{n+1}-measurable and by taking the limit in the relation

$$\frac{1}{2}E[c(u^i_{n+1})+c(u^i_n)]\geq E\left[c\left(\frac{u^i_{n+1}+u^i_n}{2}\right)\right]\geq E[c(u^i_{n+1})]$$

we obtain $\lim_{n\to\infty}E[c((u^i_{n+1}+u^i_n)/2)]=g^i$. Lemma 4.3.1 then yields the first part of the theorem.

Let $j\in A(i)$. Then there exist sequences $\{m_k\}$ and $\{n_k\}$ of positive integers such that $\lim_{k\to\infty}m_k = \lim_{k\to\infty}n_k=\infty$ and m_k,n_k are the times of transmission and reception, respectively, of the k-th message from processor j to processor i. Therefore, $u^j_{m_k}$ is $F^i_{n_k}$-measurable, for all k, and $E[c(u^i_{n_k})]\leq E[c(u^j_{m_k})]$ which shows that $g^i\leq g^j$. Using Assumption 4.2.3, we conclude that $g^i=g^i$, $\forall i,j$.

We note that $(u^i_{n_k}+u^j_{m_k})/2$ is $F^i_{n_k}$-measurable and, therefore,

$$\frac{1}{2}E[c(u^i_{n_k})+c(u^j_{m_k})]\geq E\left[c\left(\frac{u^i_{n_k}+u^j_{m_k}}{2}\right)\right]\geq E[c(u^i_{n_k})] \quad . \tag{4.3.3}$$

Taking the limit in (4.3.3), using Lemma 4.3.1 and the boundedness assumptions on the communications, we obtain the second half of the theorem.

We now assume Assumption 4.2.5. Let $i(n)$ be a sequence of processors such that $i(n)\in I(n)$, $\forall n$. Then, $u^{i(n)}_n$ is $F^{i(n+1)}_{n+1}$-measurable and $E[c(u^{i(n)}_n)]$ is a decreasing sequence. Similarly with the first part of the proof, we conclude that $u^{i(n+1)}_{n+1}-u^{i(n)}_n$ converges to zero in $L_2(\Omega,F,\mu)$ and in probability. It follows that $u^i_{n+1}-u^i_n$ and $u^i_n-u^j_n$ converge to zero, for $j\in A(i)$. Using Assumption 4.2.3, $u^i_n-u^j_n$ converges to zero, for all i,j.■

Consider the following situation: At time zero, before any observations are obtained, the sequence of transmissions and receptions is selected in random, according to a statistical law which is independent from all observations to be obtained in the future and from $c(v)$, for any $v\in U$. In other words communications do not carry any information relevant to the decision problem, other than the content of the message being communicated. Suppose that the sequence of communications that has been selected becomes known to all processors. From that point on, the situation is identical with that of deterministic communications. In fact, a moment's thought will show that it is sufficient for the history of communications to become commonly known as it occurs: processor i only needs to know, at time n, what communications have occured up to that time, so that it can interpret correctly the meaning of the messages it is receiving.

We can formalize these ideas as follows: We are given a product probability space $(\Omega \times \Omega^*, F \times F^*, P \times P^*)$ where (Ω, F, P) describes the decision problem and where (Ω^*, F^*, P^*) describes the communications process. We assume that for each $\omega^* \in \Omega^*$, the resulting process of communications satisfies the assumptions of Theorem 4.3.1. Then, note that for each $\omega^* \in \Omega^*$ we obtain a distributed decision problem on (Ω, F, P) with deterministic communications. In that case:

<u>Theorem 4.3.2</u>: Under Assumptions 4.2.1.2, either 4.2.4 or 4.2.5, and independent, commonly known communications (as described above), $\lim_{n\to\infty}(u^i_{n+1}-u^i_n) =$ $= \lim_{n\to\infty}(u^i_n-u^j_n)=0$, in probability with respect to $P \times P^*$.

<u>Proof</u>: Theorem 4.3.1 and the discussion preceding the statement of Theorem 4.3.2 show that $\lim_{n\to\infty}(u^i_{n+1}-u^i_n)=0$, in probability with respect to P, for all $\omega^* \in \Omega^*$. Let $X_n(\omega,\omega^*)$ be the characteristic function of the set $\{(\omega,\omega^*): ||u^i_{n+1}-u^i_n||<\varepsilon\}$. Then

$$\lim_{n\to\infty}\int X_n d(PxP^*) = \lim_{n\to\infty}\iint X_n dPdP^* = \int \lim_{n\to\infty}\int X_n dPdP^*=1$$

(The first equality holds by the Fubini theorem; the second by the dominated convergence theorem; the third by convergence of $(u^i_{n+1}-u^i_n)$ to zero, with respect to the probability measure P.) This shows that $u^i_{n+1}-u^i_n$ convergence to zero in probability with respect to PxP^*. Similar steps show that $u^i_n-u^j_n$ also converges to zero, in probability.■

Strictly speaking, Theorems 4.3.1 and 4.3.2 do not guarantee convergence of the decisions of each agent. Suppose, however, that the agents operate under the

following rule: Fix some small $\gamma>0$. Let the sequence of communications and updates of tentative decisions take place until $|u_n^i-u_n^j|<\gamma$, $\forall i,j$ (small disagreement) and $|u_{n+1}^i-u_n^i|<\gamma$, $\forall i$ (small foreseeable changes in tentative decisions). Then, we obtain:

Corollary 4.3.1: With the above rule and the assumptions of Theorems 4.3.1 or 4.3.2, the process terminates in finite time, with probability 1, for any $\gamma>0$.

Proof: By Theorem 4.3.1, the UxU-valued sequence of random variables $(u_n^i-u_n^j,\ u_{n+1}^i-u_n^i)$ converges to (0,0) in probability. It therefore contains a subsequence converging to (0,0), almost surely. Therefore, $\forall\gamma>0$ and for almost all $\omega\in\Omega$, $\exists n_o$ such that $|u_{n_o}^i-u_{n_o}^j|<\gamma$, $|u_{n_o+1}^i-u_{n_o}^i|<\gamma$ and the termination condition is eventually satisfied with probability one.■

When Ω and U are finite, convergence and agreement are obtained after finitely many stages:

Theorem 4.3.3: If Ω and U are finite sets, if each processor communicates all the values of v that minimize $E[c(v)|F_n^i]$ and if Assumption 4.2.4 holds, then there exists some positive integer M such that

$$u_M^i = u_M^j,\quad \forall i,j \quad\text{and}\quad u_{M+n}^i = u_M^i,\quad \forall i,\ \forall n,\ \forall\omega\in\Omega.$$

Proof: Because of the finiteness of Ω, there exists a finite (non-random) time after which communications (conditioned on past events) are deterministic. We may take that time as the initial time and assume, without loss of generality, that all communications are deterministic.

Let u_n^i be the set of elements of U which are optimal, given F_n^i. Let w_n^i denote a F_n^i-measurable random variable such that $w_n^i(\omega) \in u_n^i(\omega)$, $\forall \omega \in \Omega$. (Note that $E[c(w_n^i)]$ is independent of how w_n^i has been selected). By finiteness of Ω and U, there exist finitely many U-valued random variables and, since $E[c(w_{n+1}^i)] \leq E[c(w_n^i)]$, we conclude that there exists some positive integer T and some g^i such that $E[c(w_n^i)] = g^i$, $\forall n > T$. For any $n > T$, $E[c(w_n^i)] = E[c(w_{n+1}^i)]$ and since w_n^i is F_{n+1}^i-measurable, w_n^i minimizes $E[c(w)]$ over all F_{n+1}^i-measurable random variables. Hence, $w_n^i(\omega) \in u_{n+1}^i(\omega)$, $\forall \omega \in \Omega$ which shows that $u_n^i(\omega) \subset u_{n+1}^i(\omega)$, $\forall \omega \in \Omega$. Again, by finiteness of U and Ω, there exists some positive integer M such that $u_{n+1}^i(\omega) = u_n^i(\omega)$, $\forall n > M$, $\forall \omega \in \Omega$, $\forall i$.

If $j \in A(i)$, there exist $m,n > M$ such that w_m^j is F_n^i-measurable and this shows that $g^i \leq g^j$. By Assumption 4.2.3, we obtain $g^i = g^j$, $\forall i,j$. Therefore, w_m^j minimizes $E[c(w)]$ over all F_n^i-measurable random variables and, therefore, $w_m^j(\omega) \in u_n^i(\omega)$, or, $u_m^j(\omega) \subset u_n^i(\omega)$, $\forall \omega \in \Omega$. Recalling Assumption 4.2.3, we obtain $u_m^j(\omega) = u_n^i(\omega)$, $\forall i,j$, $\forall m,n > M$, $\forall \omega$. ■

Strictly speaking, tentative decisions in the above theorem are not elements of U but subsets of U. This is to compensate for the possibility of non-uniqueness of optimal tentative decisions. The equalities appearing in Theorem 4.3.3 have to be interpreted, therefore, as equalities of sets.

We now assume that the processors have perfect memory. We obtain results similar to Theorems 4.3.1 and 4.3.2 under much more relaxed assumptions on the communications process. Namely, we only need to assume the following:

Assumption 4.3.1: Let M_k^{ij} be the k-th message sent by processor j to processor i. We assume that when processor i receives M_k^{ij}, it knows that this is indeed the k-th message sent to it by processor j.

Remark: This assumption is trivially satisfied if messages arrive at exactly the same order as they are sent, with probability 1.

Lemma 4.3.2: Let T be a finite stopping time of an increasing family $\{F_n\}$ of σ-fields. Let u_n, n=1,2,... be random variables that minimize $E[c(w)|F_n]$, almost surely, over all F_n-measurable random variables w. Then, u_T minimizes $E[c(w)]$ over all F_T-measurable random variables w, where u_T is defined to be equal to u_n if T=n.

Proof: Let X_n be the indicator function of the set $\{\omega:T(\omega)=n\}$. Since T is a stopping time, X_n is F_n-measurable. Note that $X_n c(u_n) = X_n c(u_T)$. Let w be a F_T-measurable random variable and note that $X_n c(w) = X_n c(X_n w)$ and $X_n w$ is F_n-measurable. Therefore,

$$E[X_n c(w)|F_n] = X_n E[c(X_n w)|F_n] \geq X_n E[c(u_n)|F_n] = E[X_n c(u_n)|F_n] = E[X_n c(u_T)|F_n].$$

Taking expectations, we obtain

$$E[X_n c(w)] \geq E[X_n c(u_T)]$$

and summing over all n's (and using the monotone convergence theorem to interchange summation and expectation) we obtain $E[c(w)] \geq E[c(u_T)]$. ∎

Theorem 4.3.3: Under Assumptions 4.2.1.2 (convex costs), 4.2.7 (perfect memory) and 4.3.1, there exists a U-valued random variable u^* such that $\lim_{n\to\infty} u_n^i = u^*$, $\forall i$, in probability and in $L_2(\Omega,F,\mu)$.

<u>Proof</u>: Since u_n^i is F_{n+1}^i-measurable, we have $E[c(u_{n+1}^i)] \leq E[c(u_n^i)]$. Since c is non-negative, $E[c(u_n^i)]$ converges to some constant g^i. We also note that $(u_n^i+u_{n+m}^i)/2$ is F_{n+m}^i-measurable. Therefore, $E[c((u_{n+m}^i+u_n^i)/2)] \geq E[c(u_{n+m}^i)] \geq g^i$. Fix some $\varepsilon>0$, and let n be large enough so that $E[c(u_n^i)] \leq g^i+\varepsilon$. Then, using Assumption 4.2.2, we obtain $E[A(\omega)||u_{n+m}^i-u_n^i||^2] \leq \varepsilon$, $\forall m \geq 0$. Therefore, $\{u_n^i\}$ is a Cauchy sequence in $L_2(\Omega,F,\mu)$. By the completeness of L_2 spaces, there exists a U-valued random variable u^i such that $\lim_{n\to\infty} u_n^i = u^i$, in $L_2(\Omega,F,\mu)$ and, therefore, in probability, with respect to P. (The proof of the last implication is contained in the proof of Lemma 4.3.1). Since

$$E[E[c(u_{n+1}^i)|F_{n+1}^i]|F_n^i] \leq E[E[c(u_n^i)|F_{n+1}^i]|F_n^i] = E[c(u_n^i)|F_n^i] ,$$

$\{E[c(u_n^i)|F_n^i], n=1,2,\ldots\}$ is a supermartingale, with respect to $\{F_n^i\}$. Moreover, since for any fixed $v \in U$, $E[c(u_n^i)|F_n^i] \leq E[c(v)|F_n^i]$, it is a uniformly integrable supermartingale [Meyer, 1966, Theorem T19, p.90].

Let $j \in A(i)$. Let m_k, n_k be the times of transmission and reception, respectively, of the k-th message from j to i. Because of Assumption 4.3.1, m_k and n_k are stopping times of $\{F_n^j\}$, $\{F_n^i\}$, respectively, and since $j \in A(i)$, they are almost surely finite stopping times, for all k. Moreover, $k \leq m_k \leq n_k$ and by the optional sampling theorem [Meyer, 1966, Theorem T28, p.90].

$$E[c(u_k^i)] \geq E[c(u_{n_k}^i)] \geq g^i$$

which shows that $\lim_{k\to\infty} E[c(u_{n_k}^i)] = g^i$. Similarly, $\lim_{k\to\infty} E[c(u_{m_k}^j)] = g^j$.

Note that $u^j_{m_k}$ is $F^i_{n_k}$-measurable and, by Lemma 4.3.2, $E[c(u^j_{m_k})] \geq E[c(u^i_{n_k})]$. Taking the limit, we obtain $g^j \geq g^i$, and by Assumption 4.2.3, $g^i = g^j$, $\forall i,j$.

We now take the limit of the inequalities

$$\frac{1}{2} E[c(u^i_{n_k}) + c(u^j_{m_k})] \geq E\left[c\left(\frac{u^i_{n_k} + u^j_{m_k}}{2}\right)\right] \geq E[c(u^i_{n_k})]$$

to obtain $\lim_{k\to\infty} E[c((u^i_{n_k} + u^j_{m_k})/2)] = g$ and, by Lemma 4.3.1, $\lim_{k\to\infty}(u^i_{n_k} - u^j_{m_k}) = 0$, in $L_2(\Omega, F, \mu)$ and in probability.

We also take the limit of the inequalities

$$\frac{1}{2} E[c(u^i_{n_k}) + c(u^i_k)] \geq E\left[c\left(\frac{u^i_{n_k} + u^i_k}{2}\right)\right] \geq E[c(u^i_{n_k})]$$

to obtain $\lim_{k\to\infty} E[c((u^i_k + u^i_{n_k})/2)] = g^i$ and, by Lemma 4.3.1, $\lim_{k\to\infty}(u^i_{n_k} - u^i_k) = 0$. Similarly, we obtain $\lim_{k\to\infty}(u^j_{m_k} - u^j_k) = 0$, which shows that $u^i = u^j$, almost surely.■

For estimation problems ($u^i_n = E[x|F^i_n]$), Theorem 4.3.4 can be slightly strengthened: [Borkar and Varaiya, 1982, Theorem 2].

<u>Theorem 4.3.5</u>: For estimation problems, under the assumption of Theorem 4.3.4, convergence to u* takes place with probability 1.

We now consider the case where U is finite but (unlike Theorem 4.3.3) Ω is allowed to be infinite. Several complications may arise, all of them due to the fact that optimal decisions, given some information, are not guaranteed to be unique. We discuss these issues briefly, in order to motivate the next theorem.

Suppose that $U = \{v_1, v_2\}$. It is conceivable that $E[c(v_1)|F^i_n] - E[c(v_2)|F^i_n]$ is never zero and changes sign an infinite number of times, on a set of positive probability.

In that case, the decisions of processor i do not convege. Even worse, it is conceivable that $E[c(v_1)|F_n^i] > E[c(v_2)|F_n^i]$ and $E[c(v_1)|F_n^j] < E[c(v_2)|F_n^j]$, for all n and for all ω in a set of positive probability, in which case processors i and j disagree forever. It is not hard to show that in both of the above cases $E[c(v_1)|F_\infty^i] = E[c(v_2)|F_\infty^i]$, on a set of positive probability and this non-uniqueness is the source of the pathology. The following theorem states that convergence and agreement are still obtained, provided that we explicitly exclude the possibility of non-uniqueness.

<u>Theorem 4.3.6</u>: Under Assumption 4.2.1.1 (finite U) and 4.2.7 (perfect memory) and if the random variable u^i that minimizes $E[c(w)]$ over all F_∞^i-measurable random variables is unique up to a set of measure zero, for all i, then $\lim_{n\to\infty} u_n^i = u^i$, almost surely, and $u^i = u^j$, $\forall i,j$.

<u>Proof</u>: Fix some $v \in U$ and let $B = \{\omega: u^i(\omega) = v\}$. Then, $E[c(v)|F_\infty^i] < E[c(v^*)|F_\infty^i]$, $\forall v^* \neq v$ for almost all $\omega \in B$. By the martingale convergence theorem [Meyer, 1966, Theorem T17, p.84], we conclude that for almost all $\omega \in B$, there exists some $N(\omega)$ such that

$$E[c(v)|F_n^i] < E[c(v^*)|F_n^i] \qquad \forall n \geq N(\omega)$$

Therefore, $\lim_{n\to\infty} u_n^i(\omega) = v$, for almost all $\omega \in B$ and, by considering the other elements of U as well, $\lim_{n\to\infty} u_n^i = u^i$, almost surely.

If $j \in A(i)$, u^j is F_∞^i-measurable and $E[c(u^j)] \geq E[c(u^i)]$. By Assumption 4.2.3, $E[c(u^i)] = E[c(u^j)]$, $\forall i,j$. Therefore, for $j \in A(i)$, u^j minimizes $E[c(w)]$ over all F_∞^i-measurable random variables and by the assumptions of the theorem, $u^i = u^j$, almost surely. Using Assumption 4.2.3 once more, we obtain $u^i = u^j$, $\forall i,j$.■

Although the preceding theorems guarantee that (under certain conditions) all processors will agree, nothing has been said concerning the particular decision to which all agents' decisions converge. In particular, some simple examples show that the limit decision can be different from the optimal centralized solution (that is, the solution to be obtained if all agents were to communicate all their information). On the other hand, the centralized solution is reached for LQG problems, under the perfect memory assumption [Borkar and Varaiya, 1982] and is also reached generically for an estimation problem on a finite probability space [Geanakopoulos and Polemarchakis, 1978]. This issue will be touched again in the next section.

Robustness with respect to Communication Noise

Schemes that centralize information by coding (e.g. by using the least significant bits of the allowed messages [Aoki, 1973; Sandell and Athans, 1974] tend to require high bandwidth and are sensitive to noise in the communication channel. In our scheme, although real numbers are being transmitted (infinite information content), the least significant bits are not as essential. As a result, the qualitative convergence properties of our scheme are retained even if communications of the tentative decisions are assumed to be noisy. We provide a proof of this fact for estimation problems, under the perfect memory assumption.

Suppose, as before, that at random times processor j communicates its optimal tentative decision u_n^j. However, the message received by the other processors is $\tilde{u}_n^j = u_n^j + q_n^j$, where q_n^j is a random vector representing the noise in the channel. For simplicity, we assume that the noise vectors are independent, identically distributed.

Theorem 4.3.7: Assume noisy communications (as described above). For estimation problems, under Assumption 4.2.7 (perfect memory), there exists a U-valued random variable u* such that $\lim_{n\to\infty} u_n^i = u^*$, $\forall i$, with probability 1.

Proof: Let x be the unknown vector to be estimated. Then $u_n^i = E[x|F_n^i]$ and converges almost surely to $u^i = E[x|F_\infty^i]$. Moreover, u^i minimizes $E[c(w)]$ over all F_∞^i-measurable random variables w. Let $j \in A(i)$ and let m_k be the time of transmission of the k-th message from j to i. Note that $\frac{1}{M} \sum_{k=1}^{M} \hat{u}_{m_k}^j$ is $F_{n_M}^i$-measurable* (and hence F^i-measurable) and converges to u^j almost surely. Therefore, u^j is F_∞^i-measurable and $E[c(u^i)] \leq E[c(u^j)]$ and, by Assumption 4.3.3, $E[c(u^i)] = E[c(u^j)]$, $\forall i,j$. The minimizing property of u^i and the strict convexity of the quadratic cost function imply that $u^i = u^j$, almost surely.■

4.4 THE LINEAR QUADRATIC GAUSSIAN (LQG) MODEL

In this Section we specialize and strengthen some of our results by restricting to the Linear Quadratic Gaussian model described in Section 4.2. (Recall that any such problem is equivalent to an estimation problem; therefore, $u_n^i = \hat{x}_i^n = E[x|F_n^i]$, for some random vector x). Theorems 4.3.1, 4.3.4 and 4.3.5 are applicable. Moreover, the results of [Borkar and Varaiya, 1982] guarantee that, under Assumption 4.2.7 (perfect memory), u_n^i converges to the optimal centralized estimate, given the information possessed by all processors. The following theorem states that the same is true under the weaker Assumption 4.2.6.

---

*Here, n_M is the time of reception of the M-th message from j to i.

<u>Theorem 4.4.1</u>: For the LQG problem, under the assumptions of Theorem 4.3.1 and Assumption 4.2.6 (imperfect memory; own data remembered), $\lim_{n\to\infty}\hat{x}_n^i=\hat{x}$, in the mean square, where $\hat{x}=E[x|F_\infty]$ and F_∞ is the smallest σ-field containing F_n^i, for all i,n.

<u>Proof</u>: As is usual in linear least squares estimation, we use the setting of Hilbert spaces of square integrable random variables. Let G be a Hilbert space of zero mean, jointly Gaussian random variables on (Ω,F,P) such that each component of the unknown vector x and the observations belong to G. The inner product in G is defined by $\langle x,y\rangle = E[xy]$.

For each processor i, let H^i denote the smallest closed subspace of G containing all observations obtained by it. Let H_n^i be the smallest closed subspace of G containing all observations obtained by processor i up to time n. (Note that H_n^i does not contain all random variables known by processor i at time n, because it does not need to contain any of the messages received by processor i). Note also that, by Assumption 4.2.6, $H_n^i \subset H_{n+1}^i \subset H^i$ and that $\sum_{k=1}^{N} H_n^k$ is the total knowledge available to all processors at time n. The centralized estimate is the projection of x on $\sum_{k=1}^{N} H^k$. We assume, without loss of generality, that x is a scalar random variable, since each component can be separately estimated.

Let $\hat{x}_n^i = E[x|F_n^i]$ and $e_n^i = x-\hat{x}_n^i$ and, by the orthogonality of errors and observations, we have $E[xy] = E[\hat{x}_n^i y]$, $\forall y\in H_n^i$. As in the proof of Theorem 4.3.1 we have $||e_{n+1}^i||^2 \le ||e_n^i||^2$, $\forall n,i$ which implies that

$$||x||^2 \ge ||\hat{x}_{n+1}^i||^2 \ge ||\hat{x}_n^i||^2 \tag{4.4.1}$$

In particular, (4.4.1) implies that $\{\hat{x}_n^i\}$ is a norm-bounded sequence. By the weak local sequential compactness of Hilbert spaces [Yosida, 1980, p.126], $\{\hat{x}_n^i\}$ contains a weakly convergent subsequence $\{\hat{x}_{n_\ell}^i\}$. In other words, there exists an element $\hat{x}_\infty \in G$ such that $\langle y, \hat{x}_{n_\ell}^i \rangle$ converges to $\langle y, \hat{x}_\infty \rangle$, $\forall y \in G$. Moreover, $\hat{x}_n^i \in \sum_{k=1}^{N} H_n^k \subset \sum_{k=1}^{N} H^k$ and since closed subspaces are also weakly closed [Yosida, 1980, Theorem 11, p. 125], $\hat{x}_\infty \in \sum_{k=1}^{N} H^k$. Now let $y \in H_n^i$. Then, $\langle y, \hat{x}_{n_\ell}^i \rangle = \langle y, x \rangle$, for all ℓ such that $n_\ell \geq n$, which implies that $\langle y, \hat{x}_\infty \rangle = \langle y, x \rangle$. Moreover, the sequence of subspaces $\{H_n^i\}$ generates H^i which implies that $\langle y, \hat{x}_\infty \rangle = \langle y, x \rangle$, $\forall y \in H^i$.

By Theorem 4.3.1, $(x_n^i - x_n^j)$ converges in the mean square (and therefore weakly) to zero, which implies that $\hat{x}_\infty$ is also a weak limit point of $\{\hat{x}_{n_\ell}^j\}$. The same argument as before shows that $\langle y, \hat{x}_\infty \rangle = \langle y, x \rangle$, $\forall y \in H^j$, $\forall j$. Therefore, $\langle y, \hat{x}_\infty \rangle = \langle y, x \rangle$, $\forall y \in \sum_{k=1}^{N} H^k$. But this is exactly the condition that $\hat{x}_\infty$ is the centralized estimate, given the observations of all processors. So, $\{\hat{x}_n^i\}$ has a unique weak limit point, which is the same for all i and coincides with the centralized estimate.

It only remains to show that $\hat{x}_n^i$ converges to $\hat{x}_\infty$ strongly (in the mean square). We know from [Yosida, 1980, p.120] that $||\hat{x}_\infty|| \leq \liminf_{n\to\infty} ||\hat{x}_n^i||$. On the other hand,

$$||x||^2 - ||\hat{x}_\infty||^2 = ||x - \hat{x}_\infty||^2 \leq ||x - \hat{x}_n^i||^2 = ||x||^2 - ||\hat{x}_n^i||^2 \tag{4.4.2}$$

which shows that $||\hat{x}_\infty|| \geq \limsup_{n\to\infty} ||\hat{x}_n^i||$. Therefore, $||\hat{x}_\infty|| = \lim_{n\to\infty} ||\hat{x}_n^i||$ and by Theorem 8, p.124 of [Yosida, 1980], we conclude that $\lim_{n\to\infty} ||\hat{x}_n^i - \hat{x}_\infty||^2 = 0$. ■

Note that Theorem 4.4.1 is much stronger than Theorem 4.3.1 which was proved for the general case of imperfect memory. We have here convergence to a limit solution which is also guaranteed to be the optimal centralized solution.

Our next result concerns the finite dimensional LQG problem in which the total number of observations is finite. Namely, the smallest σ-field containing F_n^i for all i, n is generated by a finite number of (jointly Gaussian) random variables. In that case, the centralized solution is going to be reached by all processors in a finite number of stages, provided that all processors have perfect memory.

<u>Theorem 4.4.2</u>: For the LQG problem with finitely many observations and under Assumption 4.2.7 (perfect memory), the centralized solution is reached by all processors in a finite number of stages.

<u>Proof</u>: We use again the Hilbert space formalism of the previous proof. Let G_n^i be the subspace of G describing the knowledge of processor i at time n (both its observations and the messages it has received). By Assumption 4.2.7, we have $G_n^i \subset G_{n+1}^i \subset G$. Since G can be chosen to be finite dimensional, there exists some M (depending on the sequence of communications but deterministic) such that $G_{M+n}^i = G_M^i$, $\forall n \geq 0$, $\forall i$. Equivalently, $\hat{x}_{M+n}^i = \hat{x}_M^i$, $\forall n \geq 0$, $\forall i$, and by Theorem 4.4.1, $\hat{x}_M^i = \hat{x}_M^j = \hat{x}_\infty$, $\forall i,j$. ■

Theorem 4.4.1 and 4.4.2 imply that the scheme considered in this Section may be viewed as an algorithm for solving static linear estimation problems, an issue that we discuss below.

The intuitive argument behind Theorem 4.4.2 is the following: once a processor has received enough messages, it is able to infer exactly the values of the observations

of the other processors (or of some appropriate linear combinations of these observations) and compute the centralized solution itself. So, communicating optimal tentative decisions is in this case just another way for communicating all information to all other processors. This scheme does not seem to have any particular advantages (in terms of communication and computation requirements) over the scheme where each processor communicates all its data directly.

However, the scheme of Theorem 4.4.1 (imperfect memory) seems to have some appealing features, which we will now discuss. Suppose that a central processor obtains a NM-dimensional vector of observations. Instead of inverting the NMxNM covariance matrix (which would require $O(N^3M^3)$ operations) it splits its observations into N M-dimensional vectors. Each M-dimensional vector is to be handled by a different processor and suppose that the scheme of Theorem 4.4.1 is to be used. If the ring protocol is to be used, there will be one inversion for each M-dimensional vector of observations and for each round. This leads to $O(NM^3)$ operations per round. If, for example, an acceptable estimate is obtained after O(N) rounds, the final objective will have been accomplished with a total of $O(N^3M^3)$ operations, which is one order of magnitude less than the standard algorithm. It is not hard to show that if the noises in observations belonging to different blocks of data are uncorrelated, agreement on the optimal is obtained after two rounds only. Accordingly, if the noises in observations in different blocks are weakly correlated, we expect this scheme to be faster than the standard algorithm. This suggests that our scheme corresponds to a potentially advantageous <u>decomposition</u> algorithm for static linear estimation problems. This algorithm has some similarities with those suggested by Laub and Bailey [1978].

We now continue with an analysis of the convergence rate of the decomposition algorithm, for two particular communication protocols. Let G be a Hilbert space of zero-mean, Gaussian random variables. Let $H^1,\ldots,H^N$ be closed subspaces and let H be the smallest subspace of G containing $H^1,\ldots,H^N$. Let $x\in G$ be a random variable to be estimated.

Algorithm 1 (Star Protocol):

$$y_1^o = 0 \tag{4.4.3}$$

$$y_n^o = E[x|y_n^1,\ldots,y_n^N], \qquad n>1 \tag{4.4.4}$$

$$y_{n+1}^i = E[x|H^i,y_n^o], \qquad n\geq 1,\ i\in\{1,\ldots,N\} \tag{4.4.5}$$

Algorithm 2 (Ring Protocol):

$$y_1^1 = E[x|H^1], \tag{4.4.6}$$

$$y_n^{i+1} = E[x|H^{i+1},y^i], \qquad n\geq 1,\ i\in\{1,\ldots,N-1\}, \tag{4.4.7}$$

$$y_{n+1}^1 = E[x|H^1,y_n^N], \qquad n\geq 1\ . \tag{4.4.8}$$

By Theorem 4.4.1, y_n^i converges to $E[x|H]$, in the mean square, for either of the above algorithms. Theorem 4.4.3 below states that the mean square error converges geometrically to the optimal mean square error, which is a stronger result. Similar convergence rate results may be proved for a variety of other protocols as well, with much more "chaotic" sequences of communications. However, the proof for these two examples are sufficient to illustrate the main idea of a more general proof.

<u>Theorem 4.4.3</u>: Let H be finite dimensional. For any $z \in H$, $z \neq 0$, let

$$\alpha^i(z) = \frac{||E[z|H^i]||^2}{||z||^2} \tag{4.4.9}$$

and

$$\alpha = \inf_{\substack{z \in H \\ z \neq 0}} \max_i \alpha^i(z). \tag{4.4.10}$$

Then, $0 < \alpha \leq 1$ and

a) For the star protocol

$$||y^o_{n+1} - \hat{x}||^2 \leq (1-\alpha) \; ||y^o_n - \hat{x}||^2 . \tag{4.4.11}$$

b) For the ring protocol

$$||y^N_{n+1} - \hat{x}||^2 \leq (1 - \frac{\alpha}{N}) \; ||y^N_n - \hat{x}||^2 , \tag{4.4.12}$$

where

$$\hat{x} = E[x|H].$$

<u>Proof</u>: The inequality $\alpha \leq 1$ is trivial. Suppose, to derive a contradiction, that $\alpha = 0$. It is easy to see that the infimum, in the definition of α, is attained by some $z \in H$, $z \neq 0$. For that particular z, we will have $\alpha^i(z) = 0$, $\forall i$, which implies that z is orthogonal to H^i, $\forall i$. But this contradicts the assumptions $z \in H$, $z \neq 0$.

a) Using (4.4.9) and simple orthogonality relations:

$$\begin{aligned} ||\hat{x} - E[\hat{x}|H^i, y^o_n]||^2 &= ||\hat{x} - y^o_n - E[\hat{x} - y^o_n|H^i, y^o_n]||^2 = \\ &= ||\hat{x} - y^o_n||^2 - ||E[\hat{x} - y^o_n|H^i, y^o_n]||^2 \leq \\ &\leq ||\hat{x} - y^o_n||^2 - ||E[\hat{x} - y^o_n|H^i]||^2 = \\ &= ||\hat{x} - y^o_n||^2 (1 - \alpha^i(\hat{x} - y^o_n)) . \end{aligned} \tag{4.4.13}$$

Therefore,

$$\min_i ||\hat{x}-E[\hat{x}|H^i,y_n^o]||^2 \leq ||\hat{x}-y_n^o||^2(1-\max_i \alpha^i(\hat{x}-y_n^o)) \leq$$

$$\leq (1-\alpha)||\hat{x}-y_n^o||^2 . \tag{4.4.14}$$

Now note that

$$E[\hat{x}|H^i,y_n^o]=E[E[x|H]|H^i,y_n^o] = E[x|H^i,y_n^o] = y_{n+1}^i \tag{4.4.15}$$

and that

$$||\hat{x}-y_{n+1}^o||^2 \leq \min_i ||\hat{x}-y_{n+1}^i||^2 . \tag{4.4.16}$$

Inequalities (4.4.14)-(4.4.16) yield (4.4.11).

b) For notational convenience, we will interpret y_{n+1}^{i-1}, with i=1, as y_n^N.

Again, starting from (4.4.9), we have:

$$\alpha||\hat{x}-y_n^N||^2 \leq \max_i \alpha^i(\hat{x}-y_n^N)\cdot ||\hat{x}-y_n^N||^2 =$$

$$= \max_i ||E[\hat{x}-y_n^N|H^i]||^2 \leq$$

$$\leq \max_i ||E[\hat{x}-y_n^N|H^i,y_{n+1}^{i-1}]||^2 =$$

$$= \max_i ||E[y_{n+1}^i-y_n^N|H^i,y_{n+1}^{i-1}]||^2 \leq$$

$$\leq \max_i ||y_{n+1}^i-y_n^N||^2 . \tag{4.4.17}$$

Let

$$\tilde{y}_{n+1}^i = y_{n+1}^i - y_{n+1}^{i-1} , \tag{4.4.18}$$

and note that

$$y_{n+1}^{i-1} = E[x|y_{n+1}^{i-1}] = E[E[x|H^i, y_{n+1}^{i-1}]|y_{n+1}^{i-1}] = E[y_{n+1}^{i}|y_{n+1}^{i-1}] \ . \tag{4.4.19}$$

This implies that $\tilde{y}_{n+1}^{i}$ is orthogonal to y_{n+1}^{i-1}.

Therefore,

$$||y_{n+1}^{i}||^2 = ||y_{n+1}^{i-1}||^2 + ||\tilde{y}_{n+1}^{i}||^2, \tag{4.4.20}$$

$$||y_{n+1}^{N}||^2 = ||y_{n}^{N}||^2 + \sum_{i=1}^{N} ||\tilde{y}_{n+1}^{i}||^2 \ . \tag{4.4.21}$$

Using (4.4.20) and (4.4.21) in (4.4.17), we have

$$\alpha||\hat{x}-y_n^N||^2 \leq \max_i ||\sum_{k=1}^{i} \tilde{y}_{n+1}^{k}||^2 \leq$$

$$\leq \max_i i \sum_{k=1}^{i} ||\tilde{y}_{n+1}^{k}||^2 =$$

$$= N \sum_{i=1}^{N} ||\tilde{y}_{n+1}^{i}||^2 =$$

$$= N(||y_{n+1}^{N}||^2 - ||y_n^N||^2) =$$

$$= N(||\hat{x}-y_n^N||^2 - ||\hat{x}-y_{n+1}^N||^2) \ . \tag{4.4.22}$$

(The last equality is obtained from

$$||\hat{x}||^2 = ||\hat{x}-y_n^N||^2 + ||y_n^N||^2 \tag{4.4.23}$$

which is a consequence of the orthogonality conditions for linear estimation). Rearranging (4.4.22), we obtain (4.4.12).■

The main conclusion from Theorem 4.4.3 is that the estimation error decreases geometrically, at a rate which is independent of the vector being estimated, but which depends on the subspaces $H^1,\ldots,H^N$. In particular, the constant α depends on the angles between these subspaces. Accordingly, the convergence rate of the algorithm may vary significantly when different decomposition of the space H are tried. Apparently, α is larger when the subspaces $H^1,\ldots,H^N$ are nearly orthogonal, in which case α behaves like 1/N, as N changes. This suggests that more rounds of computations are needed, in general, if a finer decomposition is used. On the other hand, finer decompositions require fewer computations per round, because smaller covariance matrices have to be inverted. Theorem 4.4.3 is not adequate to resolve this tradeoff, primarily because inequalities (4.4.11), (4.4.12) are not particularly tight.

Theorem 4.4.3 suggests that the ring protocol can lead to a much slower algorithm than the star protocol. Numerical experimentation, however, did not reveal any such effect. This may be explained by observing that the proof of part (b) of the Theorem utilizes bounds which are likely to be much less tight than those in part (a), most notably, the inequality $\|\sum_{k=1}^{i} \tilde{y}_{n+1}^k\|^2 \leq i \sum_{k=1}^{i} \|\tilde{y}_{n+1}^k\|^2$.

We continue this Section with a discussion of the implementation of the decomposition algorithm. First, it may be used by a single processor (centralized computation), as an alternative to a standard algorithm for inverting the covariance matrix. Or, it

may be implemented in a decentralized manner, whereby each block of data corresponds to a physically distinct processor. Note that, for any i, n, we have $\hat{x}_n^i = a_n^i y$, where $\hat{x}_n^i$ is the estimate of the i-th processor at time n, y is the vector of all available observations and a_n^i is a row vector. When processor j receives $\hat{x}_n^i$, it must also know a_n^i, in order to be able to extract information from $\hat{x}_n^i$. There are two choices:

a) Processor j computes a_n^i, possibly off-line.

b) Processor i transmits a_n^i to processor j.

Which of the two should be done depends on whether communications or computations are more costly. Whether any one of the above two variations can be useful depends on the particularities of the actual situation and its inherent communication and computation limitations. More numerical experience is needed before a definite answer can be given.

Numerical Results

Let x be an unknown scalar, zero mean, random variable to be estimated ($E[x^2]=5.$) Let $y_i = x + w_i$, (i=1,...,18) be the observations. The noises w_i are assumed to be independent of x.† We split the 18-dimensional observation vector into blocks of data (corresponding to distinct processors) and used the decomposition algorithm of Theorem 4.4.1. We employed the ring protocol and assumed that at each stage a processor only knows its own observations and the most recent message it received (Assumptions 4.2.5, 4.2.6).

† The covariance matrix Σ_w of the vector of noises was randomly generated.

Let M_i be the number of observations assigned to processor i. We considered two alternative decompositions: (i) N=2, M_1=10, M_2=8; (ii) N=6, $M_1=\ldots=M_6=3$. We first executed the algorithm using the covariance Σ_w and, then, once more using the covariance Σ_w+I.

The results are presented in Figures 4.4.1, 4.4.2. The horizontal axis denotes stages (each stage corresponds to an update by some processor) and the vertical axis indicates the associated mean square error. The dotted horizontal line indicates the centralized mean square error. The curves D1 and D2 corresponds to the first and second decomposition, respectively. As expected, convergence was much faster when the identity was added to the initial covariance; moreover the first decomposition converged much faster than the second.

To illustrate the merits of the decomposition algorithm we performed a rough count of operations. We only took matrix inversions into account, assuming that the inversion of a MxM matrix requires M^3 operations, which is accurate enough for our purposes. With this counting scheme, the centralized algorithm required 5832 operations. The points A,B in the graphs were reached after 4100, 1152 operations, respectively. This leads to the following conclusion: While the first decomposition needs very few stages to converge, it does not have any particular computational advantages. The second decomposition, however, leads to an estimate close to the optimal with much fewer operations than the centralized algorithm.

It is fair to say, however, that the decomposition algorithm we have studied above should be compared to other decomposition algorithms for solving linear equations, rather than algorithms requiring $O(N^3)$ operations.

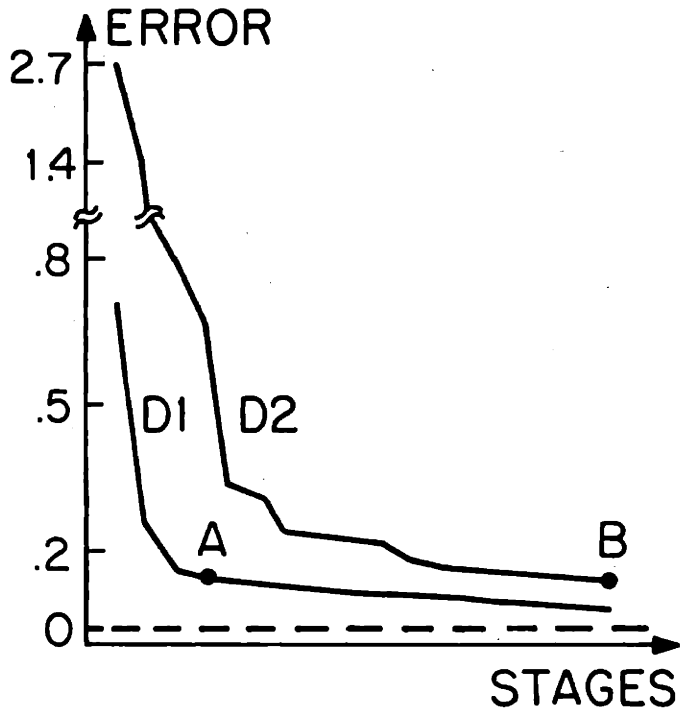

Figure 4.4.1: Mean Square Errors of Decomposition Algorithms, for Covariance Σ_w; Dashed Line Indicates the Optimal Mean Square Error.

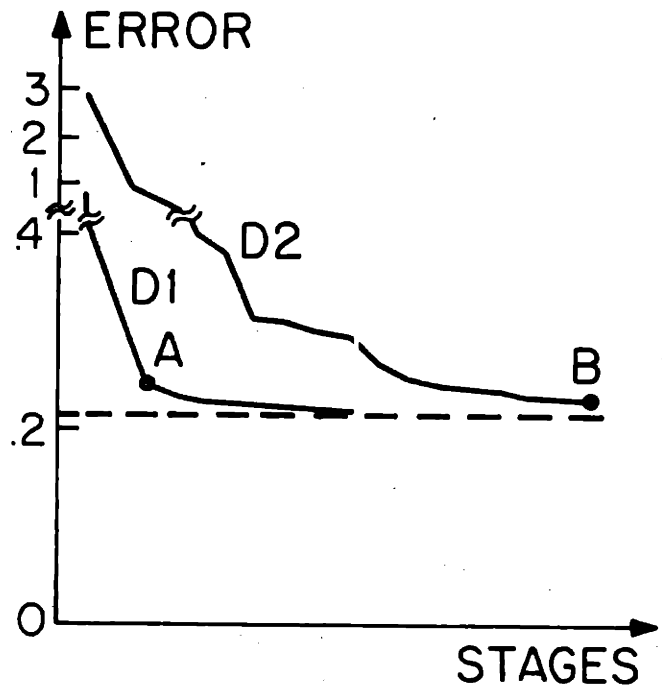

Figure 4.4.2: Mean Square Errors of Decomposition Algorithms, for Covariance $\Sigma_W + I$; Dashed Line Indicates the Optimal Mean Square Error.

4.5 A MODEL INVOLVING A COORDINATOR

In the previous sections we had assumed that for any pair of processors i,j, processor i is allowed to communicate to j. In this section we assume that a particular processor (denoted by the superscript o) has special status and acts like a coordinator. The scheme we envisage is the following: At each instance of time n, processor i evaluates u_n^i which it communicates to the coordinator. The coordinator then combines u_n^1 to u_n^N to produce a tentative decision u_n^o. We assume that the coordinator has no data of its own). It then transmits u_n^o to all other processors which accordingly update their decisions. Were the coordinator to combine u_n^1 to u_n^N "optimally", the above scheme would reduce to the one of the previous sections and our past results would apply. We assume, however, that the coordinator simply sets.

$$u_n^o = \sum_{i=1}^{N} a^i u_n^i$$

where the coefficients a^i are deterministic, positive and $\sum_{i=1}^{N} a^i = 1$. The implicit behavioral assumptions are: (i) The coordinator has no memory and (ii) it need not have a good knowledge of the problem. It only knows how much it can rely on each of the other processors; this is reflected by its choice of the coefficients a^i which may be thought of as a "reliability index" for processor i in the eyes of the coordinator. We then obtain:

Theorem 4.5.1: The conclusions of Theorem 4.3.1, 4.3.4, 4.4.1, remain true (under their respective assumptions) with the scheme introduced in this section.

<u>Proof</u>: Since u_n^o is F_{n+1}^i-measurable, it follows that

$$E[c(u_{n+1}^i)] \leq E[c(u_n^o)] = E[c(\sum_{j=1}^N a^j u_n^j)] \leq \sum_{j=1}^N a^j E[c(u_n^j)] \quad . \tag{4.5.1}$$

Also, since u_n^i is F_{n+1}^i-measurable, $E[c(u_n^i)]$ decreases and converges to some g^i. Taking the limit in (4.5.1) we conclude that $g^i = g^j$, $\forall i,j$. From this point on, the proofs of Theorems 4.3.1, 4.3.4, 4.4.1 (with minor modifications) are valid and establish the desired conclusions.■

The above scheme can be viewed as a framework for cooperation, where the coordinator simply aids the processors; or, for LQG problems, as another decomposition algorithm. It can be also interpreted, however, from an entirely different point of view: Suppose that the processors are selfish and independent individuals, faced with identical situations, possessing different information and having to make repetitive decisions. They can certainly benefit by observing past decisions of the other processors but assume that this is not possible. They are able, however, to observe a weighted average u_n^o of all decisions made in the last stage, which they take into account for their future actions. The motivation for such a model comes primarily from economics: Each processor is a buyer (or seller) in the same market and at each stage he obtains some aggregate information (e.g. the average price) on the transactions that were made in the previous stage. In this sense the "coordinator" simply represents a market mechanism. Our results state that, eventually, an "informational equilibirium" will be reached. Such an equilibrium has been studied by Radner [1979] in a different setting. However, there was no demonstration of an adjustment process that could lead to such an equilibrium. Our scheme provides a model of rational behavior which, if followed by each agent, leads to equilibrium.

Moreover, within such a context (of selfish individuals confronted with identical situations) and for LQG problems with perfect memory, optimal tentative decisions constitute a set of strategies in Nash equilibrium for a certain game. (This is why optimal tentative decisions can be called "a model of rational behavior"). Let us define the game of interest more precisely.

Let y_n^i be a vector of jointly Gaussian random variables that generate G_n^i, the σ-algebra of events known to processor i at time n if it had received no messages. At each stage, processor i selects a decision u_n^i and incurs (but does not observe) a cost $\alpha^n c(u_n^i)$, where $0<\alpha<1$ and c is a quadratic cost function. Then $u_n^o = \sum a^i u_n^i$ is formed and communicated to all processors. The total cost to processor i is $J^i = \sum_{n=1}^{\infty} \alpha^n E[c(u_n^i)]$. A strategy Γ^i for processor i is a sequence $\{\gamma_n^i, i=1,2,\ldots\}$ of measurable functions such that $u_n^i = \gamma_n^i(y_n^i, u_1^o, \ldots, u_{n-1}^o)$. A set $\{\Gamma^1, \ldots \Gamma^N\}$ of strategies is said to be in Nash equilibrium if

$$J^i(\Gamma^1, \ldots, \Gamma^{i-1}, \hat{\Gamma}^i, \Gamma^{i+1}, \ldots, \Gamma^N) \geq J^i(\Gamma^1, \ldots, \Gamma^i, \ldots, \Gamma^N)$$

for any strategy $\hat{\Gamma}^i$, for any i. Let $\Gamma=\{\Gamma^i: i=1,\ldots,N\}$ denote the particular set of strategies where each processor at each stage plays its optimal tentative decision (Note that these are linear strategies). Then,

<u>Theorem 4.5.2</u>: Γ is a set of strategies in Nash equilibrium.

<u>Proof</u>: Let u_n^o, u_n^i be the coordinator messages and decisions, respectively, when all processors use Γ^i. Let F_n^i be the smallest σ-algebra generated by y_n^i and $u_1^o, \ldots, u_{n-1}^o$.

Now suppose that a particular processor, say processor 1, uses a strategy $\hat{\Gamma}^1$ different from Γ^1, while all other processors use Γ^i. Let $\hat{u}_n^o$, $\hat{u}_n^i$ be the coordinator messages and decisions resulting from the application of this new set of strategies.

We proceed by induction. Clearly, $\hat{u}_1^1$ is F_1^1-measurable. Assume that $\hat{u}_n^1$ and $\hat{u}_1^o,\ldots,\hat{u}_{n-1}^o$ are F_n^1-measurable. Then, because of the linearity of the Γ^i's, $u_n^i - \hat{u}_n^i$, $(i\neq 1)$, is linear in $u_1^o - \hat{u}_1^o,\ldots,u_{n-1}^o - \hat{u}_{n-1}^o$ and hence F_n^1-measurable. Since u_n^o is F_{n+1}^1-measurable and $F_n^1 \subset F_{n+1}^1$ (perfect memory), $\hat{u}_n^o$ is F_{n+1}^1-measurable, and so is $\hat{u}_{n+1}^1$. The induction shows that this is true for all n.

Therefore, by the minimizing property of u_n^i, $E[c(u_n^i)] \leq E[c(\hat{u}_n^i)]$, $\forall n$ and this completes the proof.■

4.6 PROCESSORS WITH DIFFERENT MODELS

The results of the preceding sections rest on the crucial assumption that all processors have identical models of the situation; that is, they all employ the same probability measure in their calculations. In this Section we discuss what could happen if this assumption is violated. The motivation comes from the case in which the processors are human decision makers. Then, it is very likely that their models will have some differences, even if they are "experts," on the problem facing them.

It is important to realize that there can be no unique normative answer on what should the processors do in the face of model differences. Rather, we may assume some general rules of behavior and then proceed to study the consequences of such assumptions. We sketch here a few possible lines of approach, leaving more detailed analysis for future research.

Note that two processors with different models will make, in general, different decisions, even if they have the same information. This is the same as what would happen if they had identical models but different cost functions. (In fact, the distinction between model and cost function cannot be made perfectly sharp). So, differences in models may lead to situations best modelled by game theory. Moreover, since the processors have to reach consensus, we would effectively obtain a cooperative (bargaining) game, consensus being reached somewhere on an appropriate negotiation set. This game theoretic approach can have some drawbacks, mainly because game theory requires fairly strong "rationality" assumptions. For example, it might be necessary to assume that

a) The models of each processor are common knowledge, or

b) The models of each processor are drawn randomly from a set of possible models and the probability distribution associated with this random selection is common knowledge. (This latter assumption is similar to those introduced by Harsanyi [1967; 1968a; 1968b] to address non-cooperative games with different models. However, very little can be said, in general, under such an assumption, mainly because a probability distribution over a set of models may be hard to handle).

An additional objection to game theoretic approaches can be the following: If two decision makers - with different models - want to cooperate and reach consensus, they should not bargain on an appropriate decision, but they should first try to reconcile their models.

A different situation is obtained if modelling differences exist but no processor knows that this is the case. So, suppose that two processors employ the scheme of

the preceding sections. Each one has a different model, but each one believes that they have the same model. What happens then is that each processor interprets incorrectly the messages it receives. Let us elaborate on the above statement.

Suppose, for example, that processor 1 computes a decision u^1 based on its own model. Its decision is given by $u^1=f^1(\omega)$, for some function $f^1: \Omega \to U$. Clearly, f^1 depends on the cost function being minimized as well as the prior probabilities (model) assumed by processor 1. Processor 2, however, believes that processor 1 uses a different set of prior probabilities in its calculations. So, processor 2 believes that $u^1=f^2(\omega)$, for some other function $f^2: \Omega \to U$. When processor 1 receives a message with the value of u^1, it deduces that the event $(f^2)^{-1}(u^1)$ has occured, whereas in fact event $(f^1)^{-1}(u^1)$ has occured. (Here, the superscript -1 stands for the inverse image of a function). In other words, messages code information; by not knowing the coding rule used by processor 1, processor 2 decodes messages incorrectly.

Suppose, however, that $(f^2)^{-1}(u^1)=\phi$. This means, in the eyes of processor 2: "if processor 1 had the same model with me, it would not send me such a message; therefore, it has a different model". Once the processors (or a subset of them) realize that they do not share a common model, we are back to the game-like situation discussed earlier in this section. Concerning awareness of modelling differences, there are three possibilities:

a) Some processors find out about modelling differences after the scheme has operated for finite time only.

b) No one finds out in finite time, but they may find out asymptotically. (This can be made more precise).

c) Modelling differences are not detected, not even asymptotically.

We may then pose questions of the following type:

(i) What are some conditions for cases (a),(b) or (c) above to occur?

(ii) For cases (a) and (b) what can be said about disagreement at the time that modelling differences are detected?

(iii) For case (c), the processors must asymptotically agree. What is it they agree upon?

(iv) Suppose that in addition to running the scheme of this chapter, each processor performs a (sequential) hypothesis test, to test for modelling differences. When will be modelling differences detected?

We have no results on the above questions which are topics for future research. We can draw the conclusion, however, that modelling differences may lead to a variety of qualitatively different situations. Exploring such situations may lead to some understanding of decision making in the face of modelling differences which is an area of great practical importance.

Remark: A recent paper by Teneketzis and Varaiya [1984] shows that in the case of an estimation problem on a finite probability space and under perfect memory, case (b) cannot occur and if case (c) occurs then the processors agree in finite time. In fact, it is easy to show (using Theorem 4.3.3) that this result remains valid under imperfect memory and for an arbitrary cost function.

4.7 CONCLUSIONS

A set of processors with the same objective who start communicating to each other their tentative optimal decisions are guaranteed to agree in the limit. Under certain assumptions, this is true even if the messages are received in the presence of noise and even if the memory of the processors is limited and they are allowed to forget some of their past knowledge. Moreover, they are guaranteed to converge to the centralized optimal decision for linear estimation problems, provided that no processor forgets its own raw observations. This corresponds to a geometrically convergent decomposition algorithm for static linear estimation problems, on which some numerical results are reported.

Similar results are obtained if the processors do not communicate directly but receive messages from a coordinator who evaluates a weighted average of all tentative decisions. In the latter framework, for linear estimation problems and with perfect memory, optimal tentative decisions correspond to Nash strategies for a certain sequential game and admit an economic interpretation.

These results are valid when all processors share the same model (identical prior probabilities). The characterization of the behavior of processors with different models is an unexplored problem.

Remark: In a recent paper, Washburn and Teneketzis [1984] have studied similar schemes from a more abstract point of view. Their approach is limited, however, to the special case in which all processors have perfect memory.

CHAPTER 5: DECENTRALIZED DETERMINISTIC AND STOCHASTIC ITERATIVE ALGORITHMS

5.1 INTRODUCTION AND OVERVIEW

From a conceptual point of view, this Chapter may be viewed as a continuation of Chapter 4. We consider again decentralized schemes in which a set of processors make (tentative) decisions, perform computations or obtain observations and exchange relevant messages, with the end-goal of minimizing a certain cost function. In the schemes of Chapter 4, each processor was assumed to be as "rational" as possible: everything was imbedded in a Bayesian framework and processors were making tentative decisions which were optimal, given their information. The main practical difficulty with such schemes is that, at each stage, each processor may face an optimization problem which is difficult by itself; the only exception is the LQG problem of Section 4.4. For this reason, it may be more useful to assume that processors do not update in an "optimal" (Bayesian) way, but rather that they update by moving a little bit in a direction of improvement. In other words, we are interested in gradient-like (or descent) iterative schemes. Although a descent assumption could be considered to be an arbitrary restriction it should be stressed that such an assumption underlies most centralized iterative schemes for deterministic optimization, recursive identification, stochastic approximation, random search algorithms, adaptation and training algorithms. [Poljak and Tsypkin, 1973; Ljung, 1977a]. Given the multitude of application areas for centralized descent algorithms, it should not be surprising that their decentralized counterparts may be useful in a variety of contexts as well. In fact, we discuss in later sections applications in parallel computing systems, distributed signal processing (identification), decision making in large organizations and data communication networks.

Let us now elaborate on what is a genuine "decentralized algorithm." Given any centralized algorithm, it is often straightforward to design a decentralized (parallel) implementation, by employing a set of perfectly synchronized processors. Such a decentralized algorithm is mathematically identical to the original centralized one and no new analysis is required. Synchronous algorithms may have, however, certain drawbacks: a) Synchronism may be hard to enforce, or its enforcement may introduce substantial overhead. b) Communication delays may introduce bottlenecks to the speed of the algorithm (the time required for one stage of the algorithm will be constrained by the slowest communication channel). c) Synchronous algorithms may require far more communications than are actually necessary. d) Even if all processors are equally powerful some will perform certain computations faster than others, due solely to the fact that they operate on different inputs. This in turn, may lead to having many processors idle for a large proportion of time. For these reasons, we choose to study asynchronous decentralized iterative optimization algorithms in which each processor does not need to communicate to each other processor at each time instance; also, processors may keep performing computations(or, in a decision making context, update their decisions) without having to wait until they receive the messages that have been transmitted to them; processors are allowed to remain idle some of the time; finally, some processors may perform computations faster than others. Such schemes can alleviate communication overloads and they are not excessively slowed down by neither communication delays, nor

by differences in the time it takes processors to perform one computation. (A similar discussion of the merits of asynchronous algorithms is provided by H.T. Kung, [1976].)

We now outline the contents of this Chapter. In Section 5.2, we present a model for decentralized computation which is a general description of the structure of most algorithms to be studied in this Chapter. This model allows communications between processors to be infrequent and fairly chaotic, as well as communication delays. An interesting and useful feature of this model is that a global (aggregate) state of computation may be associated (in a non-trivial way) with a decentralized algorithm.

In Section 5.3 we prove convergence of decentralized gradient-like stochastic algorithms. We consider constant step-size algorithms (deterministic gradient methods being a special case), as well as decreasing step-size algorithms. For the latter case, we show that convergence is obtained even if the time between consecutive communications goes to infinity, as the algorithm progresses.

In Section 5.4 we consider stochastic algorithms with correlated noise and prove convergence under assumptions similar to those that are employed to show convergence of centralized algorithms via the ODE approach.

In Section 5.5 we present some applications of our results in decentralized system identification.

In Section 5.6 we consider a decentralized (deterministic) gradient algorithm. This being a particularly simple algorithm, we are able to study in more detail the effect of the various parameters describing the structure of the

communication process. In Section 5.7 we indicate that these results may form the basis of a new approach to organizational design problems. Then, in Section 5.8 we discuss a few more potential applications of our results.

In Section 5.9 we outline some topics for further research; finally, Section 5.10 contains a summary and some general conclusions.

A simpler version of Sections 5.2, 5.3 appears in [Tsitsiklis, Bertsekas and Athans, 1984]. Also, a major part of this Chapter is discussed in [Bertsekas, Tsitsiklis and Athans, 1984].

5.2 A MODEL OF DECENTRALIZED COMPUTATION

We present here the model of decentralized computation employed in this chapter. We also define the notation and conventions to be followed. Related models of decentralized computation have been used by Chazan and Miranker [1969], Baudet [1978] and Bertsekas [1982,1983] where each processor specialized in updating a different component of some vector. The model developed here is more general, in that it allows different processors to update the same component of some vector. If their individual updates are different, their disagreement is (asymptotically) eliminated through a process of communicating and combining their individual updates. In such a case, we will say that there is overlap between processors. Overlapping processors are probably not very useful in the context of deterministic algorithms, unless redundancy is intended to provide a certain degree of fault tolerance. For stochastic algorithms, however, overlap essentially corresponds to having different processors obtain noisy measurements of the same unknown quantity and effectively increases the "signal-to-noise ratio."

Let $H_1, H_2, \ldots, H_L$ be Banach spaces† and let $H = H_1 x H_2 x \ldots x H_L$. If $x = (x_1, x_2, \ldots, x_L)$, $x_\ell \in H_\ell$, we will refer to x_ℓ as the ℓ-th component of x. We endow H with the norm

$$||(x_1, x_2, \ldots, x_L)|| = \max_\ell ||x_\ell|| \tag{5.2.1}$$

Let $\{1, \ldots, M\}$ be the set of processors that participate in the distributed computation. As a general rule concerning notation, we use subscripts to indicate a component of an element of H, superscripts to indicate an associated processor; we indicate time by an argument that follows.

The algorithms to be considered evolve in discrete time. Even if a distributed algorithm is asynchronous and communication delays are real (i.e., not integer) variables, the events of interest (an update by some processor, transmission or reception of a message) may be indexed by a discrete variable; so, the restriction to discrete time entails no loss of generality.

It is important here to draw a distinction between "global" and "local" time. The time variable we have just referred to corresponds to a global clock. Such a global clock is needed only for analysis purposes: it is the clock of an analyst who watches the operation of the system. On the other hand the processors may be working without having access to a global clock. They may have access to a local clock or to no clock at all. We will see later that our results, based on the existence of a global clock, may be used in a straightforward way to prove convergence of algorithms implemented on the basis of local clocks only.

We assume that each processor has a buffer in its memory in which it keeps some element of H. The value stored by the i-th processor at time n is denoted

† In most situations of interest, each H_i will turn out to be finite dimensional. However, the generalization to Banach spaces does not introduce any new difficulties in our analysis.

by $x^i(n)$. At time n, each processor may receive some exogenous measurements and/or perform some computations. This allows it to compute a "step" $s^i(n) \in H$, to be used in evaluating the new vector $x^i(n+1)$. Besides their own measurements and computations, processors may also receive messages from other processors, which will be taken into account in evaluating their next vector. The process of communications is assumed to be as follows:

At any time n, processor i may transmit some (possibly all) of the components of $x^i(n)$ to some (possibly all or none) of the other processors. (In a physical implementation, messages do not need to go directly from their origin to their destination; they may go through some intermediate nodes. Of course, this does not change the mathematical model presented here.) We allow for the time being, arbitrary communication delays. For convenience, we also assume that for any pair (i,j) of processors, for any component x_ℓ and any time n, processor i may receive at most one message originating from processor j and containing an element of H_ℓ. This leads to no significant loss of generality: for example, a processor that receives two messages simultaneously could keep only the one which was most recently sent; if messages do not carry timestamps, there could be some other arbitration mechanism. Physically, of course, simultaneous receptions are impossible; so, a processor may always identify and keep the most recently received message, even if all messages arrived at the same discrete time n.

If a message from processor j, containing an element of H_ℓ is received by processor i ($i \neq j$) at time n, let $t_\ell^{ij}(n)$ denote the time that this message was sent. Therefore, the content of such a message is precisely $x_\ell^j(t_\ell^{ij}(n))$.

Naturally, we assume that $t_\ell^{ij}(n) \leq n$. For notational convenience, we also let $t_\ell^{ii}(n)=n$, for all i, ℓ, n. We will be assuming that the algorithm starts at time 1; accordingly, we assume that $t_\ell^{ij}(n) \geq 1$. Finally, we denote by T_ℓ^{ij} the set of all times that processor i receives a message from processor j, containing an element of H_ℓ. To simplify matters we will assume that, for any i,j,ℓ, the set T_ℓ^{ij} is either empty or infinite.

Once processor i has received the messages arriving at time n and has also evaluated $s^i(n)$, it evaluates its new vector $x^i(n+1) \in H$ by forming (componentwise) a convex combination of its old vector and the values in the messages it has just received, as follows:

$$x_\ell^i(n+1) = \sum_{j=1}^{M} a_\ell^{ij}(n) x_\ell^j(t_\ell^{ij}(n)) + \gamma^i(n) s_\ell^i(n), \quad n \geq 1, \tag{5.2.2}$$

where $s_\ell^i(n)$ is the ℓ-th component of $s^i(n)$ and the coefficients $a_\ell^{ij}(n)$ are scalars satisfying:

$$\text{(i)} \qquad a_\ell^{ij}(n) \geq 0, \qquad \forall i,j,\ell,n, \tag{5.2.3}$$

$$\text{(ii)} \qquad \sum_{j=1}^{M} a_\ell^{ij}(n) = 1, \qquad \forall i,\ell,n, \tag{5.2.4}$$

$$\text{(iii)} \qquad a_\ell^{ij}(n) = 0, \qquad \forall n \notin T_\ell^{ij},\ i \neq j \tag{5.2.5}$$

Remarks:

1. Note that $t_\ell^{ij}(n)$ has been defined only for those times n that processor i receives a message of a particular type, i.e. for $n \in T_\ell^{ij}$. However, whenever

$t_\ell^{ij}(n)$ is undefined, we have assumed above that $a_\ell^{ij}(n)=0$, so that equation (5.2.2) has an unambiguous meaning.

2. When we refer to a processor performing a "<u>computation</u>", we mean the evaluation and addition of the term $\gamma^i(n)s_\ell^i(n)$ in (5.2.2). With this terminology, forming the convex combination in (5.2.2) is not called a computation. We denote by T_ℓ^i the set of all times that processor i performs a computation involving the ℓ-th component. Whenever $n \notin T_\ell^i$, it is understood that $s_\ell^i(n)$ in (5.2.2) equals zero. We assume again that for any i,ℓ the set T_ℓ^i is either infinite or empty. Accordingly, processor i will be called <u>computing</u>, or <u>non-computing</u>, <u>for component ℓ</u>.

3. The quantities $\gamma^i(n)$ in (5.2.2) are nonnegative scalar step-sizes. These step-sizes may be constant (e.g. $\gamma^i(n)=\gamma_o$, $\forall n$), or time-varying, e.g. $\gamma^i(n)=1/t_n^i$, where t_n^i is the number of times that processor i has performed a computation up to time n. Notice that with such a choice each processor may evaluate its step-size using only a local counter rather than a global clock.

4. The envisaged sequence of events underlying (5.2.2) at any time n, is as follows: processor i

   (i) Transmits $x_\ell^i(n)$ to other processors.

   (ii) Receives messages $x_\ell^j(t_\ell^{ij}(n))$ from other processors.

   (iii) Computes $s_\ell^i(n)$.

   (iv) Evaluates $x_\ell^i(n+1)$.

5. Note that the combining of the vectors of different processors, in equation (5.2.2), is done componentwise. Consequently, we can argue inductively that $x_\ell^i(n+1)$ is only a function of $\{s_\ell^j(k): j\in\{1,\ldots,M\}, k\in\{1,\ldots,n\}\}$ and $\{x_\ell^j(1): j\in\{1,\ldots,M\}\}$.

Examples

We now introduce a collection of simple examples representing various classes of algorithms we are interested in, so as to illustrate the nature of the assumptions to be introduced later. We actually start with a broad classification and then proceed to more special cases. In these examples, we model the message receptions and transmissions (i.e. the sets T_ℓ^{ij} and the variables $t_\ell^{ij}(n)$), the times at which computations are performed (i.e. the sets T_ℓ^i) and the combining coefficients $a_\ell^{ij}(n)$ as deterministic. (This does not mean, however, that they have to be a priori known by the processors). We will see in Section 5.3 that this assumption may be relaxed.

Specialization: This is the case considered by Bertsekas [1982, 1983], where each processor updates a particular component of the x vector specifically assigned to it and relies on messages from the other processors for the remaining components. Formally:

(i) M=L. (There are as many processors as there are components).

(ii) $s_\ell^i(n)=0$, $\forall \ell\neq i$, $\forall n$. (A processor may update only its own component; $T_\ell^i=\phi, \forall i\neq\ell$).

(iii) Processor j only sends messages containing elements of H_j; if processor i receives such a message, it uses it to update x_j^i by setting x_j^i equal to the value received. Equivalently,

a) If $i \neq j$ and $j \neq \ell$, then $T_\ell^{ij} = \phi$ and $a_\ell^{ij}(n) = 0$, $\forall n$.

b) If processor i receives a message from processor j at time n, i.e. if $n \in T_j^{ij}$, then $a_j^{ij}(n) = 1$. Otherwise, $a_j^{ij}(n) = 0$, and $a_j^{ii}(n) = 1$.

Overlap: This is the other extreme, at which L=1 (we do not distinguish components of elements of H), messages contain elements of H (not just components) and each processor may update any component of x. (For this case subscripts are redundant and will be omitted.

We now let H be finite-dimensional and assume that $J: H \to [0,\infty)$ is a continuously differentiable nonnegative function with a Lipschitz continuous derivative.

Example I: Deterministic Gradient Algorithm; Specialization. Let $\gamma^i(n) = \gamma_o > 0$, $\forall n, i$. At each time $n \in T_i^i$ that processor i updates x_i^i, it computes $s_i^i(n) = -\frac{\partial J}{\partial x_i}(x^i(n))$ and lets $s_j^i(n) = 0$, for $j \neq i$. We assume that each processor i communicates its component x_i^i to every other processor at least once every B_1 time units, for some constant B_1. Other than this restriction, we allow the transmission and reception times to be arbitrary. (A related stochastic algorithm could be obtained by letting $s_i^i(n) = -\frac{\partial J}{\partial x_i}(x^i(n))(1+w_i^i(n))$, where $w_i^i(n)$ is unit variance white noise, independent for different i's).

Example II: Newton's Method; Overlap. For simplicity we assume that there are only two processors (M=2). Let $\gamma^i(n) = \gamma_o > 0$, $\forall n$. We also assume that J is twice continuously differentiable, strictly convex and its Hessian matrix, denoted

by $G(x)$, satisfies $0<\delta_1 I \leq G(x) \leq \delta_2 I, \forall x \in H$. At each time $n \in T^i$, processor i computes $s^i(n) = -G^{-1}(x^i(n)) \frac{\partial J}{\partial x}(x^i(n))$. If at time n processor 1 (respectively, 2) receives a message $x^2(t^{12}(n))$ (resp. $x^1(t^{21}(n))$), it updates its state vector by $x^1(n+1) = a_{11}x^1(n) + a_{12}x^2(t^{12}(n))$, (resp. $x^2(n+1) = a_{21}x^1(t^{21}(n)) + a_{22}x^2(n)$). Here we assume that $0<a_{ij}<1$ and that $a_{11}+a_{12} = a_{21}+a_{22} = 1$. We make the same assumptions on transmission and reception times as in Example I.

Example III: Distributed Stochastic Approximation; Specialization. Let $\gamma^i(n)$ be such that, for some positive constants A_1, A_2, $A_1/n \leq \gamma^i(n) \leq A_2/n$, $\forall n$. Notice that the implementation of such a stepsize only requires a local clock that runs in the same time scale (i.e. within a constant factor) as the global clock. For $n \in T_i^i$, let $s_i^i(n) = -\frac{\partial J}{\partial x_i}(x^i(n)) + w_i^i(n)$. Also, $s_j^i(n)=0$, for $i \neq j$ and for all n. We assume that $w_i^i(n)$, conditioned on the past history of the algorithm has zero mean and that $E[||w_i^i(n)||^2 | x^i(n)] \leq K(J(x^i(n))+1)$, for some constant K. We assume that for some $B_1 \geq 0$, $\beta \geq 1$ and for all n, each processor communicates its component x_i^i to every other processor at least once during the time interval $[B_1 n^\beta, B_1(n+1)^\beta]$. Other than the above restriction, we allow transmission and reception times to be arbitrary. Notice that the above assumptions allow the time between consecutive communications to grow without bound.

Example IV: Distributed Stochastic Approximation; Overlap. Let $\gamma^i(n)$ be as in Example III and let M=2. For $n \in T^i$, let $s^i(n) = -\frac{\partial J}{\partial x}(x^i(n)) + w^i(n)$, where $w^i(n)$ is as in Example III. We make the same assumptions on transmission and reception times as in Example III. Whenever a message is received, a processor combines its vector with the content of that message using the combining rules of Example II.

Example V: This example is rather academic but will serve to illustrate some of the ideas to be introduced later. Consider the case of overlap, assume that H is one-dimensional, and let $\gamma^i(n)=1$, $\forall n$. Assume that, at each time n, either all processors communicate to each other, or no processor sends any message. Let the communication delays be zero (so, $t^{ij}(n)=n$, whenever $t^{ij}(n)$ is defined) and assume that $a^{ij}(n) = a^{ij}$ (constant) at those times n that messages are exchanged. We define vectors $x(n) = (x^1(n),\ldots,x^M(n))$ and $s(n) = (s^1(n),\ldots,s^M(n))$. Then, the algorithm (5.2.2) may be written as

$$x(n+1) = A(n)x(n) + s(n) \quad . \tag{5.2.6}$$

For each time n, either $A(n)=I$ (no communications) or $A(n)=A$, the matrix consisting of the coefficients a^{ij}. The latter is a "stochastic" matrix: it has nonnegative entries and each row sums to 1. We assume that a^{ij} is positive. It follows that $\bar{A} = \lim_{n\to\infty} A^n$ exists and has identical rows with positive elements. We assume that the time between consecutive communications is bounded but otherwise arbitrary. Clearly then, $\lim_{n\to\infty} \prod_{m=k}^{n} A(m)=\bar{A}$, for all k. This example corresponds to a set of processors who individually solve the same problem and, from time to time, simultaneously exchange their partial results. It is interesting to compare equation (5.2.6) with the generic equation

$$x(n+1) = x(n) + \gamma(n)s(n)$$

which arises in centralized algorithms.

Assumptions on the communications and the combining coefficients

We now consider a set of assumptions on the nature of the communications and combining process, so that the preceding examples appear as special cases. We

start with a relatively simple set of assumptions which are very easy to enforce, examine their consequences and finally suggest a more general version.

For each component $\ell \in \{1,\ldots L\}$ we introduce a directed graph $G_\ell=(V,E_\ell)$ with nodes $V=\{1,\ldots,M\}$ corresponding to the set of processors. An edge (j,i) belongs to E_ℓ if and only if T_ℓ^{ij} is infinite, that is, iff processor j sends an infinite number of messages to processor i with a value of the ℓ-th component x_ℓ^j.

<u>Assumption 5.2.1</u>: For each component $\ell \in \{1,\ldots,L\}$, the following hold:

a) There is at least one computing processor for component ℓ.

b) There is a directed path in G_ℓ, from every computing processor (for component ℓ) to every other processor (computing or not).

c) There is some $\alpha>0$ such that:

(i) If processor i receives a message from processor j at time n (i.e. if $n \in T_\ell^{ij}$), then $a_\ell^{ij}(n) \geq \alpha$.

(ii) For every computing processor i, $a_\ell^{ii}(n) \geq \alpha$, $\forall n$.

(iii) If processor i has in-degree (in G_ℓ) larger or equal than 2, then $a_\ell^{ii}(n) \geq \alpha$, $\forall n$.

Let us pause to indicate the intuitive content of part (c) of Assumption 5.2.1. Part (i) states that a processor should not ignore the messages it receives. Part (ii) requires the past updates of any computing processor to have a lasting effect on its state of computation. Finally, part (iii) implies that if processor i receives messages from two processor (say i_1,i_2), it does not forget the effects of messages of processor i_1 upon reception of a message from processor i_2. These

conditions, together with part (b) of the Assumption, guarantee that any update by any computing processor has a lasting effect on the states of computation of all other processors.

Assumption 5.2.2: The time between consecutive transmissions of component x_ℓ^j from processor j to processor i is bounded by some $B_1 \geq 0$, for all $(j,i) \in E_\ell$.

Assumption 5.2.3: There are constants $B_1 \geq 0$, $\beta \geq 1$ such that, for any $(j,i) \in E_\ell$, and for any n, at least one message x_ℓ^j is sent from processor j to processor i during the time interval $[B_1 n^\beta, B_1 (n+1)^\beta]$. Moreover, the total number of messages transmitted and/or received during any such interval is bounded.

Assumption 5.2.4: Communication delays are bounded by some $B_o \geq 0$.

Note that Assumption 5.2.2 is a special case of 5.2.3, with $\beta=1$. Assumption 5.2.1 holds for all the examples introduced above. Assumption 5.2.2 holds for Examples I,II,V; Assumption 5.2.3 holds for Examples III,IV, except for its last part which has to be explicitly introduced. Assumption 5.2.4 also needs to be explicitly introduced.

We now investigate the consequences of Assumptions 5.2.1-5.2.4. Note that equation (5.2.2) which defines the algorithm is a linear system. In particular, it is easy to see that, for any i,n,ℓ, the variable $x_\ell^i(n+1)$ is a linear combination of the initial conditions $\{x_\ell^j(1): j=1,\ldots,M\}$ and the "inputs" $\{\gamma^j(k)s_\ell^j(k): k=1,\ldots,n;\ j=1,\ldots,M\}$. Therefore, there exist scalars $\Phi_\ell^{ij}(n|k)$, for $n \geq k$, such that

$$x_\ell^i(n) = \sum_{j=1}^{M} \Phi_\ell^{ij}(n|0)x_\ell^j(1) + \sum_{k=1}^{n-1} \sum_{j=1}^{M} \gamma^j(k)\Phi_\ell^{ij}(n|k)s_\ell^j(k) \quad . \tag{5.2.7}$$

The coefficients $\Phi_\ell^{ij}(n|k)$ are determined by the sequence of transmission and reception times and the combining coefficients. Consequently, they are unknown, in general. Nevertheless, they have the following qualitative properties.

<u>Lemma 5.2.1</u>: (i) $$0 \leq \Phi_\ell^{ij}(n|k), \quad \forall i,j,\ell,n \geq k, \tag{5.2.8}$$

$$\sum_{j=1}^{M} \Phi_\ell^{ij}(n|k) \leq 1, \quad \forall i,\ell,n \geq k . \tag{5.2.9}$$

(ii) Under Assumptions 5.2.1, 5.2.4 and either Assumption 5.2.2 or 5.2.3, $\lim_{n\to\infty} \Phi_\ell^{ij}(n|k)$ exists, for any i,j,k,ℓ. The limit is independent of i and will be denoted by $\Phi_\ell^{j}(k)$. Moreover, there is a constant $\eta>0$ such that, if j is a computing processor for component ℓ, then

$$\Phi_\ell^{j}(k) \geq \eta , \quad \forall k. \tag{5.2.10}$$

The constant η, depends only on the constants introduced in our assumptions (i.e. B_o, B_1, β, α).

(iii) Under Assumptions 5.2.1, 5.2.2, 5.2.4, there exist $d \in [0,1)$, $B \geq 0$ (depending only on B_o, B_1, α) such that

$$\max_{i,j} |\Phi_\ell^{ij}(n|k) - \Phi_\ell^{j}(k)| \leq Bd^{n-k}, \quad \forall \ell,n \geq k . \tag{5.2.11}$$

(iv) Under Assumption 5.2.1, 5.2.3, 5.2.4, there exist $d \in [0,1)$, $\delta \in (0,1]$, $B \geq 0$ (depending only on B_o, B_1, β, α) such that

$$\max_{i,j} |\Phi_\ell^{ij}(n|k) - \Phi_\ell^{j}(k)| \leq Bd^{n^\delta - k^\delta}, \quad \forall \ell,n \geq k . \tag{5.2.12}$$

The proof of Lemma 5.2.1 is given in Appendix A, so as not to disrupt continuity. Let us rather try to interpret this result. Consider a scenario whereby at time n all processors cease computing; that is $s^i(m)=0, \ \forall m \geq n$. Suppose, however, that they keep communicating and combining. For this scenario, equation (5.2.7) yields

$$x_\ell^i(m) = \sum_{j=1}^{M} \Phi_\ell^{ij}(m|0)x_\ell^j(1) + \sum_{k=1}^{n-1} \sum_{j=1}^{M} \gamma^j(k)\Phi_\ell^{ij}(m|k)s_\ell^j(k). \tag{5.2.13}$$

Taking the limit, as $m\to\infty$, and using part (ii) of Lemma 5.2.1, we conclude that the limit exists, is independent of i and equals $y_\ell(n)$ defined by

$$y_\ell(n) = \sum_{j=1}^{M} \Phi^j(0)x_\ell^j(1) + \sum_{k=1}^{n-1} \sum_{j=1}^{M} \gamma^j(k)\Phi_\ell^j(k)s_\ell^j(k) \quad . \tag{5.2.14}$$

So, all processors asymptotically agree on a common value. Moreover, (5.2.10) states that the limit depends by a non-negligible factor on the updates of any computing processor. Finally, parts (iii) and (iv) of the Lemma quantify the natural relationship between the frequency of inter-processor communications and the speed at which agreement is reached.

It turns out that the results to be derived later depend only on the fact that Lemma 5.2.1 holds. For this reason we could, for example, remove Assumptions 5.2.3 and 5.2.4 and introduce inequality (5.2.12) as an independent Assumption. This would add some more generality to our results: for example, inequality (5.2.12) may be valid without the communication delays being bounded. Nevertheless, we choose to retain Assumptions 5.2.1-5.2.4 because they are easy to enforce or verify and are simpler to visualize.

We now continue with the development of the consequences of Lemma 5.2.1. For any pair (i,j) of processors, we define a linear transformation $\Phi^{ij}(n|k):H\to H$ by

$$\Phi^{ij}(n|k)x = (\Phi_1^{ij}(n|k)x_1,\ldots,\Phi_L^{ij}(n|k)x_L) \quad , \tag{5.2.15}$$

where $x=(x_1,\ldots,x_L)\in H$. Note that if each H_ℓ is one-dimensional, then $\Phi^{ij}(n|k)$ may be represented by a diagonal matrix. If each H_ℓ is finite-dimensional, $\Phi^{ij}(n|k)$ is a block-diagonal matrix, with the ℓ-th block being a multiple of the identity matrix of dimension equal to the dimension of H_ℓ. Finally, in the infinite dimensional case, $\Phi^{ij}(n|k)$ is a (bounded) linear operator, with very simple structure. We norm bounded linear operators Φ: $H\to H$, using the norm induced by the norm of H. That is,

$$||\Phi|| = \sup_{||x||\neq 0} \frac{||\Phi x||}{||x||} \quad . \tag{5.2.16}$$

It is then a trivial consequence of Lemma 5.2.1 that $\lim_{n\to\infty} \Phi^{ij}(n|k)$ exists and is independent of i; it will be denoted by $\Phi^j(k)$. In fact,

$$||\Phi^{ij}(n|k)-\Phi^j(k)|| = \max_\ell |\Phi^{ij}(n|k)-\Phi_\ell^j(k)|, \tag{5.2.17}$$

which in conjunction with (5.2.11) or (5.2.12) provides bounds on the convergence rate of $\Phi^{ij}(n|k)$.

We define a vector $y(n)\in H$ by

$$y(n) = \sum_{j=1}^{M} \Phi^j(0)x^j(1) + \sum_{k=1}^{n-1} \sum_{j=1}^{M} \gamma^j(k)\Phi^j(k)s^j(k) \quad . \tag{5.2.18}$$

This definition is consistent with (5.2.14). In fact, $y(n)=(y_1(n),\ldots,y_L(n))$. Notice that $y(n)$ is recursively generated by

$$y(n+1) = y(n) + \sum_{j=1}^{M} \gamma^j(n)\Phi^j(n)s^j(n) \quad . \tag{5.2.19}$$

The vector $y(n)$ is the element of H at which all processors would asymptotically agree if they were to stop computing (but keep communicating and combining) at a time n.

It may be viewed as a concise global summary of the state of computation at time n, in contrast with the vectors $x^i(n)$ which are the local states of computation; it allows us to look at the algorithm from two different levels: an aggregate and a more detailed one. We will see later that this vector $y(n)$ is also a very convenient tool for proving convergence results. We have noted earlier that equation (5.2.2) is a linear system. However, it is a fairly complicated one, whereas the recursion (5.2.19) corresponds to a very simple linear system in standard state space form. The content of the vector $y(n)$ and of the $\Phi^j(n)$'s is easiest to visualize in two special cases:

<u>Specialization</u>: (e.g. Examples I and III). Here $y(n)$ takes each component from the processor who specializes in that component. That is, $y(n)=(x_1^1(n),\ldots,x_M^M(n))$. Accordingly, $\Phi_j^i(n)=0$, for $i\neq j$, and $\Phi_j^j(n)=1$.

<u>Example V</u>: Here $\Phi^{ij}(n|k)$ is the ij-th entry of the matrix $\prod_{m=k+1}^{n-1} A(m)$. It follows that the limit of $\Phi^{ij}(n|k)$ is the ij-th entry of $\overline{A}$, which by our assumptions

depends only on j. Moreover, $y(n)$ equals any component of $\bar{A}x(n)$. (All components are equal by our assumptions.) If we multiply both sides of (5.2.6) by $\bar{A}$ and note that $\bar{A}A(n)=\bar{A}$, we obtain $y(n+1)=y(n)+\sum_{j=1}^{M} \bar{A}_{ij}s^j(n)$, which is precisely (5.2.19).

We conclude this section with some more discussion of the linear system (5.2.2). We first note that (5.2.2) is not a state space representation of a linear system because of the presence of delays. In other words, the present states of computation $\{x^i(n): i=1,\ldots,M\}$ and the future inputs $\{s^i(k): i=1,\ldots,M;\ k\geq n\}$ are not sufficient to determine the future values $x^i(k)$, $k\geq n$. A linear system with delays may be always cast in state space form, by means of a standard state augmentation procedure. In this context, the augmented state at time n should incorporate all messages that have been transmitted but not yet received. Without any assumptions on communication delays, this might require a state space of unbounded dimension. We note, however, a few special cases.

1. A finite-dimensional state space representation is possible if each H_ℓ is finite dimensional and the total number of messages that have been transmitted before time n and have not been received until time n is bounded by some constant independent of n. The augmented state consists of all these messages, together with the values of $x^i(n)$, for each i.

2. The boundedness condition above certainly holds if communication delays are bounded by some B_o. In such a case, we might consider the augmented state

$$(x^1(n),\ldots,x^M(n);\ldots;x^1(n-B_o),\ldots,x^M(n-B_o)) \ .$$

3. Finally, if communication delays are zero, then $t_\ell^{ij}(n)=n$, whenever $n\in T_\ell^{ij}$. In this case, (5.2.2) is already a linear system is standard state space form. (Compare with Example V and equation (5.2.6).) The state vector is $(x^1(n),\ldots,x^M(n))\in H^M$ and we may obtain easily

$$x^i(n+1) = \sum_{j=1}^{M} \Phi^{ij}(n+1|n-1)x^j(n) + \gamma^i(n)s^i(n) \tag{5.2.20}$$

$$x^i(n+1) = \sum_{j=1}^{M} \Phi^{ij}(n+1|m-1)x^j(m) + \sum_{k=m}^{n} \Phi^{ij}(n+1|k)\gamma^j(k)s^j(k) \quad , \; m<n \; , \tag{5.2.21}$$

$$y(n) = \sum_{j=1}^{M} \Phi^j(n-1)x^j(n). \tag{5.2.22}$$

We derive below a few inequalities pertaining to the zero-delay case which will be used in Section 5.4.

<u>Lemma 5.2.2</u>: Assume that communication delays are zero and that $\lim_{n\to\infty} \Phi^{ij}(n|k)$ exists, for all i,j,k, and is denoted by $\Phi^j(k)$. Then,

(i) $$\sum_{j=1}^{M} \Phi_\ell^{ij}(k|n) = \sum_{j=1}^{M} \Phi_\ell^j(n)=1, \qquad \forall i,k,n,\ell. \tag{5.2.23}$$

Let $x^1,\ldots,x^M\in H$. Then,

(ii) $$\max_i \left|\left|x^i - \sum_{j=1}^{M} \Phi^j(n)x^j\right|\right| \le \max_{i,j} \left|\left|x^i-x^j\right|\right| \; . \tag{5.2.24}$$

(iii) $$\left|\left|\sum_{j=1}^{M} \Phi^j(n)x^j\right|\right| \le \max_i \left|\left|x^i\right|\right|, \qquad \forall n. \tag{5.2.25}$$

(iv) $$\max_i \left|\left| \sum_{j=1}^{M} \Phi^{ij}(k|n)x^j \right|\right| \leq \max_i ||x^i||, \quad \forall k,n \tag{5.2.26}$$

(v) $$\left|\left| \sum_{j=1}^{M} \left(\Phi^{i_1 j}(k|n) - \Phi^{i_2 j}(k|n) \right) x^j \right|\right| \leq \max_{i,j} ||x^i - x^j||, \ \forall k,n,i_1,i_2 \tag{5.2.27}$$

(vi) $$\left|\left| \sum_{j=1}^{M} \left(\Phi^{i_1 j}(k|n) - \Phi^{i_2 j}(k|n) \right) x^j \right|\right| \leq$$

$$\leq 2M \max_{i,j} ||\Phi^{ij}(k|n) - \Phi^j(n)|| \max_{i,j} ||x^i - x^j||, \quad \forall i_1,i_2,k,n \tag{5.2.28}$$

<u>Proof</u>:

(i) Suppose that, for all i, $x^i(k)=0$, $\forall k \leq n$, that $\gamma^i(n)s^i(n)=z$, (for some $z \in H_\ell$) and that $s^i_\ell(k)=0$, $\forall k \neq n$. It follows that $x^i(n+1)=z$, $\forall i$ and, by induction, we obtain $x^i(k)=z$, $\forall k>n$, $\forall i$. Therefore, $\sum_{j=1}^{M} \Phi^{ij}_\ell(k|n)z=z$, $\forall k>n$, $\forall i$.

Since z was arbitrary, we obtain (5.2.23).

(ii) Using (5.2.23) and (5. 2.8), it follows that $\sum_{j=1}^{M} \Phi^j(n)x^j_\ell$ belongs to the convex hull of $\{x^1_\ell,\ldots,x^M_\ell\}$, for any component ℓ. Therefore,

$$\max_i ||x^i_\ell - \sum_{j=1}^{M} \Phi^j_\ell(n)x^j_\ell|| \leq \max_{i,j} ||x^i_\ell - x^j_\ell|| \tag{5.2.29}$$

We then take the maximum with respect to ℓ and recall that we are using the max-norm to obtain (5.2.24).

(iii) By convexity, again

$$\left|\left| \sum_{j=1}^{M} \Phi^j_\ell(n)x^j_\ell \right|\right| \leq \max_i ||x^i_\ell||, \quad \forall \ell \tag{5.2.30}$$

and the conclusion follows similarly, as in part (ii).

(iv) The proof is identical with that of part (iii).

(v) For any component ℓ, note that $\sum_{j=1}^{M} [\Phi_\ell^{i_1 j}(k|n) - \Phi_\ell^{i_2 j}(k|n)]x_\ell^j$ is the difference between two elements in the convex hull of $\{x_\ell^1,\ldots,x_\ell^M\}$. Hence it is bounded by $\max_{i,j} ||x_\ell^i - x_\ell^j||$. Taking the maximum over ℓ, we recover (5.2.27).

(vi) Note that, for any i, i_3, k, n,

$$\sum_{j=1}^{M} \Phi^{ij}(k|n)x^{i_3} = x^{i_3}, \tag{5.2.31}$$

due to (5.2.23). Hence, for any $i_3 \in \{1,\ldots,M\}$,

$$||\sum_{j=1}^{M} [\Phi^{i_1 j}(k|n) - \Phi^{i_2 j}(k|n)]x^j|| = ||\sum_{j=1}^{M} [\Phi^{i_1 j}(k|n) - \Phi^{i_2 j}(k|n)][x^j - x^{i_3}]|| \leq$$

$$\leq \sum_{j=1}^{M} \left(||\Phi^{i_1 j}(k|n) - \Phi^j(n)|| + ||\Phi^j(n) - \Phi^{i_2 j}(k|n)|| \right) ||x^j - x^{i_3}|| \leq$$

$$\leq 2M \max_{i,j} ||\Phi^{ij}(k|n) - \Phi^j(n)|| \max_{i,j} ||x^i - x^j|| . \quad \blacksquare \tag{5.2.32}$$

The model of computation introduced in this Section may be generalized in several directions [Bertsekas, Tsitsiklis and Athans, 1984]. To name a few examples, the updating rules of each processor need not have the linear structure of equation (5.2.2); also, it may be convenient to communicate other information, besides the values of components (e.g. derivatives of the cost function; see Section 5.6). In the most general case, we would expect that a model as general as that of Section 2.1 might be needed. However, except for Section 5.6 (which treats a special case), the present model is sufficient for our purposes.

5.3 DECENTRALIZED GRADIENT-LIKE ALGORITHMS

There is a large number of well-known centralized deterministic and stochastic optimization algorithms which have been analyzed using a variety of analytical tools [Avriel, 1976; Poljak and Tsypkin, 1973; Ljung, 1977a; Kushner and Clark, 1978; Poljak, 1976, 1977, 1978]. A large class of them, the so-called "pseudo-gradient" algorithms [Poljak and Tsypkin, 1973] have the distinguishing feature that the (expected) direction of update (conditioned upon the past history of the algorithm) is a descent direction with respect to the cost function to be minimized. Examples I-IV of Section 5.2 certainly have such a property. A larger list of examples is provided by Poljak and Tsypkin [1973] who also show that the development of results for pseudo-gradient algorithms leads easily to results for broader classes of algorithms, such as Kiefer-Wolfowitz stochastic approximation. In this section we present convergence results for the natural decentralized asynchronous versions of pseudo-gradient algorithms. We adopt the model of computation and the corresponding notation of Section 5.2.

We allow the initialization $\{x^1(1),\ldots,x^M(1)\}$ of the algorithm to be random with finite mean and variance. We also allow the updates $s^i(n)$ and the step-size $\gamma^i(n)$ of each processor, the combining coefficients $a_\ell^{ij}(n)$, the times of transmission, reception and computation to be random. (So, $t_\ell^{ij}(n)$ and the sets T_ℓ^i, T_ℓ^{ij} are random.) We assume that all random variables of interest are defined on some probability space $(\Omega,\mathcal{F},P)$†. We also introduce an increasing sequence $\{\mathcal{F}_n\}$

† In the case where H is infinite dimensional we must be precise on the meaning of measurability. We use the concept of strong measurability [Yosida, 1980]: a function $x:\Omega\to H$ is called $\mathcal{F}$-measurable if and only if there exists a sequence $\{x_n\}$ of finitely valued functions $x_n:\Omega\to H$ such that $\lim_{n\to\infty} x_n(\omega)=x(\omega)$, for almost all $\omega\in\Omega$. (The limit is is with respect to the norm on H.) A similar definition applies concerning measurability with respect to any other σ-field $\mathcal{F}_n\subset\mathcal{F}$.

of σ-fields contained in F, where F_n is a σ-field describing the history of the algorithm up to time n. In particular, we define F_n as the smallest σ-field such that $x^i(m)$, $\gamma^i(m)$, $m \leq n$ and $s^i(m)$, $m<n$, are F_n-measurable, for each i, such that events $n \in T^{ij}$, $n \in T^{ij}_\ell$, belong to F_n, for each i,j,ℓ,n and such that $t^{ij}_\ell(n)$, $a^{ij}_\ell(n)$ are F_n-measurable, for any $n \in T^{ij}_\ell$, for any i,j,ℓ.

Notice that in the above setting, $\Phi^{ij}_\ell(n|k)$ becomes a random variable, determined by the random sequence of communication times, delays and combining coefficients. Any interesting assumptions of a statistical nature on the communicating and combining process have direct counterparts in terms of $\Phi^{ij}_\ell(n|k)$.

While we would like to allow as much randomness in the algorithm as possible, certain restrictions have to be imposed. Namely, we have to avoid the possibility that a processor chooses to transmit at those times that it receives a "bad measurement" $s^i(n)$ or that it tends to give larger weights to bad measurements when forming convex combinations. This may be avoided in two ways: either by not allowing the processors to base their transmission decisions on the state of the algorithm, or by somehow ensuring that the long run outcome is independent of such decisions. This suggests the following assumption.

<u>Assumption 5.3.1</u>: (i) The limit $\Phi^j(k)$ of $\Phi^{ij}(n|k)$, as $n \to \infty$, exists and is independent of i, with probability 1. (This assumption will be strengthened later.)
(ii) For any $i \in \{1,\ldots,M\}$ and any m,n such that $m \leq n$, $\Phi^i(m)$ and $s^i(n)$ are conditionally independent, given F_n.

The main cases in which this assumption holds are the following:

1. If interprocessor communications and the combining coefficients are modelled as being deterministic. This does not mean that they have to be known in advance but forbids the processors to decide whether to transmit by looking at the value of a random update $s^i(n)$ or of their state vector $x^i(n)$.

2. More generally, if interprocessor communications and the combining coefficients are random but are generated by an "exogenous" source.

3. If $s^i(n)$ is a deterministic function of $x^i(n)$. In such a case, $s^i(n)$, conditioned upon F_n, is a constant, hence independent of $\Phi^i(k)$, for all k.

4. If $\Phi^i(k)$ is deterministic, for all k. This is much weaker than assuming that $\Phi^i(n|k)$ is deterministic, or that interprocessor communications are deterministic. For example, in the specialization case, $\Phi^i(k)$ is guaranteed to be deterministic.

In terms of the Examples of Section 5.2, the assumption of deterministic transmission and reception times may be relaxed, except for Example IV. This is because $\Phi^i(k)$ is deterministic in Examples I,III,V, and $s^i(n)$ is a deterministic function of $x^i(n)$ in Example II.

We assume that the objective of the algorithm is to minimize a nonnegative cost function $J:H \to [0,\infty)$. For the time being, we only assume that J is a smooth function. In particular, J is allowed to have several local minima.

<u>Assumption 5.3.2</u>: J is Frechet differentiable and its derivative satisfies the Lipschitz condition

$$||\nabla J(x)-\nabla J(x')|| \leq K||x-x'||, \quad \forall x,\ x' \in H\ , \tag{5.3.1}$$

where K is some nonnegative constant.

<u>Remark</u>: Since we allow the possibility of infinite dimensional spaces, $\nabla J(x)$ should be viewed as an element of H^*, the dual of H. As usual H^* is endowed with

the norm inherited from H. Similarly, we denote by $\nabla_\ell J(x)$ the derivative of J with respect to x_ℓ. This partial derivative belongs to H_ℓ^*, the dual of H_ℓ. In the finite dimensional case, however, $\nabla J(x)$ and $\nabla_\ell J(x)$ may be just viewed as vectors in H and H_ℓ, respectively. In fact, with the notation we employ below, they should be viewed as <u>row</u> vectors.

<u>Assumption 5.3.3</u>: The updates $s^i(n)$ of each processor satisfy

$$\nabla_\ell J(x^i(n))E[s_\ell^i(n)|F_n] \leq 0, \quad \text{a.s.}, \quad \forall i,\ell,n. \tag{5.3.2}$$

This assumption states that each component of each processor's update is in a descent direction, when conditioned on the past history of the algorithm and is satisfied by Examples I-IV of Section 5.2.

As an immediate consequence of (5.3.2) we obtain

$$E[\nabla J(x^i(n))\Phi^i(n)s^i(n)|F_n] = \sum_{\ell=1}^{L} E[\Phi_\ell^i(n)|F_n]\nabla_\ell J(x^i(n))E[s_\ell^i(n)|F_n] \leq 0 \text{ a.s.} \tag{5.3.3}$$

It turns out that (5.3.3) is all that is required in our proofs. On the other hand, it can be shown by means of simple examples that the condition

$$E[\nabla J(x^i(n))s^i(n)|F_n] \leq 0, \quad \text{a.s.}$$

is not sufficient for proving convergence.

The next assumption is easily seen to hold for Examples I and II of Section 5.2. For stochastic algorithms, it requires that the variance of the updates (and hence of any noise contained in them) goes to zero, as the gradient of the cost function goes to zero.

<u>Assumption 5.3.4</u>: For some $K_o \geq 0$ and for all i,ℓ,n,

$$E[||s_\ell^i(n)||^2] \leq -K_o E[\nabla_\ell J(x^i(n))s_\ell^i(n)] \ . \tag{5.3.4}$$

As a matter of verifying Assumption 5.3.4, one would typically check the validity of the slightly stronger condition

$$E[||s_\ell^i(n)||^2|F_n] \leq -K_o \nabla_\ell J(x^i(n))E[s_\ell^i(n)|F_n] \ .$$

Notice that, using Lemma 5.2.1 (ii) and Assumption 5.3.4 we obtain

$$\begin{aligned} E[||s^i(n)||^2] \leq \sum_{\ell=1}^{L} E[||s_\ell^i(n)||^2] &\leq -\frac{K_o}{\eta} \sum_{\ell=1}^{L} E[\nabla_\ell J(x^i(n))\Phi_\ell^i(n)s_\ell^i(n)] \\ &= -\bar{K}_o E[\nabla J(x^i(n))\Phi^i(n)s^i(n)], \end{aligned} \tag{5.3.5}$$

where $\bar{K}_o = K_o/\eta \geq 0$. It turns out that (5.3.5) is all we need for our results to hold.

Our first convergence result states that the algorithm converges in a suitable sense, provided that the step-size employed by each processor is small enough and that the time between consecutive communications is bounded, and applies to Examples I and II of Section 5.2. It should be noted, however, that Theorem 5.3.1 (as well as Theorem 5.3.2 later) does not prove yet convergence to a minimum or a stationary point of J. In particular, there is nothing in our assumptions that prohibits having $s^i(n)=0$, $\forall i,n$. Optimality is obtained later, using a few auxiliary and fairly natural assumptions (see Corollary 5.3.1).

Theorem 5.3.1: Let Assumptions 5.2.1, 5.2.2, 5.2.4 hold, with probability 1, and assume that the constants (B_o, B_1, α) involved in them are deterministic. Let also Assumptions 5.3.1-5.3.4 hold. Suppose that $\gamma^i(n) \geq 0$ and that $\sup_{i,n} \gamma^i(n) \leq \gamma_o < \infty$. (Here, γ_o is deterministic).
There exists a constant $\gamma^* > 0$ (depending on the constants introduced in the Assumptions) such that the inequality $0 < \gamma_o \leq \gamma^*$ implies:

a) $J(x^i(n))$, $i=1,2,\ldots,M$, as well as $J(y(n))$, converge almost surely, and to the same limit.

b) $\lim_{n\to\infty}(x^i(n)-x^j(n)) = \lim_{n\to\infty}(x^i(n)-y(n))=0$, $\forall i,j$, almost surely and in the mean square.

c) The expression

$$\sum_{n=1}^{\infty} \sum_{i=1}^{M} \gamma^i(n) \nabla J(x^i(n)) E[s^i(n) | F_n] \tag{5.3.6}$$

is finite, almost surely. Its expectation is also finite.

Leaving technical issues aside, the idea behind the proof of the above (and the next) Theorem is rather simple: the difference between $y(n)$ and $x^i(n)$, for any i, is of the order of $B\gamma_o$, where B is proportional to a bound on communication delays plus the time between consecutive communications between processors. Therefore, as long as γ_o remains small, $\nabla J(x^i(n))$ is approximately equal to $\nabla J(y(n))$; hence, $s^i_\ell(n)$ (and consequently $\Phi^i(n)s^i(n)$) is approximately in a descent direction, starting from point $y(n)$. Therefore, iteration (5.2.19) is approximately the same as a centralized descent (pseudo-gradient) algorithm which is, in general convergent [Poljak and Tsypkin, 1973].

Remark on Notation: In the course of the proofs in this section, we will use the symbol A to denote non-negative constants which are independent of n, γ_0, $\gamma^i(n)$, $x^i(n)$, $s^i(n)$ etc., but which may depend on the constants introduced in the various assumptions (that is, M, L, K, K_0, B_0, B_1, α, etc.). When A appears in different expressions, or even in different sides of the same equality (or inequality), it will not necessarily represent the same constant. (With this convention, an inequality of the form $A + 1 \leq A$ is meaningful and has to be interpreted as saying that A+1, where A is some constant, is smaller than some other constant, denoted again by A.) This convention is followed so as to avoid the introduction of unnecessarily many symbols.

Proof of Theorem 5.3.1: We introduce a new increasing sequence $\{F_n^o\}$ of σ-fields contained in F. In particular, we let F_n^o be the smallest σ-field containing F_n and such that $\Phi^i(k)$ is F_n^o-measurable, for all $k \leq n$. Equation (5.2.18) then implies that y(n) (and, therefore, J(y(n))) is F_n^o-measurable. Using the fact that $\Phi_\ell^i(n)$ is F_n^o-measurable, we obtain

$$\begin{aligned} E[\nabla_\ell J(x^i(n))\Phi_\ell^i(n)s_\ell^i(n)\,|\,F_n^o] &= \\ \Phi_\ell^i(n)\nabla_\ell J(x^i(n))E[s_\ell^i(n)\,|\,F_n^o] &= \\ \Phi_\ell^i(n)\nabla_\ell J(x^i(n))E[s_\ell^i(n)\,|\,F_n], & \end{aligned} \tag{5.3.6a}$$

where the last equality follows from Assumption 5.3.1. Using Assumption 5.3.3 we conclude that

$$E[\nabla J(x^i(n))\Phi^i(n)s^i(n)\,|\,F_n^o] \leq 0, \quad \text{a.s.} \tag{5.3.7}$$

Without loss of generality, we will assume that the algorithm is initialized so that $x^i(1)=0$, $\forall i$. In the general case where $x^i(1)\neq 0$, we may think of the algorithm as having started at time 0, with $x^i(0)=0$; then, a random update $s^i(0)$ sets $x^i(1)$ to a nonzero value. So, the case in which the processors initially disagree may be easily reduced to the case where they initially agree.

Note that we may define $\bar{s}^i(n) = \frac{\gamma^i(n)}{\gamma_0} s^i(n)$ and view $\bar{s}^i(n)$ as the new step with step-size γ_o. It is easy to see that Assumptions 5.3.3 and 5.3.4 also hold for $\bar{s}^i(n)$. For these reasons, no generality is lost if we assume that $\gamma^i(n) = \gamma_0$, $\forall n$ and this is what we will do.

Let us define

$$b(n) = \sum_{i=1}^{M} ||s^i(n)|| \tag{5.3.8}$$

and note that

$$b^2(n) \leq M \sum_{i=1}^{M} ||s^i(n)||^2 \leq Mb^2(n) \ .$$

Using (5.2.7), (5.2.18) and Lemma 5.2.1 (iii), we obtain

$$||y(n)-x^i(n)|| \leq \sum_{k=1}^{n-1} \sum_{j=1}^{M} \gamma_0 ||\Phi^j(k)-\Phi^{ij}(n|k)|| \ ||s^j(k)|| \leq$$

$$\leq A\gamma_0 \sum_{k=1}^{n-1} d^{n-k} b(k) \ . \tag{5.3.9}$$

From a Taylor series expansion for J we obtain

$$J(y(n+1)) = J\left(y(n)+\gamma_0 \sum_{i=1}^{M} \Phi^i(n)s^i(n)\right) \leq$$

$$\leq J(y(n)) + \gamma_0 \nabla J(y(n)) \sum_{i=1}^{M} \Phi^i(n)s^i(n) + A\left|\left|\gamma_0 \sum_{i=1}^{M} \Phi^i(n)s^i(n)\right|\right|^2$$

$$\leq J(y(n)) + \gamma_0 \nabla J(y(n)) \sum_{i=1}^{M} \Phi^i(n)s^i(n) + A\gamma_0^2 b^2(n) \ . \tag{5.3.10}$$

Assumption 5.3.3 is in terms of $\nabla J(x^i(n))$, whereas above we have $\nabla J(y(n))$. To overcome this difficulty, we use the Lipschitz continuity of the derivative of J and invoke (5.3.9) to obtain

$$\left|\left|\nabla J(y(n)) \sum_{i=1}^{M} \Phi^i(n)s^i(n) - \sum_{i=1}^{M} \nabla J(x^i(n))\Phi^i(n)s^i(n)\right|\right| \leq$$

$$\leq A \sum_{i=1}^{M} \left|\left|y(n)-x^i(n)\right|\right| \ \left|\left|s^i(n)\right|\right| \leq$$

$$\leq \gamma_0 A \sum_{k=1}^{n-1} d^{n-k} b(k) \sum_{i=1}^{M} \left|\left|s^i(n)\right|\right| = \gamma_0 A \sum_{k=1}^{n-1} d^{n-k} b(k) b(n) \leq$$

$$\leq \gamma_0 A \sum_{k=1}^{n-1} d^{n-k} [b^2(k)+b^2(n)] \ . \tag{5.3.11}$$

Let us define

$$G^i(n) = -\nabla J(x^i(n))\Phi^i(n)s^i(n) \ , \tag{5.3.12}$$

$$G(n) = \sum_{i=1}^{M} G^i(n), \tag{5.3.13}$$

and note that (5.3.3) implies that $E[G(n)] \geq 0$. We now rewrite inequality

(5.3.10), using (5.3.11) to replace the derivative term, to obtain:

$$J(y(n+1)) \leq J(y(n)) - \gamma_0 G(n) + A\gamma_0^2 b^2(n) + A\gamma_0^2 \sum_{k=1}^{n-1} d^{n-k}[b^2(k)+b^2(n)] \leq$$

$$\leq J(y(n)) - \gamma_0 G(n) + A\gamma_0^2 \sum_{k=1}^{n} d^{n-k} b^2(k) \quad . \tag{5.3.14}$$

Assumption 5.3.4 implies that [cf. inequality (5.3.5)]

$$E[b^2(k)] \leq AE[G(k)] \quad . \tag{5.3.15}$$

Taking expectations in (5.3.14) and using (5.3.15), we obtain

$$E[J(y(n+1))] \leq E[J(y(n))] - \gamma_0 E[G(n)] + A\gamma_0^2 \sum_{k=1}^{n} d^{n-k} E[G(k)] \quad . \tag{5.3.16}$$

We then sum (5.3.16), for different values of n, to obtain

$$0 \leq E[J(y(n+1))] \leq E[J(y(1))] + \sum_{k=1}^{n} \left[A \frac{\gamma_0^2}{1-d} - \gamma_0 \right] E[G(k)] \quad . \tag{5.3.17}$$

We now let $\gamma^* = \frac{1-d}{2A}$, where A is the constant of inequality (5.3.17), and assume that $0<\gamma_0 \leq \gamma^*$. Then, inequality (5.3.17) implies

$$\gamma_0 \sum_{k=1}^{n} E[G(k)] \leq 2E[J(y(1))], \ \forall n, \tag{5.3.18}$$

and letting n tend to infinity,

$$\gamma_0 \sum_{k=1}^{\infty} E[G(k)] < \infty \quad . \tag{5.3.19}$$

By Assumption 5.3.3 and (5.3.6a), $E[G(k)|F_k^o]\geq 0$, $\forall k$; we may apply the monotone convergence theorem to (5.3.19) and obtain

$$E\left[\sum_{k=1}^{\infty} E[G(k)|F_k^o]\right] = \sum_{k=1}^{\infty} E[G(k)] < \infty \tag{5.3.20}$$

which implies

$$\sum_{k=1}^{\infty} E[G(k)|F_k^o] < \infty, \quad \text{a.s.} \tag{5.3.21}$$

From (5.3.21) together with (5.3.6a) we obtain

$$\sum_{n=1}^{\infty} \nabla_\ell J(x^i(n))\Phi_\ell^i(n)E[s_\ell^i(n)|F_n] > -\infty, \qquad \text{a.s.}$$

Now use the fact (Lemma 5.2.1 (ii), inequality (5.2.10)) that $\Phi_\ell^i(n)\geq\eta>0$, for any computing processor i for component ℓ. This implies that

$$\sum_{k=1}^{\infty} \nabla_\ell J(x^i(n))E[s_\ell^i(n)|F_n] > -\infty, \quad \text{a.s.}$$

and establishes part (c) of the theorem.

<u>Lemma 5.3.1:</u> Let $X(n)$, $Z(n)$ be non-negative stochastic processes (with finite expectation) adapted to $\{F_n^o\}$ and such that

$$E[X(n+1)|F_n^o] \leq X(n) + Z(n), \tag{5.3.22}$$

$$\sum_{n=1}^{\infty} E[Z(n)] < \infty \quad . \tag{5.3.23}$$

Then $X(n)$ converges almost surely, as $n\to\infty$.

Proof of Lemma 5.3.1: By the monotone convergence theorem and (5.3.23) it follows that $\sum_{n=1}^{\infty} Z_n < \infty$, almost surely. Then, Lemma 5.3.1 becomes the same as Lemma 4.C.1 in [Ljung and Soderstrom, 1983, p.453].□

Now let A be the constant in the right hand side of (5.3.14) and let

$$Z(n) = A\gamma_0^2 E\left[\sum_{k=1}^{n} d^{n-k} b^2(k) \,|\, F_n^o\right]. \tag{5.3.24}$$

Then, $Z(n) \geq 0$ and by (5.3.15)

$$E[Z(n)] \leq A \sum_{k=1}^{n} d^{n-k} E[G(k)] \ . \tag{5.3.25}$$

Therefore,

$$\sum_{n=1}^{\infty} E[Z(n)] \leq A \sum_{n=1}^{\infty} \sum_{k=1}^{n} d^{n-k} E[G(k)] =$$

$$= A \frac{1}{1-d} \sum_{k=1}^{\infty} E[G(k)] < \infty \quad , \tag{5.3.26}$$

where the last inequality follows from (5.3.19). Therefore, Z(n) satisfies (5.3.23). We take the conditional expectation of (5.3.14), given F_n^o. Note that J(y(n)) is F_n^o-measurable and that $E[G(n) \,|\, F_n^o] \geq 0$. Therefore Lemma 5.3.1 applies and J(y(n)) converges almost surely.

Using Assumption 5.3.4 once more, together with (5.3.19),

$$E\left[\sum_{k=1}^{\infty} b^2(k)\right] \leq A \sum_{k=1}^{\infty} E[G(k)] < \infty \tag{5.3.27}$$

which implies that $b(k)$ converges to zero, almost surely. Recall (5.3.9) to conclude that $y(n)-x^i(n)$ converges to zero, almost surely. Also, by squaring (5.3.9), taking expectations and using the fact that $E[b^2(k)]$ converges to zero, we conclude that $E[||y(n)-x^i(n)||^2]$ also converges to zero, and this proves part (b) of the Theorem.

Lemma 5.3.2: Under Assumption 5.3.2, there exists some $A \geq 0$ such that

$$||\nabla J(x)||^2 \leq AJ(x), \qquad \forall x \in H . \tag{5.3.28}$$

Proof: By the definition of the induced norm on H^*

$$||\nabla J(x)|| = \sup_{\substack{y \in H \\ y \neq 0}} \frac{|\nabla J(x) y|}{||y||} .$$

Therefore, for any $\varepsilon > 0$, there exists some $y \in H$ such that $\nabla J(x) y \geq (1-\varepsilon) ||\nabla J(x)|| \; ||y||$. Moreover y can be scaled so that $||y|| = \frac{1}{L} ||\nabla J(x)||$. Then, a second order expansion for J yields

$$J(x-y) \leq J(x) - \frac{(1-\varepsilon)}{L} ||\nabla J(x)||^2 + \frac{1}{2L} ||\nabla J(x)||^2 . \tag{5.3.29}$$

Taking $\varepsilon < \frac{1}{2}$ and using the assumption that J is a nonnegative function, (5.3.28) follows.□

Since $J(y(n))$ converges, it is bounded; hence $\nabla J(y(n))$ is also bounded, by (5.3.28). We then use the fact that $y(n)-x^i(n)$ converges to zero, to concludes that $J(x^i(n))-J(y(n))$ also converges to zero. This proves part (a) and concludes the proof of the Theorem.■

Remark:

The choice of γ^* in our proof is not tight enough. A tighter value may be obtained in terms of the data of the problem (that is, in terms of M, L, K, K_0, etc.) by tracing back the inequalities which led to (5.3.17). Finally, the conclusions of the Theorem remain true, with the same γ^*, even if we replace the condition $\gamma_0 \leq \gamma^*$ by the weaker requirement $\limsup_{n\to\infty} \gamma^i(n) \leq \gamma^*$.

Decreasing Step-Size Algorithms

We now introduce a different set of assumptions. We allow the magnitude of the updates $s^i(n)$ to remain nonzero, even if $\nabla J(x^i(n))$ is zero (Examples III and IV of Section 5.2). Such situations are common in stochastic approximation algorithms or in system identification applications. Since the noise is persistent, the algorithm can be made convergent only by letting the step-size $\gamma^i(n)$ decrease to zero. The choice $\gamma^i(n) = 1/n$ is most commonly used and in the sequel we will assume that $\gamma^i(n)$ decreases at least as fast as $1/n$. Note that even if each processor selects its step-size according to a local clock or counter (which may be itself random), as long as these clocks do not operate in different time scales, we may assume that $\gamma^i(n) \leq A/n$, for all i and for some $A \geq 0$.

Since the step-size is decreasing, the algorithm becomes progressively slower as $n\to\infty$. This allows us to let the communications process become progressively slower as well, provided that it remains fast enough, when compared with the natural time scale of the algorithm, the latter being determined by the rate of decrease of the step-size. Such a possibility is captured by Assumption 5.2.3.

The following assumption, intended to replace Assumption 5.3.4, allows the noise to be persistent. Similarly with Assumption 5.3.4, it could be more naturally stated in terms of conditional expectations, but such a stronger version turns out to be unnecessary.

<u>Assumption 5.3.5</u>: For some $K_o, K_1, K_2, \geq 0$,

$$E[||s_\ell^i(n)||^2] \leq -K_o E[\nabla_\ell J(x^i(n)) s_\ell^i(n)] + K_1 E[J(x^i(n))] + K_2 . \tag{5.3.30}$$

In fact we will only use the following immediate consequence of (5.3.30) and Lemma 5.2.1 (iii). (Compare with inequality (5.3.5).)

$$E[||s^i(n)||^2] \leq -\bar{K}_o E[\nabla J(x^i(n)) \Phi^i(n) s^i(n)] + \bar{K}_1 E[J(x^i(n))] + \bar{K}_2 . \tag{5.3.31}$$

<u>Theorem 5.3.2</u>: Let Assumptions 5.2.1, 5.2.3, 5.2.4 hold, with probability 1, and assume that the constants B_o, B_1, α, β involved in them are deterministic. Let also Assumptions 5.3.1, 5.3.2, 5.3.3, 5.3.5 hold. Assume that, for some $K_3 \geq 0$, $\gamma^i(n) \leq K_3/n$, $\forall n, i$. Then,

a) $J(x^i(n))$, $i=1,2,\ldots,M$, as well as $J(y(n))$ converge almost surely, and to the same limit.

b) $\lim_{n\to\infty} (x^i(n) - x^j(n)) = \lim_{n\to\infty} (x^i(n) - y(n)) = 0$, $\forall i,j$, almost surely and in the mean square.

c) The expression

$$\sum_{n=1}^{\infty} \sum_{i=1}^{M} \gamma^i(n) \nabla J(x^i(n)) E[s^i(n) | F_n] \tag{5.3.32}$$

is finite, almost surely. Its expectation is also finite.

For the proof of Theorem 5.3.2, we will start with an auxiliary Lemma. In this Lemma we will bound certain infinite series by corresponding infinite integrals. This is justified as long as the integrand cannot change by more than a constant factor between any two consecutive integer points. For notational convenience, we use $c(n|k)$ to denote $d^{n^\delta - k^\delta}$, where d and δ are as in Lemma 5.2.1 (iv).

<u>Lemma 5.3.3:</u> The following hold:

(i) $$\sum_{k=1}^{\infty} \sum_{n=k}^{\infty} \frac{1}{n} \frac{1}{k} c(n|k) < \infty . \tag{5.3.33}$$

(ii) Let $\alpha \geq 0$. Then, there exists some $A \geq 0$ such that

$$\sum_{k=1}^{n} \frac{n}{k^{2+\alpha}} c(n|k) \leq An^{-\delta-\alpha}, \quad \forall n \geq 1 . \tag{5.3.34}$$

(iii) $$\lim_{n\to\infty} \sum_{k=1}^{n} \frac{n}{k^2} c(n|k) = \lim_{n\to\infty} \sum_{k=1}^{n} \frac{1}{k} c(n|k) = 0 . \tag{5.3.35}$$

(iv) $$\lim_{k\to\infty} \sum_{n=k}^{\infty} \frac{1}{n} c(n|k) = 0 . \tag{5.3.36}$$

<u>Proof of Lemma 5.3.3:</u> Let $t^\delta = y$; then $t = y^{1/\delta}$ and $dt = (1/\delta) y^{\frac{1}{\delta}-1} dy$. Notice that the left hand side of (5.3.34) is bounded by

$$An \int_1^n \frac{1}{t^{2+\alpha}} d^{n^\delta - t^\delta} dt = An \int_1^{n^\delta} \frac{1}{y^{(2+\alpha)/\delta}} d^{n^\delta - y} \frac{1}{\delta} y^{\frac{1}{\delta}-1} dy =$$

$$= An \int_1^{n^\delta} y^{-\frac{1}{\delta}-\frac{\alpha}{\delta}-1} d^{n^\delta - y} dy \leq An^{-\alpha-\delta},$$

which proves part (ii); part (iii) follows immediately, by letting $\alpha=0$ in (5.3.34). For part (i), we notice that the left hand side of (5.3.33) is bounded by

$$A \sum_{n=1}^{\infty} \frac{1}{n} \sum_{k=1}^{n} \frac{n}{k^2} c(n|k) \leq A \sum_{n=1}^{\infty} n^{-1-\delta} < \infty,$$

where the first inequality follows from (5.3.34) with $\alpha=0$. Finally, part (iv) is an immediate consequence of part (i).□

<u>Proof of Theorem 5.3.2</u>: Using the same arguments as in the proof of Theorem 5.3.1, we may assume, without loss of generality, that $x^i(1)=0$ and that $\gamma^i(n)=1/n$, $\forall i,n$. (Otherwise we could define $\bar{s}^i(n) = n\gamma^i(n)s^i(n)$.) Moreover, we define the σ-field $\mathcal{F}_n^o$ as in that proof and note that (5.3.7) is again valid.

We still use $c(n|k)$ to denote $d^{n^\delta - k^\delta}$. We define again $b(n)$, $G^i(n)$, $G(n)$ by (5.3.8), (5.3.12), (5.3.13), respectively, as in the proof of Theorem 5.3.1. Also, let

$$\phi(n|k) = \begin{cases} \frac{1}{n^2} + \sum_{m=1}^{n-1} \frac{1}{n} \frac{1}{m} c(n|m), & n=k, \\ \frac{1}{n} \frac{1}{k} c(n|k), & n>k, \\ 0, & n<k. \end{cases} \tag{5.3.37}$$

By replicating the steps leading to inequality (5.3.14) in the proof of Theorem 5.3.1 and using the definition (5.3.37) we obtain, for some $A \geq 0$,

$$J(y(n+1)) \leq J(y(n)) - \frac{1}{n} G(n) + A \sum_{k=1}^{n} \phi(n|k) b^2(k), \quad \forall n . \tag{5.3.38}$$

Taking expectations in (5.3.38), we have

$$E[J(y(n+1))] \leq E[J(y(n))] - \frac{1}{n} E[G(n)] + A \sum_{k=1}^{n} \phi(n|k) E[b^2(k)], \quad \forall n . \tag{5.3.39}$$

We would like to use Assumption 5.3.5 to bound $E[b^2(k)]$. However, these bounds are in terms of $E[J(x^i(n))]$, while (5.3.39) involves $E[J(y(n))]$. Nevertheless, we have:

<u>Lemma 5.3.4</u>: There is a constant $A \geq 0$ such that, for all $n \geq 1$,

$$E[b^2(n)] \leq A \sum_{k=1}^{n-1} \frac{n}{k^2} c(n|k) E[b^2(k)] + AE[G(n)] + AE[J(y(n))] + A \tag{5.3.40}$$

<u>Proof of Lemma 5.3.4</u>: Assumption 5.3.5 may be written as

$$E[b^2(n)] \leq AE[J(y(n))] + AE[G(n)] + A + A \sum_{i=1}^{M} E[J(x^i(n)) - J(y(n))], \tag{5.3.41}$$

and we need to bound the last term.

Using Lemma 5.3.2 and a second order series expansion for J we obtain:

$$J(x^i(n)) - J(y(n)) \leq ||\nabla J(y(n))|| \; ||x^i(n) - y(n)|| + A||x^i(n) - y(n)||^2 \leq$$

$$\leq \frac{1}{2} ||\nabla J(y(n))||^2 + A||x^i(n) - y(n)||^2 \leq AJ(y(n)) + A||x^i(n) - y(n)||^2 . \tag{5.3.42}$$

Similarly with (5.3.9), we have

$$||y(n)-x^i(n)|| \leq A \sum_{k=1}^{n-1} \frac{1}{k} c(n|k)b(k) \quad . \tag{5.3.43}$$

We now use (5.3.43) to bound the last term in (5.3.42). We obtain

$$||x^i(n)-y(n)||^2 \leq A\left[\sum_{k=1}^{n-1} \frac{1}{k} c(n|k)b(k)\right]^2 \leq$$

$$\leq An \sum_{k=1}^{n-1} \frac{c^2(n|k)}{k^2} b^2(k) \leq An \sum_{k=1}^{n-1} \frac{c(n|k)}{k^2} b^2(k) \quad . \tag{5.3.44}$$

We now take expectations in (5.3.42) and use (5.3.44) to recover (5.3.40). This completes the proof of Lemma 5.3.4.□

Inequality (5.3.40) bounds $b^2(n)$ in terms of past values of $b^2(k)$. We may recursively eliminate such past values and obtain:

<u>Lemma 5.3.5</u>: There exists a finite constant A and a kernel g(n,k) such that

$$E[b^2(n)] \leq A \sum_{k=1}^{n} g(n,k)\left[E[G(k)] + E[J(y(k)]+1\right] , \tag{5.3.45}$$

$$\sum_{k=1}^{n} g(n,k) \leq A, \quad \forall n , \tag{5.3.46}$$

$$\sum_{n=k}^{\infty} g(n,k) \leq A, \quad \forall k . \tag{5.3.47}$$

Proof of Lemma 5.3.5: Consider the linear (time-varying) system

$$X_n = A \sum_{k=1}^{n-1} \frac{n}{k^2} c(n|k) X_k + U_n , \tag{5.3.48}$$

where A is the constant in (5.3.40). By linearity, there exists a kernel $g(n,k)$ such that

$$X_n = \sum_{k=1}^{n} g(n,k) U_k . \tag{5.3.49}$$

If we let

$$U_n = A\left[E[G(n)] + E[J(y(n))]+1\right] \tag{5.3.50}$$

and use (5.3.40) we obtain (by an easy induction) $E[b^2(n)] \leq X_n$, $\forall n$, which is essentially (5.3.45).

In order to prove (5.3.46), we must show that the step response of (5.3.48) is bounded. That is, we assume $U_n=1$, $\forall n$, and we must show that the resulting sequence $\{X_n\}$ is bounded. Let us define a sequence $\{B_n\}$ by $B_1=1$ and, for $n>1$,

$$B_n = \max\left\{B_{n-1}, A \sum_{k=1}^{n-1} \frac{n}{k^2} c(n|k) B_{n-1}+1\right\} . \tag{5.3.51}$$

It follows from (5.3.48), with $U_n=1$, that

$$B_n \geq \max_{k \leq n} X_k . \tag{5.3.52}$$

So, it is sufficient to show that $\{B_n\}$ is a bounded sequence. But this follows easily from (5.3.51) and (5.3.34), with $\alpha=0$.

Now, in order to show that (5.3.47) holds, we need to show that the response of the system (5.3.48) to an impulse at time k_0 is summable and that the sum is bounded by a constant which is independent of k_0. So, we assume that $U_{k_0}=1$, $U_k=0$, $\forall k\neq k_0$ and we need to show

$$\sum_{n=k_0}^{\infty} X_n \leq A \ , \tag{5.3.53}$$

where A does not depend on k_0. Since the step response of (5.3.48) has been shown to be bounded, the impulse responses must be uniformly bounded; that is,

$$X_n \leq A, \quad \forall n \geq k_0 \tag{5.3.54}$$

where X_n is the response to an impulse at time k_0 and A is independent of k_0. We will show by induction that for every $\varepsilon>0$ and every integer $m\geq 0$, there exists a constant A, independent of k_0, such that

$$X_n \leq An^{-m\delta} + A\,\frac{n^{1+\varepsilon m}}{k_0^2}\,c(n|k_0), \quad \forall n > k_0 \ . \tag{5.3.55}$$

Let us fix some $\varepsilon>0$. Clearly, (5.3.55) is satisfied for m=0, due to (5.3.54). Assume it is true for some arbitrary integer $m\geq 0$. We then use (5.3.55) in (5.3.48) (recall that $U_{k_0}=1$, $U_k=0$, $\forall k\neq k_0$ and $X_{k_0}=1, X_{k_0}=0$, $\forall k<k_0$) to obtain, for $n>k_0$:

$$X_n \leq A\,\frac{n}{k_0^2}\,c(n|k_0)+A\sum_{k=k_0+1}^{n-1}\frac{n}{k^2}\,c(n|k)\left[Ak^{-m\delta}+A\,\frac{k^{1+\varepsilon m}}{k_0^2}\,c(k|k_0)\right] .$$

Using (5.3.34), with $\alpha=m\delta$,

$$\sum_{k=k_0+1}^{n-1} \frac{n}{k^2} c(n|k)k^{-m\delta} \leq An^{-(m+1)\delta},$$

where A is independent of k_0. Also note that $c(n|k)c(k|k_0) = c(n|k_0)$ and that

$$\sum_{k=k_0+1}^{n-1} \frac{n}{k^2} c(n|k) \frac{k^{1+\varepsilon m}}{k_0^2} c(k|k_0) \leq \frac{n^{1+\varepsilon m}}{k_0^2} c(n|k_0) \sum_{k=k_0+1}^{n-1} \frac{1}{k} \leq A \frac{n^{1+\varepsilon(m+1)}}{k_0^2} c(n|k_0) \tag{5.3.56}$$

where A depends on ε but not on k_0. (The last inequality follows from the fact that for any $\varepsilon>0$, there exist A, A_1, such that $\sum_{k=1}^{n} \frac{1}{k} \leq A_1 \log n \leq An^{\varepsilon}$.) This shows that (5.3.55) is also valid for m+1, and completes the induction.

Taking m large enough so that $m\delta>1$, the term $An^{-m\delta}$ in (5.3.55) becomes summable, and the sum is independent of k_0. It remains to show the summability of the last term in (5.3.55). Proceeding as in the proof of Lemma 5.3.3 and letting $\varepsilon_0 = m\varepsilon$, we have:

$$\sum_{n=k_0}^{\infty} \frac{n^{1+\varepsilon_0}}{k_0^2} c(n|k_0) \leq A \int_{t=k_0}^{\infty} \frac{t^{1+\varepsilon_0}}{k_0^2} d^{t^\delta - k_0^\delta} dt \leq$$

$$\leq \frac{A}{k_0^2} \int_{y=k_0^\delta}^{\infty} y^{\left(\frac{2+\varepsilon_0}{\delta} - 1\right)} d^{y-k_0^\delta} dy \leq \frac{A}{k_0^2} k_0^{2+\varepsilon_0-\delta} = Ak_0^{\varepsilon_0-\delta} . \tag{5.3.57}$$

Since ε was arbitrary, it may be chosen small enough so that $\varepsilon_0=m\varepsilon<\delta$. Then, the right hand side of (5.3.57) converges to zero, as $k_0\to\infty$, and (a fortiori) it is bounded by a constant independent of k_0. This concludes the proof of Lemma 5.3.5.□

We may now use the bounds given by Lemma 5.3.5 in (5.3.39) to obtain, after some rearrangement,

$$E[J(y(n+1))] \leq E[J(y(n))] - \frac{1}{n} E[G(n)] + A \sum_{m=1}^{n} q(n,m)[E[G(m)]+E[J(y(m))]+1] \tag{5.3.58}$$

where

$$q(n,m) = \sum_{k=m}^{n} \phi(n|k) g(k,m) \ . \tag{5.3.59}$$

<u>Lemma 5.3.6</u>:

$$\sum_{n=1}^{\infty} \sum_{k=1}^{n} q(n,k) < \infty, \tag{5.3.60}$$

$$\lim_{k\to\infty} k \sum_{n=k}^{\infty} q(n,k) = 0 \ . \tag{5.3.61}$$

<u>Proof of Lemma 5.3.6</u>: Using the definition of $\phi(n|m)$ and (5.3.46), we obtain after some rearrangement

$$\sum_{n=1}^{\infty} \sum_{k=1}^{n} q(n,k) \leq A \sum_{n=1}^{\infty} \sum_{m=1}^{n} \phi(n|m) < \infty \ , \tag{5.3.62}$$

where the last inequality follows from (5.3.33), (5.3.37). Similarly, using (5.3.47)

$$k \sum_{n=k}^{\infty} q(n,k) = k \sum_{n=k}^{\infty} \sum_{m=k}^{n} \phi(n|m) g(m|k) = k \sum_{m=k}^{\infty} g(m,k) \sum_{n=m}^{\infty} \phi(n|m) \leq$$

$$\leq Ak \sup_{m\geq k} \sum_{n=m}^{\infty} \phi(n|m) \leq A \sup_{m\geq k} \left\{ \frac{k}{m^2} + \sum_{\ell=1}^{m-1} \frac{k}{m\ell} c(m|\ell) + \sum_{n=m+1}^{\infty} \frac{k}{mn} c(n|m) \right\} \leq$$

$$\leq \frac{A}{k} + A \sup_{m\geq k} \sum_{\ell=1}^{m-1} \frac{1}{\ell} c(m|\ell) + A \sup_{m\geq k} \sum_{n=m+1}^{\infty} \frac{1}{n} c(n|m) \ . \tag{5.3.63}$$

The second and the third expression in the right hand side of (5.3.63) are easily seen to converge to zero, as $k\to\infty$, because of (5.3.35) and (5.3.36), respectively. This concludes the proof of Lemma 5.3.6.□

Let us define

$$R_n = A \sum_{m=1}^{n} q(n,m)E[G(m)] - \frac{1}{n} E[G(n)], \tag{5.3.64}$$

and note that (5.3.58) may be written as

$$E[J(y(n+1))] \leq E[J(y(n))] + R_n + A \sum_{m=1}^{n} q(n,m)[E[J(y(m))]+1] \tag{5.3.65}$$

We also define P_n by $P_1=E[J(y(1))]$,

$$P_{n+1} = P_n + R_n + A \sum_{m=1}^{n} q(n,m)[P_m+1] \tag{5.3.66}$$

and note that $P_n \geq E[J(y(n))]$, $\forall n$. By linearity, there exists a kernel $V(n,k)$ such that

$$P_{n+1} = \sum_{k=1}^{n} V(n,k+1) \left[R_k + A \sum_{m=1}^{k} q(k,m)\right] + V(n,1)P_1 . \tag{5.3.67}$$

Clearly, $V(n,k) \geq 1$, $\forall n,k$. Moreover, inequality (5.3.60) implies

$$\prod_{n=1}^{\infty} \left[1 + \sum_{m=1}^{n} q(n,m)\right] < \infty$$

which leads to the conclusion that, for some constant V^*, $V(n,k) \leq V^*$, $\forall n,k$. Using Lemma 5.3.6,

$$\sum_{k=1}^{n} V(n,k+1) \sum_{m=1}^{k} q(k,m) \leq V^* \sum_{k=1}^{\infty} \sum_{m=1}^{k} q(k,m) \leq A < \infty . \tag{5.3.68}$$

Using the definition of R_n, we obtain

$$\sum_{k=1}^{n} V(n,k+1)R_k \leq \sum_{k=1}^{n} \left[V^*A \sum_{m=k}^{n} q(m,k) - \frac{1}{k}\right] E[G(k)]. \tag{5.3.69}$$

Lemma 5.3.6 implies that $V^*A \sum_{m=k}^{n} q(m,k) - \frac{1}{k}$ becomes negative for large enough k, which shows that $\sum_{k=1}^{n} V(n,k+1)R_k$ is bounded above by some constant A for all n. So, we may conclude from (5.3.67) that the sequence $\{P_n\}$ is bounded. Hence, for some $A \geq 0$,

$$E[J(y(n))] \leq A, \ \forall n \tag{5.3.70}$$

From (5.3.67), (5.3.68) we conclude that

$$\liminf_{n\to\infty} \sum_{k=1}^{n} V(n,k+1)R_k > -\infty \tag{5.3.71}$$

Using (5.3.69) and Lemma 5.3.6 again, we obtain

$$\sum_{k=1}^{\infty} \frac{1}{k} E[G(k)] < \infty . \tag{5.3.72}$$

Using the monotone convergence theorem, as in the proof of Theorem 5.3.1, the proof of part (c) is completed.

We now define

$$Z(n) = \sum_{k=1}^{n} \phi(n|k)b^2(k) . \tag{5.3.73}$$

Using Lemma 5.3.5 and similarly with (5.3.58), we obtain

$$\sum_{n=1}^{\infty} E[Z(n)] \leq A \sum_{n=1}^{\infty} \sum_{m=1}^{n} q(n,m)\left[E[G(m)+E[J(y(m))]+1\right]. \tag{5.3.74}$$

Using Lemma 5.3.6, (5.3.70) and (5.3.72) we conclude that $\sum_{n=1}^{\infty} E[Z(n)] < \infty$. Now recall inequality (5.3.38). Taking conditional expectations, we obtain

$$E[J(y(n+1))|F_n^o] \leq J(y(n)) + E[Z(n)|F_n^o] , \tag{5.3.75}$$

and using Lemma 5.3.1, it follows that $J(y(n))$ converges almost surely.

We now turn to the proof of part (b) of the Theorem. Let

$$B_n = \max_i \max_{1\leq k\leq n} E[||s^i(k)||^2], \tag{5.3.76}$$

$$D_n = \max_i E[||x^i(n)-y(n)||^2]. \tag{5.3.77}$$

Inequality (5.3.44) implies that

$$D_{n+1} \leq \beta(n)B_n , \tag{5.3.78}$$

where $\beta(n)$ is a sequence which converges to zero, by (5.3.35). Moreover, using (5.3.30),

$$\begin{aligned} E[||s^i(n)||^2] &\leq AE[J(x^i(n))] - E[2A\nabla J(x^i(n))\tfrac{1}{2} s^i(n))] + A \leq \\ &\leq AE[J(x^i(n))] + 4A^2E[||\nabla J(x^i(n))||^2] + \tfrac{1}{4} E[||s^i(n)||^2] + A \leq \\ &\leq AE[J(x^i(n))] + \tfrac{1}{4} E[||s^i(n)||^2] + A, \end{aligned} \tag{5.3.79}$$

where the last inequality follows from Lemma 5.3.2. This finally implies that

$$E[||s^i(n)||^2] \leq AE[J(x^i(n))] + A. \tag{5.3.80}$$

Taking expectations in (5.3.42) we have

$$E[J(x^i(n))] \leq AE[J(y(n))] + AD_n . \tag{5.3.81}$$

Inequalities (5.3.80) and (5.3.81) imply

$$E[||s^i(n)||^2] \leq AE[J(y(n))] + AD_n + A \tag{5.3.82}$$

and

$$B_n \leq AQ + A\max_{k\leq n} D_k \ , \tag{5.3.83}$$

where $Q = 1+ \sup_{n\geq 1} E[J(y(n))]$. Combining (5.3.78) and (5.3.83), we have

$$D_{n+1} \leq \beta(n)(AQ+A\max_{k\leq n} D_k) \tag{5.3.84}$$

and since $\beta(n)$ converges to zero, so does D_n. This proves that $x^i(n)-y(n)$ converges to zero in the mean square.

As a corollary to the preceding, we obtain

$$\sup_n E[b^2(n)]<\infty \ . \tag{5.3.85}$$

Let

$$C_k = \sum_{k^{1/\delta}\leq i<(k+1)^{1/\delta}} \frac{1}{i} b(i), \quad k\geq 1 \ . \tag{5.3.86}$$

Using the fact that there exists an A such that $(k+1)^{1/\delta}-k^{1/\delta} \leq Ak^{(1/\delta)-1}, \forall k$, we obtain from (5.3.86)

$$\sum_{k=1}^{\infty} E[C_k^2] \leq \sum_{k=1}^{\infty} \left[\sum_{k^{1/\delta}\leq i<(k+1)^{1/\delta}} \frac{1}{i} \right]^2 \sup_n E[b^2(n)] \leq$$

$$\leq A \sum_{k=1}^{\infty} \frac{1}{k^{2/\delta}} [(k+1)^{1/\delta}-k^{1/\delta}]^2 \leq A \sum_{k=1}^{\infty} \frac{1}{k^{2/\delta}} k^{(2/\delta)-2} <\infty \ . \tag{5.3.87}$$

It follows that C_k^2 converges to zero, almost surely. Consequently, so does C_k and $\sum_{k=1}^{n} d^{n-k}C_k$ as well. Let N denote the largest integer such that $N\leq n^{\delta}$ and use (5.3.43) to obtain

$$||x^i(n)-y(n)|| \leq A \sum_{k=1}^{n-1} \frac{1}{k} c(n|k)b(k) \leq A \sum_{1\leq k\leq n^\delta-1} \sum_{k^{1/\delta}\leq i<(k+1)^{1/\delta}} \left[\frac{1}{i} d^{n^\delta-i^\delta} b(i)\right] \leq$$

$$\leq A \sum_{1\leq k\leq n^\delta-1} d^{N-(k+1)} \sum_{k^{1/\delta}\leq i<(k+1)^{1/\delta}} \frac{1}{i} b(i) \leq A \sum_{k=1}^{N} d^{N-k} C_k \ . \qquad (5.3.88)$$

As n converges to infinity, so does N and, by the above discussion, $x^i(n)-y(n)$ converges to zero, as $n\to\infty$. Consequently, $x^i(n)-x^j(n)$ also converges to zero, for any i,j, completing the proof of part (b).

Finally, since $J(y(n))$ converges and $x^i(n)-y(n)$ converges to zero, part (a) of the theorem follows, as in the proof of Theorem 5.3.1.■

We continue with a corollary which shows that, under reasonable conditions, convergence to a stationary point or a global optimum may be guaranteed. We only need to assume that away from stationary points some processor will make a positive improvement in the cost function. Naturally, we only require the processors to make positive improvements at times that they are not idle, that is at times $t\in T_\ell^i$.

We first need some technical background: an integer-valued random variable t is called a "stopping time" (with respect to $\{F_n\}$) if and only if the event $\{t\leq n\}$ belongs to F_n, for every n. Given a stopping time t, we define F_t to be the σ-field generated by those events $A\in F$ such that $A\cap\{t\leq n\}$ belongs to F_n, for every n. Intuitively, F_t describes all events which have occurred up to the random time t.

<u>Corollary 5.3.1</u>: Suppose that for some $K_4>0$, $\gamma^i(n)\geq K_4/n$, $\forall n,i$. Assume that H is finite dimensional, J has compact level sets and that there exist continuous functions g_ℓ^i: $H\to[0,\infty)$ such that

$$\nabla_\ell J(x^i(t))E[s^i_\ell(t)|F_t]\leq -g^i_\ell(x^i(t)), \tag{5.3.89}$$

for every stopping time t satisfying $t\in T^i_\ell$, with probability 1.

We define $g: H\to[0,\infty)$ by $g(x) = \sum_{i=1}^{M}\sum_{\ell=1}^{L} g^i_\ell(x)$ and we assume that any point $x\in H$ satisfying $g(x)=0$ is a stationary point of J. Finally, suppose that the difference between consecutive elements of T^i_ℓ is bounded, for any i,ℓ such that $T^i_\ell\neq\phi$. Then,

a) Under the Assumptions of either Theorem 5.3.1 or 5.3.2,

$$\liminf_{n\to\infty} ||\nabla J(x^i(n))||=0, \quad \forall i, \quad \text{a.s.} \tag{5.3.90}$$

b) Under the Assumptions of Theorem 5.3.1 and if (for some $\varepsilon>0$) $\gamma^i(n)\geq\varepsilon$, $\forall i,n$, we have

$$\lim_{n\to\infty} ||\nabla J(x^i(n)||=0, \ \forall i, \quad \text{a.s.} \tag{5.3.91}$$

and any limit point of $\{x^i(n)\}$ is a stationary point of J.

c) Under the Assumptions of either Theorem 5.3.1 or 5.3.2 and if every point satisfying $g(x)=0$ is a minimizing point of J (this is implicitly assuming that all stationary points of J are minima), then

$$\lim_{n\to\infty} J(x^i(n)) = \inf_{x\in H} J(x) \ . \tag{5.3.92}$$

<u>Proof of Corollary 5.3.1</u>: From part (c) of either Theorem 5.3.1 or 5.3.2 and (5.3.89) we obtain

$$\sum_{i=1}^{M}\sum_{\ell=1}^{L}\sum_{n\in T^i_\ell} \gamma^i(n)g^i_\ell(x^i(n))<\infty, \quad \text{a.s.} \tag{5.3.93}$$

Because of our assumption on the sets T_ℓ^i, it follows that there exists a positive integer c such that, for any i,ℓ,m, the interval $\{cm+1, cm+2, \ldots, c(m+1)\}$ contains at least one element of T_ℓ^i. Let us choose sequences of such elements denoted by $t_{\ell,m}^{i\,*}$. By (5.3.93), we have

$$\sum_{i=1}^{M} \sum_{\ell=1}^{L} \sum_{m=1}^{\infty} \gamma^i(t_{\ell,m}^i) g_\ell^i(x^i(t_{\ell,m}^i)) < \infty, \quad \text{a.s.} \tag{5.3.94}$$

Now notice that, for some constant $K_5 > 0$,

$$\gamma^i(t_{\ell,m}^i) \geq \frac{K_4}{t_{\ell,m}^i} \geq \frac{K_4}{c(m+1)} \geq \frac{K_5}{m}, \quad \forall i,\ell,m . \tag{5.3.95}$$

Hence, (5.3.94) yields

$$\sum_{m=1}^{\infty} \frac{1}{m} \sum_{i=1}^{M} \sum_{\ell=1}^{L} g_\ell^i(x^i(t_{\ell,m}^i)) < \infty, \quad \text{a.s.} \tag{5.3.96}$$

From either Theorem 5.3.1 or 5.3.2 and its proof we obtain $\lim_{n\to\infty} (x^i(n) - y(n)) = \lim_{n\to\infty} (y(n+1) - y(n)) = 0$ which implies that

$$\lim_{m\to\infty} (x^i(t_{\ell,m}^i) - y(t_{1,m}^1)) = 0, \qquad \forall i,\ell . \tag{5.3.97}$$

Since J has compact level sets and $J(y(n))$ converges, the sequence $\{y(n)\}$ is bounded. We therefore need to consider the functions g_ℓ^i only on a compact set on which they are uniformly continuous. Therefore,

$$\lim_{m\to\infty} (g_\ell^i(x^i(t_{\ell,m}^i)) - g_\ell^i(y(t_{1,m}^1))) = 0, \quad \forall i,\ell . \tag{5.3.98}$$

---

*Clearly, we can choose these sequences so that each $t_{\ell,m}^i$ is a stopping time.

By combining (5.3.96) and (5.3.98) we obtain

$$\liminf_{m\to\infty} \sum_{i=1}^{M} \sum_{\ell=1}^{L} g_\ell^i(y(t_{1,m}^1))=0 \ . \tag{5.3.99}$$

a) By (5.3.99), there must be some subsequence of $\{t_{1,m}^1\}$ along which $g(y(t_{1,m}^1))$ converges to zero. Let y^* be a limit point of the corresponding subsequence of $\{y(t_{1,m}^1)\}$. By continuity, $g(y^*)=0$ and by assumption, y^* must be a stationary point of J, so $\nabla J(y^*)=0$. Moreover, $x^i(t_{1,m}^1)$ also converges to y^* along the same subsequence. By continuity of ∇J, (5.3.90) follows.

b) In this case, (5.3.94) implies

$$\lim_{m\to\infty} \sum_{i=1}^{M} \sum_{\ell=1}^{L} g_\ell^i(x^i(t_{\ell,m}^i))=0, \qquad \text{a.s.} \tag{5.3.100}$$

and the rest of the proof is the same as for part (a), except that we do not need to restrict ourselves to a convergent subsequence.

c) From part (a) we conclude that some subsequence of $\{y(t_{1,m}^1)\}$ converges to some y^* for which $g(y^*)=0$. Consequently, y^* minimizes J. Using the continuity of J,

$$\liminf_{n\to\infty} J(y(n)) \le \liminf_{n\to\infty} J(y(t_{1,n}^1)) \le J(y^*) = \inf_{x\in H} J(x) \ .$$

On the other hand, $J(y(n))$ converges (part (a) of either Theorem 5.3.1 or 5.3.2) which shows that (5.3.92) holds. ∎

We now discuss the above corollary and apply it to the examples of Section 5.2. Notice first that the assumption on T_ℓ^i states that, for each component ℓ, the time between successive computations of s_ℓ^i is bounded, for any computing processor i for that component. Such a condition will be always met in practice. The assumption $\gamma^i(n) \geq K_3/n$ may be enforced without the processors having access to a global clock. For example, apart from the trivial case of constant step-size, we may let $\gamma^i(n)=1/t_n^i$, where t_n^i is the number of times, before time n, that processor i has performed a computation.

For Examples I and III, (5.3.89) holds with $g_i^i(x)$ a constant multiple of $|\nabla_i J(x)|^2$; for Examples II and IV, it holds with $g^i(x)$ a constant multiple of $||\nabla J(x)||^2$. We may conclude that Corollary 5.3.1 applies and proves convergence for Examples I-IV.

5.4 DECENTRALIZED STOCHASTIC ALGORITHMS WITH CORRELATED NOISE

We have considered thus far stochastic algorithms with decreasing step-size, under the crucial descent Assumption 5.3.3. However, there is a large number of applications of stochastic algorithms in which such an assumption is violated [Ljung, 1977a]. Many such applications are related to identification of ARMAX models. or adaptive control of stochastic systems [Ljung, 1977b]. Martingale theory is not directly applicable to the study of such algorithms, but results have been obtained via other approaches. Ljung [1977a] has shown that convergence may be studied in terms of the stability properties of an autonomous, deterministic, ordinary differential equation (ODE). (Hence, this method of analysis has been called the "ODE approach".) The proof exploits the fact that as the step-size

becomes smaller, the algorithm evolves in a progressively slower time scale, while the dynamics of the underlying uncertainties evolve in a constant time scale. Hence, we have -asymptotically - a separation of time scales; the stochastic effects may be averaged out and we are left with a deterministic ordinary differential equation. Similar results have been obtained by Kushner and Clark [1978] and have been further developed and unified by Metivier and Priouret [1984].

The main deficiency of the ODE approach is that convergence may be proved only under the assumption that the algorithm returns infinitely often to a bounded region. This assumption needs to be verified by means of different approaches, or it may have to be enforced directly by modifying the algorithm, e.g. by projecting the relevant variables back into a bounded region, whenever the algorithm moves away from this region. This may create new problems because the bounded region into which we project should contain the desired point of convergence and, typically, this point is itself unknown.

In Section 5.3 we have shown that the natural decentralized versions of descent-type stochastic algorithms have essentially the same convergence properties as their centralized counterparts. We will show below that the same is true, without the descent assumption. In particular, we will show that the main conclusions of the ODE approach still hold (appropriately modified). Of course, since a "returning" assumption is essential for proving convergence of centralized algorithms, such a condition has to be assumed for decentralized algorithms as well.

We adopt again the model of computation (and the corresponding notation) of Section 5.2. We now introduce some assumptions, which will be discussed later. In particular, we assume the following:

(i) H is a finite dimensional Euclidean space.

(ii) $\lim_{n\to\infty} \Phi^{ij}(n|k)$ exists for all i,j,k and is independent of i. (See Lemma 5.2.1 for some sufficient conditions for this assumption to hold).

(iii) Communication Delays are zero. For future reference, we collect here the relevant equations from Section 5.2. Namely, for any i, $n \geq m \geq 1$ we have:

$$x^i(n+1) = \sum_{j=1}^{M} \Phi^{ij}(n+1|n-1)x^j(n) + \gamma^i(n)s^i(n) \tag{5.4.1}$$

$$x^i(n+1) = \sum_{j=1}^{M}\left[\Phi^{ij}(n+1|m-1)x^j(m) + \sum_{k=m}^{n} \Phi^{ij}(n+1|k)\gamma^j(k)s^j(k)\right] \tag{5.4.2}$$

$$y(n) = \sum_{j=1}^{M} \Phi^j(n-1)x^j(n), \tag{5.4.3}$$

$$y(n+1) = y(n) + \gamma^i(n) \sum_{j=1}^{M} \Phi^j(n)s^j(n) . \tag{5.4.4}$$

Also,

<u>Assumption 5.4.1</u>: $\gamma^i(n) = 1/n, \forall n,i.$

<u>Assumption 5.4.2</u>: There exists a stochastic process $\{\phi(n)\}$, taking values in a Euclidean space S, defined on some probability space (Ω,F,P), and a set of functions $Q^i: \mathbb{N}\times H\times S \to H$, $i=1,\ldots,M$, such that

$$s^i(n) = Q^i(n,x^i(n),\phi(n)), \quad \forall i,n . \tag{5.4.5}$$

Moreover, there exist functions K_1: $S\times R \to R$, $K_2: H\to R$ and K_3: $R \to R$ such that

$$||Q^i(n,x,\phi)-Q^i(n,x',\phi)|| \leq K_1(\phi,C)||x-x'||, \quad \forall n,\phi,i . \tag{5.4.6}$$

whenever $||x||<C$, $||x'||\leq C$; without loss of generality we assume that $K_1(\cdot,\cdot)$ is increasing in C, for each ϕ. Also, for some $N\in\mathbb{N}$,

$$||Q^i(n,x,\phi) - Q^i(n,x,\phi')|| \leq K_2(x)(||\phi||^N + ||\phi'||^N)||\phi-\phi'||, \quad \forall n,x,\phi,\phi', \tag{5.4.7}$$

and

$$||K_1(\phi,C) - K_1(\phi',C)|| \leq K_3(C)(||\phi||^N + ||\phi'||^N)||\phi-\phi'||, \quad \forall \phi,\phi',C . \tag{5.4.8}$$

<u>Assumption 5.4.3</u>: The process $\{\phi(n)\}$ is generated by

$$\phi(0) = e(0) \tag{5.4.9}$$

$$\phi(n+1) = A(n)\phi(n) + e(n), \quad n\geq 0 \tag{5.4.10}$$

where $\{e(n)\}$ is a sequence of independent random vectors. Moreover, there exist some $c>0$, $\lambda\varepsilon[0,1)$ such that

$$||\prod_{k=n}^{m-1} A(k)|| \leq c\lambda^{m-n}, \quad m>n . \tag{5.4.11}$$

<u>Assumption 5.4.4</u>: $\sup_n E[||e(n)||^p]<\infty, \quad p>1.$ (5.4.12)

Note that (5.4.10),(5.4.11),(5.4.12) imply

$$\sup_n E[||\phi(n)||^p]<\infty, \ \forall p>1. \tag{5.4.13}$$

Let us define

$$f^i(n,x) = E[Q^i(n,x,\phi(n))] \tag{5.4.14}$$

$$f(n,x) = \sum_{i=1}^{M} \Phi^i(n) f^i(n,x) . \tag{5.4.15}$$

For any $\Delta>0$, let $m(n,\Delta)$ be a sequence of integers such that

$$\sum_{i=n}^{m(n,\Delta)-1} \frac{1}{i} \leq \Delta \leq \sum_{i=n}^{m(n,\Delta)} \frac{1}{i}, \tag{5.4.16}$$

and note that for large n,

$$\Delta \approx \ln \frac{m(n,\Delta)}{n}, \tag{5.4.17}$$

which implies that $m(n,\Delta)$ is approximately equal to ne^{Δ}.

<u>Assumption 5.4.5</u>: There exists a twice continuously differentiable function $V: H \to [0,\infty)$ with a unique global minimum x^* satisfying

$$\frac{\partial V}{\partial x}(x) \neq 0, \quad x \neq x^* \tag{5.4.18}$$

$$V(x^*) = 0, \quad \lim_{|x|\to\infty} V(x) = \infty \tag{5.4.19}$$

and such that for some $A_1>0$, and for any $\Delta>0$, $x\in H$,

$$\limsup_{n\to\infty} \left\{ \left[\sum_{k=n}^{m(n,\Delta)} \frac{1}{k} f(k,x) \right] \cdot \frac{\partial V}{\partial x} \right\} \leq -A_1 \left|\left| \frac{\partial V}{\partial x}(x) \right|\right|^2 \Delta . \tag{5.4.20}$$

<u>Assumption 5.4.6</u>: For some $A_2>0$ and for any $\Delta>0$, $x\in H$,

$$\limsup_{n\to\infty} \max_{n\leq k\leq m(n,\Delta)} \left|\left| \sum_{t=n}^{k} \frac{1}{t} f(t,x) \right|\right| \leq A_2 \left|\left| \frac{\partial V}{\partial x}(x) \right|\right| \Delta, \tag{5.4.21}$$

$$\limsup_{n\to\infty} \max_{\substack{n\leq k\leq m(n,\Delta) \\ 1\leq i\leq M}} \left|\left| \sum_{t=n}^{k} \frac{1}{t} \sum_{j=1}^{M} \Phi^{ij}(k|t) f^j(t,x) \right|\right| \leq A_2 \left|\left| \frac{\partial V}{\partial x}(x) \right|\right| \Delta. \tag{5.4.22}$$

Assumption 5.4.7: There exist some $\Delta_0>0$, $d\in[0,1)$ such that

$$\limsup_{n\to\infty} \max_{k\geq m(n,\Delta_0)} ||\Phi^{ij}(k|n) - \Phi^j(n)||<\frac{d}{2M}, \ \forall i,j . \tag{5.4.23}$$

Remarks and Interpretation of the Assumptions

1. The assumption that communication delays are zero is not essential for the result to be proved. Similar results may be obtained under the assumption that communication delays are bounded, provided that the returning condition (see the statement of Theorem 5.4.1) is slightly strengthened to require that at the times n_k the magnitude of state $x^i(n)$ of each processor, as well as of any message that has been transmitted but not yet received are bounded by the constant C.

2. Assumption 5.4.1 may be generalized to allow for a larger class of sequences $\{\gamma(k)\}$ satisfying $\sum_{k=1}^{\infty}\gamma(k) = \infty$, $\sum_{k=1}^{\infty}\gamma^p(k)<\infty$, for some $p>1$ (see Ljung [1977a]), including choices where processors need not to have access to a global clock. Nevertheless, the choice $\{1/k\}$ is representative enough and simplifies notation.

3. Assumption 5.4.2 states that the updates $s^i(n)$ are a function of the current state vector $x^i(n)$ and an underlying random process $\{\phi(n)\}$, not necessarily white. Inequalities (5.4.6), (5.4.7),(5.4.8) may be easily checked if Q^i is known. They effectively require that Q^i is twice continuously differentiable and that the derivatives (viewed as functions of ϕ) increase slower than some polynomial.

4. Assumption 5.4.3 defines a model for the underlying stochastic process $\{\phi(n)\}$. While this structure is convenient, it is far from necessary. In fact any non-linear model for $\{\phi(n)\}$ would still lead to the same results, provided

that the dependence of $\phi(n)$ on $\phi(k)$ decreases "fast enough", as n-k increases. Assumptions 5.4.3 and 5.4.4 are only used to prove the "averaging" Lemma 5.4.3. Consequently, our results remain valid whenever the conclusions of Lemma 5.4.3 can be shown to be true, possibly using different means.

5. The main difference of our assumptions from those of Ljung [1977a] is that we do not allow any feedback from the algorithm to influence the evolution of the process $\{\phi(n)\}$. For this reason, the Theorem that follows is not applicable to algorithms for adaptive control, nor to certain identification algorithms. More general models, allowing feedback, are considered later.

6. Assumption 5.4.5 guarantees that the direction of updates is a descent direction with respect to the function V. A simpler version of Assumption 5.4.5 would be to require

$$f(k,x) \cdot \frac{\partial V}{\partial x}(x) \leq -A_1 \left|\left|\frac{\partial V}{\partial x}(x)\right|\right|^2, \ \forall k,x, \tag{5.4.24}$$

or the even stronger version

$$f^i(k,x) \cdot \frac{\partial V}{\partial x}(x) \leq -A_1 \left|\left|\frac{\partial V}{\partial x}(x)\right|\right|^2, \ \forall k,x,i \ . \tag{5.4.25}$$

Our version (5.4.20) has the following advantages over (5.4.24) or (5.4.25):

a) We do not require the direction $f^i(k,x)$ of update of each processor to be a descent direction. Conditions are only placed on $f(k,x) = \sum_{i=1}^{M} \Phi^i(k) f^i(k,x)$ which is in a sense the total direction in which the processors (viewed as a whole) update.

b) Our condition in on $\sum_{k=n}^{m(n,\Delta)} \frac{1}{k} f(k,x)$ which is an average direction of update over some time interval, rather than $f(k,x)$ which is the direction of update at a single time instance. In terms of applications, Assumption 5.4.5 has the advantage that it allows $f(k,x)$ to be zero at some times. Thus, we do not require the processors to obtain measurements or perform computations at each time instance. Rather, we put a constraint on the average amount of computations they will perform over some time interval.

7. Assumption 5.4.5 is much weaker than the pseudogradient Assumption 5.3.3. This is because the inequality $f^i(k,x) \cdot \frac{\partial V}{\partial x}(x) \leq 0$, $\forall x$, does not imply, in general,

$$E[Q^i(k,x^i(n),\phi(x))\,|\,\phi(n-1),x^i(n)] \cdot \frac{\partial V}{\partial x}(x^i(n)) \leq 0$$

when the process $\{\phi(n)\}$ is non-white.

8. Inequality (5.4.20) also implies (using the Schwartz inequality) that

$$\liminf_{n\to\infty} \left|\left| \sum_{k=n}^{m(n,\Delta)} \frac{1}{k} f(k,x) \right|\right| \geq A_1\Delta \left|\left| \frac{\partial V}{\partial x}(x) \right|\right| . \tag{5.4.26}$$

9. Assumption 5.4.6 requires that the magnitude of the expected updates is not too big, compared with $||\frac{\partial V}{\partial x}(x)||$. It is satisfied, in particular, if, for some A

$$||f^i(t,x)|| \leq A||\frac{\partial V}{\partial x}(x)||, \quad \forall x,t,i, \tag{5.4.27}$$

but also covers a few more general cases (see Remark 6).

10. Note that if $s^i(n)$ is bounded by some constant A, for each i,n, the vector x^i will move by an amount of the order of Δ_0 during the time interval $[n,m(n,\Delta_0)]$. So, this time interval may be viewed as unit time in a time scale naturally associated with the algorithm. Assumption 5.4.7 implies that the disagreement between processors tends to decrease by a factor of $d<1$ during such a time unit. The above statement can be made precise as follows:

Let $D(n) = \max_{i,j} ||x^i(n)-x^j(n)||$ and suppose that $s^i(k)=0$, $\forall k\geq n$, $\forall i$. Then, Assumption 5.4.7 implies that, for n large enough, $D(m(n,\Delta_0))\leq d \cdot D(n+1)$. (A proof may be easily obtained from (5.4.23) and Lemma 5.2.2 (vi)).

Natural conditions that guarantee that Assumption 5.4.7 holds may be easily obtained, as in Lemma 5.2.1. Let us just point out here that for the specialization case, Assumption 5.4.7 is satisfied if every processor communicates to every other processor once during the time interval $[n,m(n,\Delta_0)]$. Since $m(n,\Delta_0)\approx ne^{\Delta_0}$, this allows the time between consecutive communications to increase exponentially. So, for the time being, we allow the communication process to be even slower than that assumed for the purposes of Theorem 5.3.2.

We now proceed to the first result of this section:

<u>Theorem 5.4.1:</u> Let $C<\infty$. Let Assumptions 5.4.1-5.4.7 hold. Then, there exist constants $\Delta_0^*(C)> 0$, $d^*(C)\in[0,1)$ such that: if the constants Δ_0, d of Assumption 5.4.7 satisfy $\Delta_0\leq\Delta_0^*(C)$, $d\leq d^*(C)$, then the following is true:

If for almost all $\omega\in\Omega$ there exists a subsequence $\{n_k\}$ of the positive integers such that

$$||x^i(n_k)||\leq C, \quad \forall i,k \tag{5.4.28}$$

and if the point x^* minimizing V satisfies

$$||x^*|| < C, \tag{5.4.29}$$

then

$$\lim_{n\to\infty} x^i(n) = x^*, \quad \forall i, \quad \text{almost surely.} \tag{5.4.30}$$

Remark: Later in this section we will indicate differences and similarities with the results of Ljung, and how an ordinary differential equation might enter the picture.

Notation: For convenience, we present here a list of some notation to be used in the course of the proof. In particular, given some $n \in \mathbb{N}$, $\bar{y} \in H$, $\Delta > 0$, $C > 0$, we let:

$$C(n) = \max\{||\bar{y}||, \max_i\{||x^i(n)||\}\} \tag{5.4.31}$$

$$D(n) = \max_{i,j} ||x^i(n) - x^j(n)|| \tag{5.4.32}$$

$$P(n) = ||y(n) - \bar{y}|| \tag{5.4.33}$$

$$q(n,\Delta,C) = \sum_{k=n}^{m(n,\Delta)} \frac{1}{k} K_1(\phi(k),C) \tag{5.4.34}$$

$$\varepsilon(n,\Delta,\bar{y}) = \left|\left|\sum_{k=n}^{m(n,\Delta)} \frac{1}{k} \sum_{i=1}^{M} \Phi^i(k)[Q^i(k,\bar{y},\phi(k)) - f^i(k,\bar{y})]\right|\right| \tag{5.4.35}$$

$$\varepsilon_1(n,\Delta,\bar{y}) = \max_{n\le k\le m(n,\Delta)} \left|\left|\sum_{t=n}^{k} \frac{1}{t} \sum_{i=1}^{M} \Phi^i(t)[Q^i(t,\bar{y},\phi(t)) - f^i(t,\bar{y})]\right|\right| \tag{5.4.36}$$

$$\varepsilon_2(n,\Delta,\bar{y}) = \max_i \max_{n\le k\le m(n,\Delta)} \left|\left|\sum_{t=n}^{k} \frac{1}{t} \sum_{j=1}^{M} \Phi^{ij}(k|t) \cdot [Q^j(t,\bar{y},\phi(t)) - f^j(t,\bar{y})]\right|\right| \tag{5.4.37}$$

$$F(n,\Delta,\bar{y}) = \sum_{k=n}^{m(n,\Delta)} \frac{1}{k} f(k,\bar{y}) \tag{5.4.38}$$

$$F_1(n,\Delta,\bar{y}) = \max_{n\le k\le m(n,\Delta)} \left|\left| \sum_{t=n}^{k} \frac{1}{t} f(t,\bar{y}) \right|\right| \tag{5.4.39}$$

$$F_2(n,\Delta,\bar{y}) = \max_{i} \max_{n\le k\le m(n,\Delta)} \left|\left| \sum_{t=n}^{k} \frac{1}{t} \sum_{j=1}^{M} \Phi^{ij}(k|t) f^j(t,\bar{y}) \right|\right| \tag{5.4.40}$$

$$\tilde{\varepsilon}(n,\Delta,\bar{y}) = \varepsilon_1(n,\Delta,\bar{y}) + \varepsilon_2(n,\Delta,\bar{y}) \tag{5.4.41}$$

$$G(n,\Delta,\bar{y}) = \tilde{\varepsilon}(n,\Delta,\bar{y}) + F_1(n,\Delta,\bar{y}) + F_2(n,\Delta,\bar{y}) \ . \tag{5.4.42}$$

We also recall for future reference some elementary inequalities, most of them consequences of Lemma 5.2.2:

$$||x^i(n)-y(n)|| \le D(n) \tag{5.4.43}$$

$$\left|\left| \sum_{j=1}^{M} \Phi^{ij}(k|n-1)x^j(n) \right|\right| \le C(n) \tag{5.4.44}$$

$$||y(n)|| \le C(n) \tag{5.4.45}$$

$$\left|\left| \sum_{j=1}^{M} [\Phi^{i_1 j}(k|n)-\Phi^{i_2 j}(k|n)]a^j \right|\right| \le \max_{i_1,i_2} ||a^{i_1}-a^{i_2}|| \le 2 \max_{i} ||a^i|| \tag{5.4.46}$$

$$\left|\left| \sum_{j=1}^{M} [\Phi^{i_1 j}(k|n)-\Phi^{i_2 j}(k|n)]a^j \right|\right| \le 2Mc(k|n) \max_{i_1,i_2} ||a^{i_1}-a^{i_2}|| \tag{5.4.47}$$

where

$$c(k|n) = \max_{i,j} ||\Phi^{ij}(k|n)-\Phi^j(n)|| \ . \tag{5.4.48}$$

The proof of Theorem 5.4.1 proceeds through a sequence of Lemmas.

<u>Lemma 5.4.1</u>: Let $\{R(k)\}$ be a sequence of zero-mean random variables such that $E[R(k)R(j)] \leq A\lambda^{|k-j|}$, for some $A>0$, $\lambda \in [0,1)$. Then for any $\Delta>0$,

$$\lim_{n\to\infty} \max_{n\leq k\leq m(n,\Delta)} \left| \sum_{\ell=n}^{k} \frac{1}{\ell} R(\ell) \right| = 0, \quad \text{almost surely} . \tag{5.4.49}$$

<u>Proof</u>: The proof is essentially given by Ljung [1977a]: An ergodic theorem of Cramer and Leadbetter implies that

$$\lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} R(k) = 0, \quad \text{almost surely} \tag{5.4.50}$$

Let

$$z(n) = \frac{1}{n} \sum_{k=1}^{n} R(k) . \tag{5.4.51}$$

Then,

$$z(n) = z(n-1) + \frac{1}{n} (R(n) - z(n-1)) \tag{5.4.52}$$

and, therefore,

$$\max_{n\leq k\leq m(n,\Delta)} \left| \sum_{\ell=n}^{k} \frac{1}{\ell} R(\ell) \right| =$$

$$= \max_{n\leq k\leq m(n,\Delta)} \left| z(k) - z(n-1) + \sum_{\ell=n}^{k} \frac{1}{\ell} z(\ell-1) \right| \leq$$

$$\leq (2+\Delta) \sup_{k\leq n-1} |z(k)| , \tag{5.4.53}$$

which converges to zero by virture of (5.4.50). □

<u>Lemma 5.4.2</u>: Let $\{R(k|i);\ k\in\mathbb{N},\ i\in\mathbb{N}\}$ be a double sequence of zero-mean random variables such that

$$E[R(k|k_1)R(k|k_2)R(k|k_3)R(k|k_4)]\leq A\lambda^{k_4-k_1} \tag{5.4.54}$$

for some $A>0$, $\lambda\varepsilon[0,1)$, whenever $k_1\leq k_2\leq k_3\leq k_4$. Fix some $\Delta>0$ and let

$$w(k|n) = \sum_{i=n}^{k} \frac{1}{i} R(k|i), \quad k\geq n \tag{5.4.55}$$

$$u(n) = \max_{n\leq k\leq m(n,\Delta)} |w(k|n)| \tag{5.4.56}$$

Then

$$\lim_{n\to\infty} u(n)=0, \quad \text{almost surely.} \tag{5.4.57}$$

<u>Proof</u>: Using (5.4.54) and (5.4.55),

$$E[w^4(k|n)]\leq \frac{24}{n^4} \sum_{k_1=n}^{k} \sum_{k_2=k_1}^{k} \sum_{k_3=k_2}^{k} \sum_{k_4=k_3}^{k} A\lambda^{k_4-k_1} =$$

$$= \frac{24A}{n^4} \sum_{k_1=n}^{k} \sum_{k_2=k_1}^{k} \lambda^{k_2-k_1} \sum_{k_3=k_2}^{k} \lambda^{k_3-k_2} \sum_{k_4=k_3}^{k} \lambda^{k_4-k_3} \leq$$

$$\leq \frac{24Ak}{n^4}\left(\frac{1}{1-\lambda}\right)^3 \tag{5.4.58}$$

On the other hand, there exists some constant α such that (see equation (5.4.17))

$$m(n,\Delta)\leq \alpha n \tag{5.4.59}$$

Therefore,

$$E[w^4(k|n)] \leq \frac{24A\alpha}{n^3}\left(\frac{1}{1-\lambda}\right)^3 , \quad n \leq k \leq m(n,\Delta) \tag{5.4.60}$$

and

$$\sum_{n=1}^{\infty} \sum_{k=n}^{m(n,\Delta)} E[w^4(k|n)] \leq \sum_{n=1}^{\infty} \frac{24A}{n^3} (m(n,\Delta)-n)\left(\frac{1}{1-\lambda}\right)^3$$

$$\leq \sum_{n=1}^{\infty} \frac{24A\alpha^2}{n^2}\left(\frac{1}{1-\lambda}\right)^3 < \infty \quad . \tag{5.4.61}$$

Therefore,

$$\lim_{n\to\infty} \sum_{k=n}^{m(n,\Delta)} w^4(k|n)=0, \quad \text{almost surely,} \tag{5.4.62}$$

which implies (5.4.57). □

<u>Lemma 5.4.3</u>: Given $C>0$, there exist constants A', $A(C)$ such that for any $y \in H$, $||y|| \leq C$, and for any $\Delta>0$, the following are true, almost surely,

a) $$\lim_{n\to\infty} \varepsilon(n,\Delta,y)=0, \tag{5.4.63}$$

b) $$\limsup_{n\to\infty} q(n,\Delta,C) < A(C)\Delta, \tag{5.4.64}$$

c) $$\lim_{n\to\infty} \varepsilon_1(n,\Delta,y)=0, \tag{5.4.65}$$

d) $$\lim_{n\to\infty} \varepsilon_2(n,\Delta,y)=0, \tag{5.4.66}$$

e) $$\limsup_{n\to\infty} G(n,\Delta,y) \leq A'\left|\left|\frac{\partial V}{\partial y}(y)\right|\right|\Delta . \tag{5.4.67}$$

<u>Proof:</u> a) Note that $0 \leq \varepsilon(n,\Delta,y) \leq \varepsilon_1(n,\Delta,y)$ and for (5.4.63) it is sufficient to prove (5.4.65).

b) By (5.4.34),

$$q(n,\Delta,C) = \sum_{k=n}^{m(n,\Delta)} \frac{1}{k} E[K_1(\phi(k),C)] + \\ + \sum_{k=n}^{m(n,\Delta)} \frac{1}{k} [K_1(\phi(k),C) - E[K_1(\phi(k),C)]] \tag{5.4.68}$$

By virtue of inequalities (5.4.8) and (5.4.13), it follows that the first sum is bounded above by $A(C)\Delta$ for some $A(C) > 0$. Concerning the second sum, we use linearity (Assumption 5.4.3) to write, for $k>t$,

$$\phi(k) = \psi(k|t)\phi(t) + \tilde{\phi}(k|t) \ , \tag{5.4.69}$$

where $\tilde{\phi}(k|t)$ is independent of $\phi(t)$. Also, (5.4.11) implies

$$||\psi(k|t)|| \leq c\lambda^{k-t} \tag{5.4.70}$$

for some $c \geq 0$, $\lambda \in [0,1)$. Then

$$|\mathrm{Cov}[K_1(\phi(k),C),\ K_1(\phi(t),C)]| \leq \\ \leq |\mathrm{Cov}[K_1(\tilde{\phi}(k|t),C),\ K_1(\phi(t),C)]| + \\ + \left|\mathrm{Cov}\left[\{K_1(\tilde{\phi}(k|t) + \psi(k|t)\phi(t),C) - K_1(\tilde{\phi}(k|t),C)\},K_1(\phi(t),C)\right]\right| \tag{5.4.71}$$

The first term in the right-hand side of (5.4.71) is zero because $\tilde{\phi}(k|t)$ is independent of $\phi(t)$. Concerning the second term we use the Schwartz inequality and (5.4.8) to bound it by

$$E\left[\left|K_1(\tilde{\phi}(k|t)+\psi(k|t)\phi(t),C) - K_1(\tilde{\phi}(k|t),C)\right|^2\right]^{1/2} E\left[\left|K_1(\phi(t),C)\right|^2\right]^{1/2} \leq$$

$$\leq E\left[(||\phi(k)||^N + ||\tilde{\phi}(k|t)||^N)^2 K_3(C)^2 ||\psi(k|t)||^2 ||\phi(t)||^2\right]^{1/2} \cdot$$

$$\cdot E[|K_1(\phi(t),C)|^2]^{1/2} . \tag{5.4.72}$$

Using (5.4.13) and inequality (5.4.70), we conclude that there is some new constant $A(C) \geq 0$ and some $\lambda \in [0,1)$ such that

$$E\left[\{K_1(\phi(k),C)-E[K_1(\phi(k),C)]\} \cdot \{K_1(\phi(t),C) - E[K_1(\phi(t),C)]\}\right]$$
$$\leq A(C)\lambda^{|k-t|}, \quad \forall k,t . \tag{5.4.73}$$

Using Lemma 5.4.1, we can see that the last sum in (5.4.68) converges to zero, thus proving (5.4.64).

c) The proof is identical to that of part (b). Instead of (5.4.8), we use (5.4.7) to conclude that

$$||\mathrm{Cov}[Q^i(k,\phi(k),y),Q^i(t,\phi(t),y)]|| \leq A\lambda^{|k-t|} , \tag{5.4.74}$$

for some A, depending on y only; we then use Lemma 5.4.1.

d) Fix some i,j,ℓ and consider the double sequence of scalar random variables $R_\ell(k|t)$ defined by

$$R_\ell(k|t) = \Phi_\ell^{ij}(k|t)[Q_\ell^j(t,y,\phi(t))-f_\ell^j(t,y)] . \tag{5.4.75}$$

(The subscript ℓ indicates that we are dealing with the ℓ-th component.) We intend to apply Lemma 5.4.2; we therefore need to show that condition (5.4.54) is satisfied. Without loss of generality, we assume that $Q_\ell^j(t,y,\phi(t))$ is

zero-mean, i.e. $f_\ell^j(t,y)=0$, $\forall t$. For the purposes of the current argument j,ℓ,y are held fixed; so, for the proof of this Lemma only, we will use the short-hand notation $Q(t,\phi(t))$ instead of $Q_\ell^j(t,y,\phi(t))$. Finally, note that $\Phi_\ell^{ij}(k|t)$ is bounded above by 1. Hence, to verify (5.4.57) we need to show that, for $A>0$, $\lambda\in[0,1)$ and for all $t_1\leq t_2\leq t_3\leq t_4$,

$$E[Q(t_1,\phi(t_1))Q(t_2,\phi(t_2))Q(t_3,\phi(t_3))Q(t_4,\phi(t_4))]\leq A\lambda^{t_4-t_1} . \tag{5.4.76}$$

By Assumption 5.4.2,

$$|Q(t_4,\phi)-Q(t_4,\phi')|< P_1(||\phi||,\ ||\phi'||)||\phi-\phi'||, \quad \forall\phi,\phi' , \tag{5.4.77}$$

where P_1 is some polynomial independent of t_4. Let

$$r_4(\phi) = E[Q(t_4,\phi(t_4))|\phi(t_3)=\phi] . \tag{5.4.78}$$

Then, using the notation introduced by (5.4.69) and (5.4.78)

$$\begin{aligned}|r_4(\phi)-r_4(\phi')|&=\Big|E\Big[Q\big(t_4,\psi(t_4|t_3)\phi + \tilde{\phi}(t_4|t_3)\big) - \\ &\qquad - Q\big(t_4,\phi(t_4|t_3)\phi' + \tilde{\phi}(t_4|t_3)\big)\Big]\Big| \\ &\leq E\Big[P_1\Big(||\psi(t_4|t_3)\phi + \tilde{\phi}(t_4|t_3)||,\ ||\psi(t_4|t_3)\phi'+ \tilde{\phi}(t_4|t_3)||\Big)\Big]\cdot \\ &\qquad ||\psi(t_4|t_3)||\cdot||\phi-\phi'||\end{aligned} \tag{5.4.79}$$

(The expectations in the above inequality, are with respect to $\tilde{\phi}(t_4|t_3)$.) Note that $||\psi(t_4|t_3)||\leq c\lambda^{t_4-t_3}$, where c,λ are the constants of Assumption 5.4.3.

Moreover, using (5.4.13), it is easy to see that there exists a polynomial $P_2(\cdot,\cdot)$, which does not depend on t_3, t_4 such that

$$E\Big[P_1\big(||\psi(t_4|t_3)\phi + \tilde{\phi}(t_4|t_3)||,||\psi(t_4|t_3)\phi' + \tilde{\phi}(t_4|t_3)||\big)\Big]$$

$$\leq P_2(||\phi||,||\phi'||) \ . \tag{5.4.80}$$

Therefore, for some polynomial $P_2(\cdot,\cdot)$,

$$|r_4(\phi)-r_4(\phi')| \leq cP_2(||\phi||,||\phi'||)\lambda^{t_4-t_3} \tag{5.4.81}$$

Using (5.4.77), with t_3 in the place of t_4 and (5.4.81), and defining

$$p_3(\phi) = Q(t_3,\phi)r_4(\phi), \tag{5.4.82}$$

it is easy to see that

$$|p_3(\phi)-p_3(\phi')| \leq P_3(||\phi||,||\phi'||)\lambda^{t_4-t_3} \ , \quad \forall \phi,\phi' \ , \tag{5.4.83}$$

for some polynomial $P_3(\cdot,\cdot)$, independent of t_3, t_4. We may then proceed similarly, and define

$$r_3(\phi) = E[p_3(\phi(t_3))|\phi(t_2)=\phi], \tag{5.4.84}$$

$$p_2(\phi) = Q(t_2,\phi)r_3(\phi), \tag{5.4.85}$$

$$r_2(\phi) = E[p_2(\phi(t_2))|\phi(t_1)=\phi], \tag{5.4.86}$$

$$p_1(\phi) = Q(t_1,\phi)r_2(\phi) \ . \tag{5.4.87}$$

Repeating the steps from (5.4.78) to (5.4.82) a few more times, we conclude that

$$|p_1(\phi)-p_1(\phi')| \leq P(||\phi||,||\phi'||)\lambda^{t_4-t_1} \tag{5.4.88}$$

for some polynomial P independent of t_1, t_2, t_3, t_4. Consequently,

$$E[p_1(\phi(t_1))] \leq A\lambda^{t_4-t_1}, \tag{5.4.89}$$

for some $A \geq 0$. On the other hand, note that

$$\begin{aligned} E[p_1(\phi(t_1))] &= E[Q(t_1,\phi(t_1))\cdot E[Q(t_2,\phi(t_2))\cdot E[Q(t_3,\phi(t_3)) \cdot \\ &\quad \cdot E[Q(t_4,\phi(t_4))|\phi(t_3)]|\phi(t_2)]|\phi(t_1)]] = \\ &= E[Q(t_1,\phi(t_1))Q(t_2,\phi(t_2))Q(t_3,\phi(t_3))Q(t_4,\phi(t_4))], \end{aligned} \tag{5.4.90}$$

which completes the proof of (5.4.76). Therefore, Lemma 5.4.2 applies to $R_\ell(k|t)$ and proves (5.4.66).

e) This follows from parts (c),(d), together with (5.4.21), (5.4.22).□

<u>Lemma 5.4.4</u>: Fix some $\omega\in\Omega$, $c>0$, $\bar{y}\in H$, $n\in\mathbb{N}$, $\rho>0$, $\Delta\geq\Delta_o$ and suppose that

$$||y(n)-\bar{y}|| \leq \rho, \tag{5.4.91}$$

$$||y(n)|| \leq C, \tag{5.4.92}$$

$$||x^i(n)|| \leq C, \quad \forall i \tag{5.4.93}$$

$$q(n,\Delta,2C) \leq \frac{1}{8} \tag{5.4.94}$$

$$F_2(n,\Delta,\bar{y}) + \varepsilon_2(n,\Delta,\bar{y}) + 7Cq(n,\Delta,2C) \leq C . \tag{5.4.95}$$

Then,

$$||y(m(n,\Delta)+1) - \bar{y} - F(n,\Delta,\bar{y})|| \leq$$

$$\leq 3\rho + 16q(n,\Delta,2C)G(n,\Delta,\bar{y}) + \varepsilon(n,\Delta,\bar{y}) +$$

$$+ 32[q(n,\Delta_0,2C) + q(n,\Delta,2C)(d+q(n,\Delta,2C))]D(n), \tag{5.4.96}$$

where d, Δ_0 are the constants of Assumption 5.4.7. (The exact values of the numerical constants appearing in (5.4.94)-(5.4.96) are not important).

<u>Proof of Lemma 5.4.4</u>: From equation (5.4.4) we obtain

$$y(m(n,\Delta)+1) = y(n) + \sum_{k=n}^{m(n,\Delta)} \frac{1}{k} \sum_{i=1}^{M} \Phi^i(k)Q^i(k,x^i(k),\phi(k)) \tag{5.4.97}$$

which may be rewritten as

$$y(m(n,\Delta)+1) = \bar{y}+(y(n)-\bar{y}) + \sum_{k=n}^{m(n,\Delta)} \frac{1}{k} f(k,\bar{y}) +$$

$$+ \sum_{k=n}^{m(n,\Delta)} \frac{1}{k} \sum_{i=1}^{M} \Phi^i(k)[Q^i(k,\bar{y},\phi(k))-f^i(k,\bar{y})] +$$

$$\sum_{k=n}^{m(n,\Delta)} \frac{1}{k} \sum_{i=1}^{M} \Phi^i(k)[Q^i(k,x^i(k),\phi(k))-Q^i(k,\bar{y},\phi(k))] \tag{5.4.98}$$

Note that

$$||x^i(k)-\bar{y}|| \leq ||x^i(k)-y(k)||+||y(k)-\bar{y}|| \leq D(k) + P(k), \tag{5.4.99}$$

where the last inequality follows from (5.4.43). We also use inequality (5.4.6) and $||\Phi^i(k)|| \leq 1$ to conclude that the last sum in inequality (5.4.98) is bounded (in norm) by

$$\sum_{k=n}^{m(n,\Delta)} \frac{1}{k} K_1(\phi(k),C(k))[P(k)+D(k)] \tag{5.4.100}$$

We will now proceed to obtain bounds for $P(k)$, $D(k)$ in terms of the quantities introduced in the statement of the Lemma. First note that

$$D(k) = \max_{i,j} ||x^i(k)-x^j(k)|| \leq 2\max_i ||x^i(k)|| \leq 2C(k), \quad \forall k \tag{5.4.101}$$

$$P(k) = ||y(k)-\bar{y}|| \leq ||y(k)||+||\bar{y}|| \leq C(k) + C \tag{5.4.102}$$

Using equation (5.4.2) and (5.4.46), (with $n \leq k \leq m(n,\Delta)+1$)

$$\begin{aligned}
||x^i(k)|| &= ||\sum_{j=1}^{M} \Phi^{ij}(k|n-1)x^j(n) + \sum_{t=n}^{k-1} \frac{1}{t} \sum_{j=1}^{M} \Phi^{ij}(k|t)Q^j(t,x^j(t),\phi(t))|| \leq \\
&\leq ||\sum_{j=1}^{M} \Phi^{ij}(k|n-1)x^j(n)|| + ||\sum_{t=n}^{k-1} \frac{1}{t} \sum_{j=1}^{M} \Phi^{ij}(k|t)f^j(t,\bar{y})|| + \\
&+ ||\sum_{t=n}^{k-1} \frac{1}{t} \sum_{j=1}^{M} \Phi^{ij}(k|t)[Q^j(t,\bar{y},\phi(t))-f^j(t,\bar{y})]|| + \\
&+ ||\sum_{t=n}^{k-1} \frac{1}{t} \sum_{j=1}^{M} \Phi^{ij}(k|t)[Q^j(t,\bar{y},\phi(t)) - Q^j(t,x^j(t),\phi(t))]|| \leq \\
&\leq C(n) + F_2(n,\Delta,\bar{y}) + \varepsilon_2(n,\Delta,\bar{y}) + \sum_{t=n}^{k-1} \frac{1}{t} K_1(\phi(t),C(t))[P(t)+D(t)] .
\end{aligned} \tag{5.4.103}$$

We will use (5.4.103) to prove inductively that

$$C(k) \leq 2C, \quad k \in [n,m(n,\Delta)+1] . \tag{5.4.104}$$

Indeed, suppose that (5.4.104) holds for all k such that $n \leq k \leq \bar{k}$, with $\bar{k} \leq m(n,\Delta)$. We start with inequality (5.4.103) and use (5.4.101),(5.4.102) and the assumption (5.4.95) of the Lemma to obtain

$$C(\bar{k}+1) \leq C+F_2(n,\Delta,\bar{y}) + \varepsilon_2(n,\Delta,\bar{y}) + \sum_{t=n}^{k-1} \frac{1}{t} K_1(\phi(t),2C)[3C+4C] \leq$$

$$\leq C+F_2(n,\Delta,\bar{y}) + \varepsilon_2(n,\Delta,\bar{y}) + 7Cq(n,\Delta,2C) \leq 2C \tag{5.4.105}$$

which completes the inductive proof of (5.4.104).

<u>Bounds for P(k)</u>: Consider equation (5.4.98) again, but let the upper limit in the summation be t, instead of $m(n,\Delta)$, where t satisfies $n \leq t \leq m(n,\Delta)$. Then,

$$P(t+1) \leq \rho + \left|\left| \sum_{k=n}^{t} \frac{1}{k} f(k,\bar{y}) \right|\right| +$$

$$+ \left|\left| \sum_{k=n}^{t} \frac{1}{k} \sum_{i=1}^{M} \Phi^i(k)[Q^i(k,\bar{y},\phi(k)) - f^i(k,\bar{y})] \right|\right| +$$

$$+ \sum_{k=n}^{t} \frac{1}{k} K_1(\phi(k),2C)[D(k)+P(k)] \ . \tag{5.4.106}$$

Let

$$P^* = \max_{n \leq k \leq m(n,\Delta)} P(k) \tag{5.4.107}$$

$$D^* = \max_{n \leq k \leq m(n,\Delta)} D(k), \tag{5.4.108}$$

and use the notation (5.4.39),(5.4.36),(5.4.34) to obtain

$$P^* \leq \rho + F_1(n,\Delta,\bar{y}) + \varepsilon_1(n,\Delta,\bar{y}) + q(n,\Delta,2C)[P^*+D^*] \tag{5.4.109}$$

<u>Bounds for D(k)</u>: Using (5.4.2),(5.4.46),

$$||x^{i_1}(k) - x^{i_2}(k)|| \leq \left|\left| \sum_{j=1}^{M} [\Phi^{i_1 j}(k|n-1) - \Phi^{i_2 j}(k|n-1)]\, x^j(n) \right|\right| +$$

$$+ \left|\left| \sum_{t=n}^{k-1} \frac{1}{t} \sum_{j=1}^{M} [\Phi^{i_1 j}(k|t) - \Phi^{i_2 j}(k|t)] Q^j(t,x^j(t),\phi(t)) \right|\right| \leq$$

$$\leq D(n) + 2F_2(n,\Delta,\bar{y}) + 2\varepsilon_2(n,\Delta,\bar{y}) + 2q(n,\Delta,2C)[D^*+P^*] \ . \tag{5.4.110}$$

Suppose now that $m(n,\Delta_0)\leq k\leq m(n,\Delta)+1$. We use (5.4.23) and (5.4.32) to strengthen (5.4.110):

$$D(k)\leq dD(n) + 2F_2(n,\Delta,\bar{y}) + 2\varepsilon_2(n,\Delta,\bar{y}) + 2q(n,\Delta,2C)[D^*+P^*] \ . \tag{5.4.111}$$

If we now add (5.4.109) and (5.4.110), use (5.4.94) to eliminate $q(n,\Delta,2C)$ and bring (D^*+P^*) to the left-hand side, we obtain

$$P^* + D^* \leq 4[D(n)+\rho+G(n,\Delta,\bar{y})] \tag{5.4.112}$$

Suppose once more that $m(n,\Delta_0) \leq k\leq m(n,\Delta)+1$ and use (5.4.112) in (5.4.111):

$$D(k)\leq(d+8q(n,\Delta,2C))D(n) + \rho + 3G(n,\Delta,\bar{y}) \tag{5.4.113}$$

Similarly, using (5.4.112) in (5.4.109):

$$P^* \leq 2\rho + 2G(n,\Delta,\bar{y}) + 8q(n,\Delta,\bar{y})D(n) \tag{5.4.114}$$

We are finally in a position to bound (5.4.100):

$$\sum_{k=n}^{m(n,\Delta)} \frac{1}{k} K_1(\phi(k),C(k))[P(k)+D(k)]\leq$$

$$\leq \sum_{k=n}^{m(n,\Delta_0)} \frac{1}{k} K_1(\phi(k),2C)[P^*+D^*]+ \sum_{k=m(n,\Delta_0)}^{m(n,\Delta)} \frac{1}{k} K_1(\phi(k),2C)[P^*+D(k)]$$

$$\leq 4q(n,\Delta_0,2C)[D(n)+\rho+G(n,\Delta,\bar{y})]$$

$$+ q(n,\Delta,2C)[3\rho+5G(n,\Delta,\bar{y})+(d+16q(n,\Delta,2C))D(n)] . \tag{5.4.115}$$

Using (5.4.115) to bound the terms in (5.4.98) we obtain

$$||y(m(n,\Delta)+1) - \bar{y} - F(n,\Delta,\bar{y})|| \leq$$

$$\leq \rho+\epsilon(n,\Delta,\bar{y}) + \sum_{k=n}^{m(n,\Delta)} \frac{1}{k} K_1(\phi(k),C(k))[P(k)+D(k)] \leq$$

$$\leq 3\rho+\varepsilon(n,\Delta,\bar{y}) + 9q(n,\Delta,2C)G(n,\Delta,\bar{y}) +$$

$$+ 16[q(n,\Delta_0,2C) + q(n,\Delta,2C)(d+q(n,\Delta,2C))]D(n). \square \tag{5.4.116}$$

<u>Lemma 5.4.5</u>: Let H_0 be a countable dense subset of H. Given $C_2>0$, there exist $\Delta_0>0$, $\Delta>0$, $d\in[0,1)$ and a subset Ω' of Ω (of measure 1), such that for any $a\in[0,1]$ $\bar{y}\in H_0$ (satisfying $||\bar{y}|| \leq C_2$) and for all $\omega\in\Omega'$ there exists N_0 so that the following is true:

If $n\geq N_0$, $||y(n)-\bar{y}|| \leq a\Delta^2$, $||x^i(n)|| \leq C_2$ and

$$||\frac{\partial V}{\partial y}(\bar{y})|| + D(n) \geq a , \tag{5.4.117}$$

then

$$V(y(m(n,\Delta)+1) + [D(m(n,\Delta)+1)]^2 \leq V(\bar{y}) + D^2(n) - \min\{1,A_1\Delta\}\frac{a^2}{4} \tag{5.4.118}$$

where A_1 is the constant of Assumption 5.4.5.

<u>Proof</u>: Let H_0, $C_2>0$ be given and suppose, for the time being, that $\Delta>0$ has been fixed so that $\Delta<1$ and so that the assumptions (5.4.94), (5.4.95) of Lemma 5.4.4 are satisfied, (with $C=C_2$) for n large enough. Let $\Delta_0=\Delta^2$, $d=\Delta$.

From Lemma 5.4.3, we obtain results on certain limits, which, for any $y\in H$, are valid, except possibly in a nullset. Now, since H_0 is countable, there exists some $\Omega'\subset\Omega$ of measure one, such that equations (5.4.63)-(5.4.67) are valid

for all $\bar{y}\in H_o, \omega\in\Omega'$. Suppose now that $a>0$, $y\in H_0, (||\bar{y}||\leq C_2)$ and $\omega\in\Omega'$ have been fixed. Let $A(2C_2)$, A' be the constants of Lemma 5.4.3, let A_1 be the constant of Assumption 5.4.5, let $a>0$ and define ρ,δ by:

$$\rho = a\Delta^2, \tag{5.4.119}$$

$$\delta = \min\{1,A_1\}\ \frac{a^2}{4}\ . \tag{5.4.120}$$

Using the short-hand notation

$$U = ||\frac{\partial V}{\partial y}\ (\bar{y})||, \tag{5.4.121}$$

let N_0 be large enough so that, for any $n\geq N_o$,

$$G(n,\Delta,\bar{y})\leq A'U\Delta + a^2\Delta^2 \tag{5.4.122}$$

$$q(n,\Delta,2C_2)\leq A(2C_2)\Delta \tag{5.4.123}$$

$$q(n,\Delta_0,2C_2)\leq A(2C_2)\Delta_0 = A(2C_2)\Delta^2 \tag{5.4.124}$$

$$\tilde{\varepsilon}(n,\Delta,\bar{y})\leq a\Delta^2 \tag{5.4.125}$$

$$F(n,\Delta,\bar{y})\ \cdot\ \frac{\partial V}{\partial y}\ (\bar{y})\leq -\ A_1U^2\Delta + \Delta^2 a\ . \tag{5.4.126}$$

Such a N_0 exists by Lemma 5.4.3, for the first four inequalities, and by Assumption 5.4.5, for the last one. Let $B=\max\{A',A(2C_2),1\}$.

Using Lemma 5.4.4 and (5.4.122)-(5.4.126),

$$||y(m(n,\Delta)+1)-\bar{y}-F(n,\Delta,\bar{y})||\leq 3a\Delta^2 + 16B\Delta[BU\Delta+a\Delta^2] + a\Delta^2 +$$

$$+ 32[B\Delta^2+B\Delta(\Delta+B\Delta)]D(n) \tag{5.4.127}$$

Using the inequality (5.4.117) to eliminate a, (5.4.127) simplifies to

$$||y(m(n,\Delta)+1)-\bar{y}-F(n,\Delta,\bar{y})|| \leq M_1(B)U\Delta^2 + M_1(B)D(n)\Delta^2 \tag{5.4.128}$$

where $M_1(B)$ is some constant depending only on B.

We also use (5.4.113):

$$D(m(n,\Delta)+1) \leq (\Delta+8B\Delta)D(n) + a\Delta^2 + 2BU\Delta + 2a\Delta^2 \leq$$

$$\leq M_2(B)\Delta(D(n)+U) \tag{5.4.129}$$

where $M_2(B)$ is a constant depending only on B. Let

$$M_3 = \max_{|y|\leq 2C} \left\{ ||\frac{\partial V}{\partial y}(y)||, ||\frac{\partial^2 V}{\partial y^2}(y)|| \right\} \tag{5.4.130}$$

and use (5.4.128) in a second order series expansion for V:

$$V(y(m(n,\Delta)+1)) \leq V(\bar{y}) + F(n,\Delta,\bar{y}) \cdot \frac{\partial V}{\partial y}(\bar{y}) + M_3 M_1(B)U\Delta^2(U+D(n)) +$$

$$+ \frac{M_3}{2} 16[B^2U^2\Delta^2 + a^2\Delta^4 + M_1^2(B)U^2\Delta^4 + M_1^2(B)U^2D^2(n)\Delta^2] \leq$$

$$\leq V(\bar{y}) - A_1U^2\Delta + M_4(B)\Delta^2(U^2+D^2(n)), \tag{5.4.131}$$

for some new constant $M_4(B)$. We also square (5.4.129) to obtain

$$[D(m(n,\Delta)+1)]^2 \leq M_5(B)\Delta^2(U^2+D^2(n)) =$$

$$= D^2(n) - (1-M_5(B)\Delta^2)D^2(n) + M_5(B)\Delta^2U^2 . \tag{5.4.132}$$

Adding (5.4.131) and (5.4.132), we have

$$V(y(m(n,\Delta)+1) + [D(m(n,\Delta)+1)]^2 \leq V(\bar{y}) + D^2(n) -$$

$$-A_1U^2\Delta - (1-M_5(B)\Delta^2)D^2(n) + (M_4(B)+M_5(B))\Delta^2(U^2+D^2(n)). \tag{5.4.133}$$

Clearly, if Δ has been chosen so that

$$1 - M_5(B)\Delta^2 \geq \frac{3}{4} \tag{5.4.134}$$

$$(M_4(B)+M_5(B))\Delta^2 \leq \frac{1}{4}\ , \tag{5.4.135}$$

$$(M_4(B)+M_5(B))\Delta \leq \frac{A_1}{2}\ , \tag{5.4.136}$$

then

$$V(y(m(n,\Delta)+1)) + [D(m(n,\Delta)+1)]^2 \leq V(\bar{y}) + D^2(n) - \frac{A_1}{2} U^2\Delta - \frac{1}{2} D^2(n) \tag{5.4.137}$$

Now note that $U^2 + D^2(n) \geq \frac{a^2}{2}$ which shows that inequality (5.4.118) is satisfied. Moreover, note that Δ was selected only as a function of B, which in turn depended only on C_2, as required. □

<u>Lemma 5.4.6</u>: Under the assumptions of the Theorem,

$$\liminf_{n\to\infty} [V(y(n))+D^2(n)]=0, \tag{5.4.138}$$

almost surely.

<u>Proof of Lemma 5.4.6</u>: Let $C<\infty$ be the constant appearing in the statement of the Theorem. Let

$$C_1 = \max_{|\bar{y}|\leq C} V(\bar{y}) + 4C^2 + 1 \tag{5.4.139}$$

and note that if $||x^i(n)||\leq C$, $\forall i$, then $V(y(n)) + D^2(n)<C_1$. Then pick a new constant C_2 such that $\inf_{||\bar{y}||\geq C_2} V(\bar{y})\geq C_1$. Such a constant exists, by (5.4.19).

Let H_0 be a countable dense subset of H and choose Δ_0, Δ, d,Ω', as in Lemma 5.4.5.

Fix some $\omega\in\Omega'$ and assume, to derive a contradiction, that

$$\liminf_{n\to\infty} [V(y(n))+D^2(n)] = b>0 \ . \tag{5.4.140}$$

We also define

$$a'= \frac{1}{3} \liminf_{n\to\infty} [||\frac{\partial V}{\partial y} (y(n))||+D(n)] \ . \tag{5.4.141}$$

Using Assumption 5.4.5 and (5.4.140) we conclude that $a>0$. Also note that $b<C_1$ and let $a=\min\{1,a'\}$.

By a standard compactness argument, we can see that there exists some $\bar{y}\in H_0$ and a sequence n_k of integers such that:

$$||\bar{y}||\leq C_2 \ , \ ||x^i(n_k)||\leq C_2 \ , \ \forall i,k \tag{5.4.142}$$

$$||y(n_k)-\bar{y}||\leq a\Delta^2, \quad \forall k, \tag{5.4.143}$$

$$||\frac{\partial V}{\partial y} (y(n_k))- \frac{\partial V}{\partial y} (\bar{y})||\leq a, \quad \forall k, \tag{5.4.144}$$

$$||V(y(n_k))-V(\bar{y})||\leq \frac{1}{12} \min\{1,A_1\Delta\}a^2, \tag{5.4.145}$$

$$V(y(n_k)) + D^2(n_k)\leq b + \frac{1}{12} \min\{1,A_1\Delta\}a^2, \tag{5.4.146}$$

$$||\frac{\partial V}{\partial y} (y(n_k))||+ D(n_k)\geq 2a \ . \tag{5.4.147}$$

If n_k is large enough (larger than the constant N_0 given by Lemma 5.4.5), then (5.4.118) holds and yields:

$$V(y(m(n_k,\Delta)+1)) + [D(m(n_k,\Delta)+1)]^2 \leq V(\bar{y}) + D^2(n_k) -$$

$$- \min\{1,A_1\Delta\}\frac{a^2}{4} \leq$$

$$\leq V(y(n_k)) + D^2(n_k) + \left(\frac{1}{12} - \frac{1}{4}\right)\min\{1,A_1\Delta\}\, a^2 \leq$$

$$\leq b - \frac{1}{12}\min\{1,A_1\Delta\}\, a^2 , \tag{5.4.148}$$

thus contradicting (5.4.140).□

To complete the proof of the Theorem, we may assume that

$$\limsup_{n\to\infty} [V(y(n))+D^2(n)] = a>0 \tag{5.4.149}$$

and derive a contradiction. The argument involved is identical to the corresponding argument of Ljung [1977a] and we only provide an outline.

Given any $a'\in(0,a)$ we may choose an integer sequence $\{n_k\}$, and a vector $(\bar{y},\bar{x}^1,\ldots,\bar{x}^M)$ with $\bar{y}\in H_0$, such that $y(n_k)$ is close to $\bar{y}$, $x^i(n_k)$ is close to $\bar{x}^i$, $V(y(n_k))+D(n_k)$ converges to a' and such that, after time n_k, the quantity $V(y(n))+D(n)$ reaches the level $a-\varepsilon$ before falling below $a'-\varepsilon$, where $\varepsilon>0$ has been chosen suitably small. There are two cases to consider:

(i) If the level $a-\varepsilon$ is reached before time $m(n_k,\Delta)+1$ (where $\Delta=\sqrt{\Delta_0}$ is chosen as in Lemma 5.4.5), then provided that a' is small enough, (5.4.96) is contradicted.

(ii) If the level $a-\varepsilon$ is reached after time $m(n_k,\Delta)+1$, then Lemma 5.4.5 implies that $V(y(n))+D(n)$ should first fall below $a'-\varepsilon$, before reaching the level $a-\varepsilon$, thus contradicting our assumption.

Therefore,

$$\lim[V(y(n))+D(n)]=0, \quad \text{almost surely,} \tag{5.4.150}$$

which implies $\lim_{n\to\infty} y(n)=x^*$, $\lim_{n\to\infty} D(n)=0$, or, equivalently, $\lim_{n\to\infty} x^i(n) = x^*$, $\forall i$, almost surely. ■

Technical Remarks on the Proof of Theorem 5.4.1

The preceding proof has certain basic similarities with the proof of Theorem 1 of Ljung [1977a]. The main difference is that instead of keeping track of the evolution of one state vector only, we also need to keep track of the magnitude of the disagreement between processors, which complicates greatly the associated inequalities. So, unlike Theorem 5.4.2 that follows, the preceding proof differs from Ljung's in non-trivial ways.

Another difference is that Ljung's Theorem 1 required a returning condition on the process $\{\phi(t)\}$, in addition to the returning condition of the state of computation. It may be seen in Ljung's proof that this condition was necessary only for algorithms in which the dynamics of the process $\{\phi(t)\}$ were influenced by the state $x(t)$ of the algorithm. Such a possibility, however, was not allowed by us and the returning condition on $\{\phi(t)\}$ was not needed.

The Associated ODE

The results of Ljung [1977a] are not so useful for rigorously demonstrating convergence of stochastic algorithms; rather, by associating an ordinary differential equation (ODE) with such an algorithm, they provide a simple heuristic

method of analysis of recursive stochastic algorithms. In fact the ODE is not essential in Ljung's result. The mathematically important entity is the Lyapunov function associated with this ODE. (This is pointed out in Appendix V of [Ljung 1977a]).

For this reason, our result has been formulated exclusively in terms of the function V, without reference to an ODE. Our result translates easily to an ODE-type result if we assume that the algorithm is (asymptotically) time invariant. More specifically, assume that $\frac{1}{\Delta}\lim_{n\to\infty}\sum_{k=n}^{m(n,\Delta)}\frac{1}{k}f(k,x)$ exists, for any $\Delta>0$, and is independent of Δ. Let us denote this limit by $f(x)$, and note that, by Assumption 5.4.5, V is a Lyapunov function for the ODE $\dot{x}=f(x)$. Conversely, if the equation $\dot{x}=f(x)$ is globally asymptotically stable, we may invoke converse stability theorems to construct a function V satisfying Assumptions 5.4.5, 5.4.6. Consequently, the algorithm will converge to the point of convergence x^* of the ODE $\dot{x}=f(x)$.

Extensions

Theorem 5.4.1 is a decentralized analog of a special case of Theorem 1 of Ljung [1977a]. Modifications or generalizations of this result are possible, along the lines of Ljung. We discuss some of them below:

1) We may assume that the function V is such that inequality (5.4.20) is only satisfied on a bounded subset H_1 of H. (In the ODE language, the equation $\dot{x}=f(x)$ has a bounded domain of attraction.) Then, there exists a subset H_2 of H_1 such that, if the algorithm returns an infinite number of times to H_2, and if Δ_0, d are small enough, then the conclusion of Theorem 5.4.1 remains valid. (The proof is effectively identical with that of Theorem 5.4.1).

2) Suppose that the algorithm is asymptotically time-invariant, so that we may associate an ODE $\dot{x}=f(x)$. Then, under certain additional smoothness assumptions (see Theorem 2 of Ljung [1977a]) we may prove that the algorithm may converge <u>only</u> to stable equilibrium points of $\dot{x}=f(x)$.

The final and most important extension is to allow the dynamics of the process $\{\phi(t)\}$ to be influenced by the states of the algorithm. This extension is necessary if we want to apply the results to algorithms for adaptive control or to certain system identification applications [Ljung, 1977a]. For technical reasons, we have not proved convergence under the assumption that the algorithm returns infinitely many times to a bounded region. We use the stronger assumption that the processors come arbitrarily close to agreeing, infinitely many times. We also assume that the process of communicating and combining is faster than the algorithm. Then, whenever the processors are very close to agreeing, the algorithm behaves (up to first order) as a centralized algorithm and the centralized results may be recovered by replicating the proof of Ljung. The result that follows, although important from the point of view of applications, is not significantly different - mathematically - from Theorem 1 of Ljung. For this reason, we only give an outline of the proof. Moreover, we formulate the Assumptions and the result in the language of Ljung, so that a direct comparison may be made.

Let $z(n)$ denote $(x^i(n),\ldots,x^M(n))$, which is therefore an element of H^M. Also, for any $x\in H$, let $z(x)$ denote the M-tuple $(x,x,\ldots,x)$.

Assumption 5.4.8

a) The steps $s^i(n)$ are given by $Q^i(n,x^i(n),\phi(n))$, where the process $\{\phi(n)\}$ is generated by $\phi(0) = e(0)$ and

$$\phi(n+1) = A(z(n))\phi(n) + B(z(n))e(n) \ . \tag{5.4.151}$$

Here $\{e(n)\}$ is a sequence of independent random vectors, defined on an underlying probability space $(\Omega,\mathcal{F},\mathcal{P})$ and such that

$$\sup_n E[||\phi(n)||^p]<\infty \ , \ \forall p>1 \tag{5.4.152}$$

Let $D_s = \{z\in H^M : A(z)$ is asymptotically stable$\}$. Let $D_R \subset H$ be open and connected and let $\underline{D}_R$ denote the Cartesian product of D_R with itself, M times. We assume that $A(\cdot)$ and $B(\cdot)$ are Lipschitz continuous on $\underline{D}_R$ and that $\underline{D}_R \subset D_s$.

Let $\lambda(z)<1$ be such that

$$||A(z)^k||< C\lambda(z)^k, \ \forall k\in N \ . \tag{5.4.153}$$

For any $z\in\underline{D}_R$, $\lambda<1$, $c>0$, we define the stochastic processes $\overline{\phi}(t,z)$, $u(t,\lambda,c)$ by

$$\overline{\phi}(t,z) = A(z)\overline{\phi}(t-1,z) + B(z)e(t); \ \overline{\phi}(0,z)=0, \tag{5.4.154}$$

$$u(t,\lambda,c) = \lambda u(t-1,\lambda,c) + c|e(t)|; \ u(0,\lambda,c)=0. \tag{5.4.155}$$

(So, $\overline{\phi}(t,z)$ is the process to be obtained from (5.4.151) if $z(n)$ was "frozen" to a value z; also, if $\lambda=\lambda(z)$ and $||B(z)||\leq c$, then $u(t,\lambda,c)$ is a bound for $\overline{\phi}(t,z)$.)

Let $\mathcal{B}(x,\rho)$ denote a (closed) ball of radius ρ around some point x. We assume that, for any function ρ: $H\to(0,\infty)$,

$$||Q^i(t,x',\phi')-Q^i(t,x'',\phi'')|| \leq K_1(x,\phi,\rho(x),u)\{||x'-x''||+||\phi-\phi''||\} \tag{5.4.156}$$

for any $u \geq 0$, x', x'', ϕ', ϕ'' satisfying $x' \in \mathcal{B}(x,\rho(x)) \cap D_R$, $x'' \in \mathcal{B}(x,\rho(x)) \cap D_R$, $\phi' \in \mathcal{B}(\phi,u)$, $\phi'' \in \mathcal{B}(\phi,0)$.

Moreover, the function K_1 satisfies

$$\begin{aligned} K_1(x,\phi',\rho,u') - K_1(x,\phi'',\rho,u'') | \leq K_2(x,\phi,\rho,u,w). \\ \{||\phi-\phi''||+|u'-u''|\} \end{aligned} \tag{5.4.157}$$

whenever $\phi' \in \mathcal{B}(\phi,w)$, $\phi'' \in \mathcal{B}(\psi,w)$, $u' \in \mathcal{B}(u,w)$, $u'' \in \mathcal{B}(u,w)$.

c) For any $x \in D_R$, the random variables $Q^i(t,x,\overline{\phi}(t,z(x)))$, $K_1(x,\overline{\phi}(t,z(x)),\rho(x), u(t,\lambda,c))$ and $K_2(x,\overline{\phi}(t,z(x)),\rho(x), u(t,\lambda,c), u(t,\lambda,c))$ have bounded p-moments for all $p>1$, and all $\lambda<1$, $c<\infty$.

d) The step-size $\gamma(n)$ is equal to $1/n$. Also, communication delays are zero (or, more generally, bounded) and for any $\Delta>0$,

$$\lim_{k\to\infty} \max_{n \geq m(k,\Delta)} ||\Phi^{ij}(n|k)-\Phi^j(k)||=0 \tag{5.4.158}$$

(That is, the process of communicating and combining is faster than the natural time scale of the algorithm. This is the case, in particular, if Assumptions 5.2.1, 5.2.3, hold.)

e) $\lim_{t\to\infty} E[\Phi^i(t)Q^i(t,x,\overline{\phi}(t,z(x))]$ exists, for any $x \in D_R$ and is denoted by $f^i(x)$. We let $f(x) = \sum_{i=1}^{M} f^i(x)$.

<u>Theorem 5.4.2:</u> Let Assumption 5.4.8 hold. Let $\overline{D}$ be a compact subset of D_R such that the trajectories of the ODE

$$\dot{x}=f(x) \tag{5.4.159}$$

that start in $\bar{D}$ remain in a closed subset D' of D_R. Assume that (5.4.159) has an invariant set D_c with domain of attraction $D_A \supset \bar{D}$. Also assume that there exists a constant C_ϕ such that, for almost all $\omega \in \Omega$, there exist sequences $\{n_k\}$, $\{\varepsilon_k\}$ ($n_k \in \mathbb{N}$, $n_{k+1} > n_k$, $\varepsilon_k > 0$, $\lim_{k\to\infty} \varepsilon_k = 0$) such that $y(n_k) \in \bar{D}$, $||\phi(n_k)|| \leq C_\phi$, $D(n_k) \leq \varepsilon_k$. (Recall that $D(n_k) = \max_{i,j} ||x^i(n_k) - x^j(n_k)||$.)

Then $x(t) \to D_c$, with probability one, as $t \to \infty$.

<u>Proof (Outline, following Ljung)</u>:

<u>Step 1</u>: Fix some $\bar{y} \in D_s$. Let $\rho = \rho(\bar{y})$ be small enough and suppose that for some n we have: $y(n) \in B(\bar{y}, \rho(\bar{y}))$, $D(n) < \rho(\bar{y})$, $|\phi(n)| < C_\phi$. Fix some small enough $\Delta > 0$ and assume that n is large enough so that $\sum_{k=n}^{m(n,\Delta)} \frac{1}{k} \Phi^i(k) Q^i(k, \bar{y}, \bar{\phi}(k, z(y)) - \Delta f(\bar{y})$ can be made as small as desired. Then, provided that ρ, n, Δ have been appropriately chosen, we may conclude that $x^i(k) \in B(\bar{y}, 2\rho)$, $\forall i$, $\forall k \in [n, m(n,\Delta)]$. It then follows that $\phi(k) \approx \phi(k, z(\bar{y}))$ and, consequently, $y(m(n,\Delta)) \approx y(n) + \Delta f(\bar{y})$. Also, with n large enough, $D(m(n,\Delta))$ becomes arbitrarily small, due to (5.4.154).

<u>Step 2</u>: Let V be a Lyapunov function for the ODE (5.4.155). If the algorithm returns infinitely often to the vicinity of some $\bar{y} \in D_R$ (with $V(\bar{y}) \neq 0$) and if at those times D(n) is small and $|\phi(n)| \leq C_\phi$, we use the results of Step 1 to show that the algorithm also returns to the vicinity of some $\bar{y}' \in D_R$, with $V(\bar{y}') < V(\bar{y})$, infinitely often and that, at those times, D(n) is small and $|\phi(n)| \leq C_\phi$. Proceeding in this manner, we conclude that $\liminf_{n\to\infty} [V(y(n)) + D(n)] = 0$.

Step 3: If $\lim_{n\to\infty} V(y(n))\neq 0$, it means that $V(y(n))$ must "upcross" an interval $[a,a']$ (with $0<a<a'$) infinitely many times. Choosing a and a' small enough, a version of Ljung's argument leads to a contradiction.■

Remarks:

1. Note that Theorem 5.4.2 assumes that the algorithm returns infinitely often to an appropriate region (as in Theorem 5.4.1) but also that the disagreement at those returning times is arbitrarily small. This condition on the disagreement may be enforced if once in a while each processor communicates to every other processor and they all combine using the same coefficients; in other words, disagreement is explicitly eliminated, once in a while. More natural conditions are also possible.

2. The ODE approach cannot be used to prove global convergence of an algorithm. However, certain algorithms with correlated noise are known to converge, most notably the ELS algorithm for system identification [Solo, 1979]. However, global convergence of decentralized versions of such algorithms does not follow automatically and has to be verified by other means. Moreover, general convergence results do not seem possible; rather, small classes of algorithms must be studied separately.

3. In Theorem 5.4.2 we have assumed that the decentralized algorithm is asymptotically time invariant, so that an ODE may be attached to it. This is not necessary and the result may be stated directly in terms of Lyapunov function V. (See Assumptions 5.4.5, 5.4.6 and their discussion, as well as Appendix V of Ljung [1977a]).

4. Finally, the proof when delays are bounded (instead of zero) is the same, provided that the appropriate state augmentation has been made, and the returning condition is appropriately modified.

5.5 APPLICATIONS IN SYSTEM IDENTIFICATION

Many recursive stochastic algorithms for on-line system identification are pseudo-gradient algorithms or, more generally, recursive stochastic algorithms of the type considered by Ljung [1977a]. For a review of such algorithms and their convergence properties, the reader may refer to [Ljung, 1981; Goodwin and Payne, 1977; Astrom and Eykhoff, 1971]. Consequently, it should be expected that our results of Section 5.3 and 5.4 may be used to study the convergence of decentralized identification schemes. An interesting related issue is the problem of how to decompose (decentralize) an identification algorithm, so that the resulting scheme is a reasonable and attractive alternative to the corresponding centralized algorithm. In this section, we present and discuss a few possibilities.

Example 1: Processors Identifying Identical ARMA Models

Let $z^1(t)$, $z^2(t)$ be autoregressive moving average processes described by

$$z^i(t) + a_1 z^i(t-1) + \ldots + a_n z^i(t-n) =$$
$$= b_0 u^i(t) + \ldots + b_m u^i(t-m) + w^i(t), \quad i = 1,2, \tag{5.5.1}$$

where $w^1(t)$, $w^2(t)$ are zero-mean white noises and $u^i(t)$ is an input process known by processor P^i, $i=1,2$. Processor P^i also observes at time t the output $z^i(t)$ and tries to estimate the parameters a_k and b_k of the process. Note that we are assuming that the parameters a_k and b_k are the same for both processes, that is, they do not depend on i. We may let q be the backward shift operator and rewrite (5.5.1) in transform notation:

$$A(q)z^i(t) = B(q)u^i(t) + w^i(t), \tag{5.5.2}$$

where

$$A(q) = 1 + a_1 q + \ldots + a_n q^n, \tag{5.5.3}$$

$$B(q) = b_0 + b_1 q + \ldots + b_m q^m. \tag{5.5.4}$$

The two processors are to cooperate, by exchanging messages, in identifying the unknown parameters. (see Figure 5.5.1). An interesting special case of the above configuration is depicted in Figure 5.5.2, in which $A(q) = 1$ and the input processes $u^1(t)$, $u^2(t)$ coincide. This is the case of two processors obtaining noisy observations of the output of the same (moving average) process.

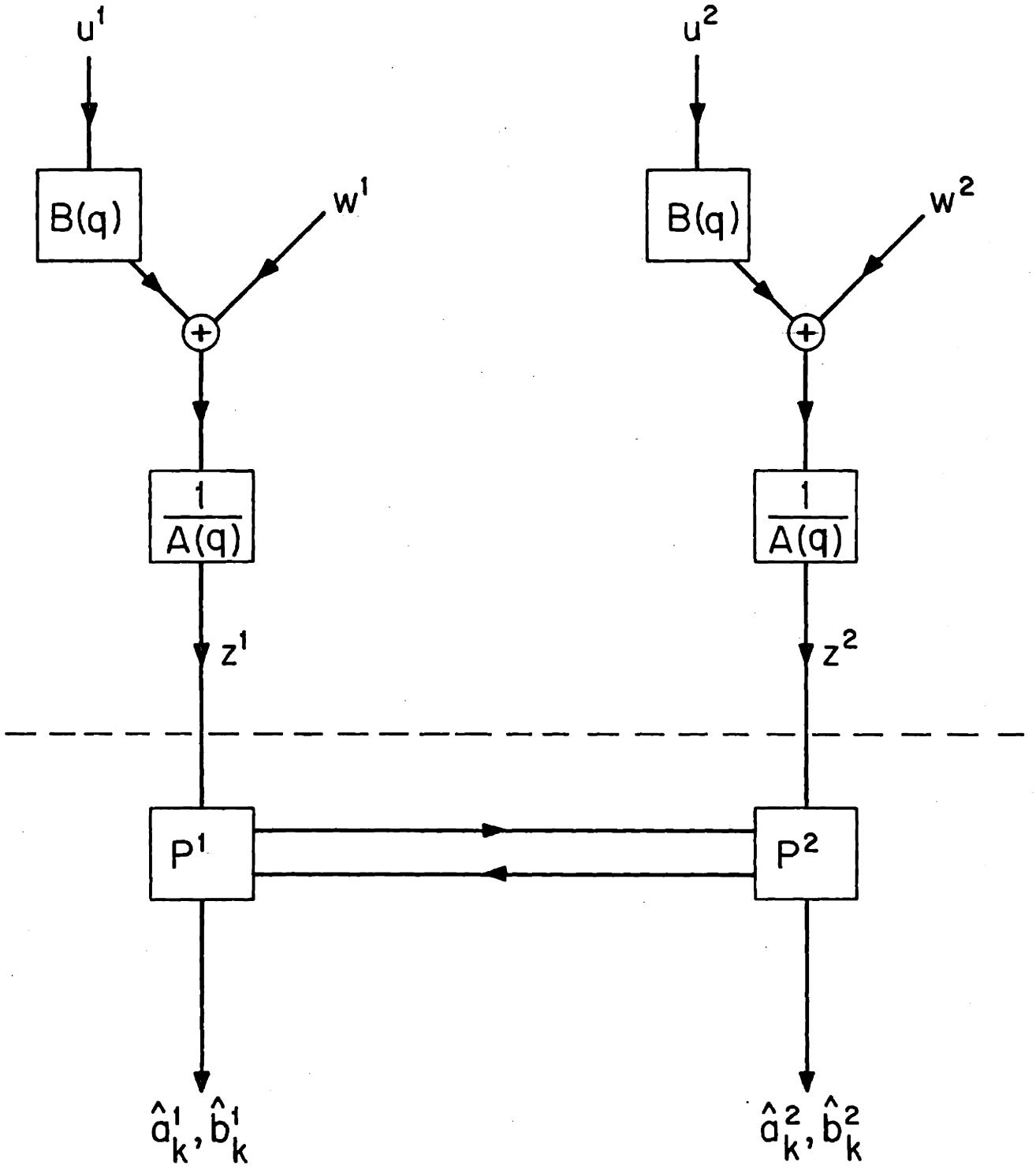

Figure 5.5.1: Two Processors Identifying Identical ARMA Systems.

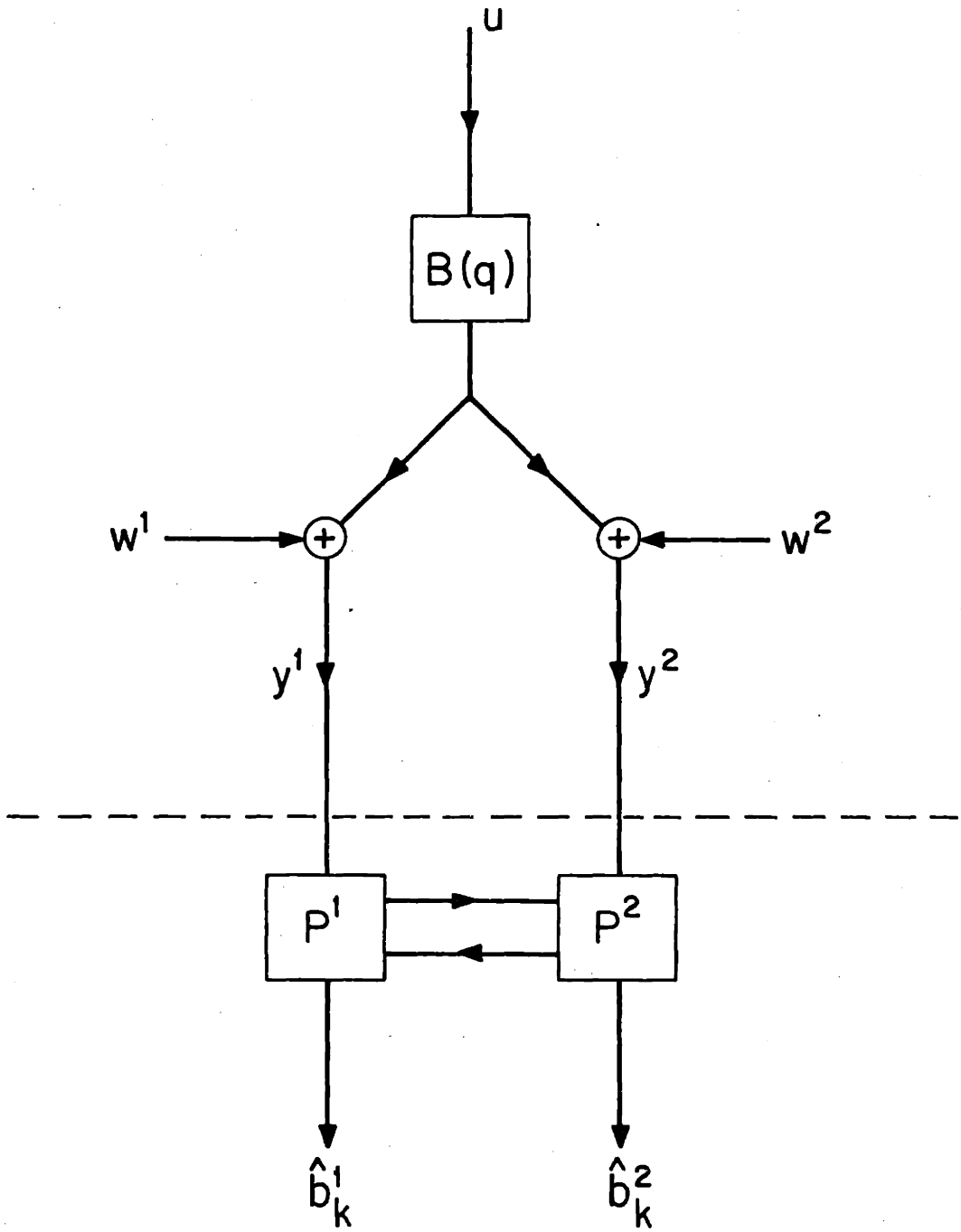

Figure 5.5.2: Two Processors Identifying the same Moving Average System.

The first possibility in the above setting is to let each processor use any of the standard ARMA identification algorithms, in isolation, without exchanging any messages at all. Then (provided that certain identifiability conditions are satisfied [Astrom and Eykhoff, 1971]), each processor will, in the limit, recover the true values of the parameters.

It is clear, however, that better estimates may be constructed (i.e. convergence may be faster) if the two processors cooperate by exchanging some information. Moreover, it is possible that none of the inputs u^1, u^2 is rich enough to allow either processor to identify all parameters alone, but that the parameters may be accurately identified on the basis of both sets of measurements.

A trivial alternative is to let processor P^1 transmit all its measurements to processor P^2, as they are being observed, and then have processor P^2 compute a "centralized" estimate. This would require, however, an excessive amount of communications.

Another alternative would be to proceed along the lines of Chapter 4, whereby at each instant of time each processor computes an estimate which is optimal given its information, including the messages it has received. This would be, however, computationally hard, in general: even if all random variables are Gaussian, processor P^1 would be faced with a <u>nonlinear</u> estimation problem unless it could directly measure u^2. (This is because the estimate of processor P^2 at any time is a nonlinear function of u^2.) If the inputs were commonly observed, then each processor would face a linear problem at each time (this is the LQG

setting of Section 4.4), but still not an easy one, except for the very special case in which communication delays are zero and the noises w^1, w^2 are independent. In the latter case, the results of Willsky et al. [1982] could provide us with an optimal updating rule.

So, there is a clear trade-off between optimality and the complexity of the scheme. An approach that lies in the middle, and which we will consider, is to let the processors combine their own estimates with the messages received (estimates of the other processor) by simply forming a convex combination, according to the model proposed in Section 5.2.

We now continue with a more detailed discussion. Let

$$x^* = (a_1, a_2, \ldots, a_n; b_0, b_1, \ldots, b_m)^T \tag{5.5.5}$$

be the vector unknown parameters and let

$$\psi^i(t) = (-z^i(t-1), \ldots, -z^i(t-n); u^i(t), \ldots, u^i(t-m))^T, \quad i=1,2 . \tag{5.5.6}$$

Then, (5.5.1) becomes

$$z^i(t) = \psi^i(t)^T x^* + w^i(t), \; i=1,2, \tag{5.5.7}$$

and $\psi^i(t)$ is a vector which is known by processor i at time t.

The simplest algorithm to be considered is the LMS algorithm [Ljung, 1981]: in the absence of any communications, processor i forms the estimate $\hat{x}^i(t)$ recursively, as follows:

$$\hat{x}^i(t+1) = \hat{x}^i(t) + \frac{1}{t}\psi^i(t)(z^i(t) - \psi^i(t)^T\hat{x}^i(t)) \; ; \tag{5.5.7}$$

in the notation of Section 5.2, $\gamma^i(t) = 1/t$ and

$$s^i(t) = \psi^i(t)(z^i(t) - \psi^i(t)^T\hat{x}^i(t)) =$$

$$= \psi^i(t)\psi^i(t)^T(x^* - \hat{x}^i(t)) + \psi^i(t)w^i(t). \tag{5.5.8}$$

We also consider the NLMS algorithm:

$$\hat{x}^i(t+1) = \hat{x}^i(t) + \frac{1}{t}\frac{\psi^i(t)}{||\psi^i(t)||^2 + \varepsilon}(z^i(t)-\psi^i(t)^T\hat{x}^i(t)) \tag{5.5.9}$$

for which

$$s^i(t) = \frac{\psi^i(t)\psi^i(t)^T}{||\psi^i(t)||^2 + \varepsilon}(x^* - \hat{x}^i(t)) + \frac{\psi^i(t)}{||\psi^i(t)||^2 + \varepsilon}w^i(t) \tag{5.5.10}$$

where $\varepsilon>0$ is some constant introduced to avoid division by zero.

Let F_t be the σ-field generated by $\{\psi^i(0),\ldots,\psi^i(t);\ i=1,2\}$, and introduce the cost function

$$J(x) = \frac{1}{2}(x-x^*)^T(x-x^*) \tag{5.5.11}$$

We assume that $E[w^i(t)|F_t] = 0$, $i=1,2$; so $w^i(t)$ is uncorrelated with past values of w^i, as well with past values of u^i. We then have, for the LMS algorithm:

$$E[<\frac{\partial J}{\partial x}(\hat{x}^i(t)),\ s^i(t)>|F_t] =$$

$$= (\hat{x}^i(t)-x^*)^T E[\psi^i(t)\psi^i(t)^T(x^* - \hat{x}^i(t)) + \psi^i(t)w^i(t)|F_t] =$$

$$= -(\hat{x}^i(t)-x^*)^T\psi^i(t)\psi^i(t)^T(\hat{x}^i(t)-x^*) \leq 0 \tag{5.5.12}$$

Similarly, for the NLMS algorithm,

$$E[<\frac{\partial J}{\partial x}(\hat{x}^i(t)),\ s^i(t)>|F_t] =$$

$$= -(\hat{x}^i(t)-x^*)^T \frac{\psi^i(t)\psi^i(t)^T}{||\psi^i(t)||^2 + \varepsilon} (\hat{x}^i(t)-x^*) \quad , \qquad (5.5.13)$$

so that, for either algorithm, the pseudogradient Assumption 5.3.3 is satisfied.

Moreover, in either of the following two cases:

(i) For the LMS algorithm and with $\psi^i(t)$ a process whose sample paths are bounded by some A>0 (so that sample paths of $u^i(t)$ and $w^i(t)$ are also bounded)*,

(ii) For the NLMS algorithm and with $w^i(t)$ a process with finite second moment. It is easy to see that

$$E[||s^i(t)||^2] \leq AE[J(\hat{x}^i(t))] + B, \quad \forall i,t \qquad (5.5.14)$$

for some constants A,B, so that Assumption 5.3.5 is also satisfied.

Concerning the process of communications, we assume Assumptions 5.2.1, 5.2.3 and 5.2.4. We are also assuming that all components of $\hat{x}^i$ are equally weighted when combining the estimates of two processors; therefore, the matrices $\Phi^{ij}(k|t)$ and $\Phi^j(t)$ of Section 5.2 are actually scalars.

It follows from Theorem 5.3.2 that

$$\sum_{t=1}^{\infty} \frac{1}{t} \sum_{i=1}^{2} \Phi^i(t)(\hat{x}^i(t) - x^*)^T Q^i(t)(\hat{x}^i(t) - x^*) < \infty \qquad (5.5.14)$$

almost surely, where $Q^i(t)$ is defined as follows:

(i) (LMS) $Q^i(t) = \psi^i(t)\psi^i(t)^T$ (5.5.15)

* Naturally, we assume that the system (5.5.1) is stable.

(ii) (NLMS) $$Q^i(t) = \frac{\psi^i(t)\psi^i(t)^T}{||\psi^i(t)||^2+\varepsilon} \tag{5.5.16}$$

We also know from Theorem 5.3.2 that $\hat{x}^1(t) - \hat{x}^2(t)$ converges to zero and that $(\hat{x}^i(t) - x^*)^T(\hat{x}^i(t) - x^*)$ converges to some value, almost surely, for each i. However, the preceding do not imply yet that $\hat{x}^i(t)$ converges to x^*. For this we need some identifiability condition; that is, the inputs $u^i(t)$ must be sufficiently rich. As an example, we present a convergence result, very similar to results for centralized identification; in which we also summarize the preceding discussion.

<u>Theorem 5.5.1</u>: Let the processes $z^i(t)$ be obtained from (5.5.1) and assume that

$$E[w^i(t)|F_t] = 0, \quad \sup_t E[|w^i(t)|^2] < \infty \tag{5.5.17}$$

Let Assumptions 5.2.1, 5.2.3, 5.2.4 be satisfied. Consider either the LMS algorithm (with $||\psi^i(t)||$ bounded by some constant) or the NLMS algorithm and assume that there exist constants $a \in \mathbb{N}, \beta > 0$ such that

$$E[\sum_{t=k}^{k+a} \sum_{i=1}^{2} Q^i(t)|F_k] > \beta I, \quad \forall k \in \mathbb{N}, \text{a.s.} \tag{5.5.18}$$

where I is the identity matrix. Then,

$$\lim_{t\to\infty} \hat{x}^i(t) = x^*, \quad i=1,2, \tag{5.5.19}$$

almost surely.

<u>Proof</u>: From the proof of Theorem 5.3.2, we obtain

$$\lim_{k\to\infty} \max_{k\le t\le k+a} E[||\hat{x}^i(k) - \hat{x}^j(t)||^2] = 0, \quad i,j=1,2 \tag{5.5.20}$$

Moreover, Theorem 5.3.2c yields

$$\liminf_{t\to\infty} E[(\hat{x}^i(t)-x^*)^T Q^i(t)(\hat{x}^i(t)-x^*)]=0, \qquad i=1,2 \tag{5.5.21}$$

Combining (5.5.20) with (5.5.21) and using the fact that $Q^i(t)$ is bounded, we obtain

$$\liminf_{k\to\infty} \max_{k\le t\le k+a} E[(\hat{x}^i(k)-x^*)^T \Phi^j(t) Q^j(t)(\hat{x}^i(k)-x^*)] = 0 \tag{5.5.21}$$

and, consequently,

$$\liminf_{k\to\infty} \sum_{j=1}^{2} \sum_{t=k}^{k+a} E[(\hat{x}^i(k)-x^*)^T \quad Q^j(t)(\hat{x}^i(k)-x^*)] = 0 \tag{5.5.22}$$

and, using Fatou's Lemma,

$$\liminf_{k\to\infty} (\hat{x}^i(k)-x^*)^T \; E\left[\sum_{j=1}^{2} \sum_{t=k}^{k+a} Q^j(t) \,|\, F_k\right] (\hat{x}^i(k)-x^*)=0 \;, \tag{5.5.23}$$

which, in view of (5.5.18) implies that

$$\liminf_{k\to\infty} ||\hat{x}^i(k) - x^*||^2 = 0. \tag{5.5.24}$$

On the other hand, by Theorem 5.2.3a, $||\hat{x}^i(k)-x^*||^2$ converges, which shows that $\hat{x}^i(k)$ converges to x^*, almost surely.■

We now continue with the same example and consider the RLS (recursive least squares algorithm). In this algorithm each processor evaluates recursively (in the absence of communications) an auxiliary quantity (a matrix) $R^i(t)$ [Ljung, 1981]:

$$R^i(t+1) = R^i(t) + \frac{1}{t}\,[\psi^i(t)\psi^i(t)^T - R^i(t)] \tag{5.5.25}$$

and updates its estimate according to

$$\hat{x}^i(t+1) = \hat{x}^i(t) + \frac{1}{t}[R^i(t+1)]^{-1}\psi^i(t)[z^i(t) - \psi^i(t)^T\hat{x}^i(t)] \ . \qquad (5.5.26)$$

This algorithm is not a pseudo-gradient algorithm, at least when the cost function is $J(x) = ||x-x^*||^2$, because $[R^i(t+1)]^{-1}\psi^i(t)\psi^i(t)^T$ is not necessarily nonnegative definite, even though it is the product of two non-negative definite matrices. It may be analyzed, however, via the ODE approach as in Ljung [1977a]. Let us assume the following:

(i) Assumptions 5.2.1, 5.2.3, 5.2.4. (The time scale of the combining process is faster than the natural time scale of the algorithm.)

(ii) All entries of R and all components of $\hat{x}$ are combined using the same weights, so that $\Phi^i(t)$ is actually a scalar for each i,t. Moreover, $\Phi^i = \lim_{t\to\infty} \Phi^i(t)$ exists, for i=1,2.

(iii) For each i, the input $u^i(t)$ and the noise $w^i(t)$ are stationary and independent stochastic processes. In particular, $w^i(t)$ is zero-mean, white, with all moments finite. Also, $u^i(t)$ is the output of a linear time invariant, asymptotically stable system, driven by some i.i.d. noise $e^i(t)$ which has finite moments.

Let

$$f^i(x) = E[\psi^i(t)(y^i(t) - \psi^i(t)^T x)] \qquad (5.5.27)$$

$$G^i = E[\psi^i(t)\psi^i(t)^T] \qquad (5.5.28)$$

(The expectations do not depend on t, because of stationarity.)

For the case of only one processor, the associated ODE was found by Ljung to be:

$$\dot{x}(t) = R^{-1}(t)f(x(t)) \tag{5.5.29a}$$

$$\dot{R}(t) = G(x(t)) - R(t) \tag{5.5.29b}$$

It follows that the ODE associated to the decentralized algorithm with two processors is:

$$\dot{x}(t) = R^{-1}(t)(\Phi^1 f^1(x(t)) + \Phi^2 f^2(x(t))) \tag{5.5.30a}$$

$$\dot{R}(t) = \Phi^1 G^1 + \Phi^2 G^2 - R(t) \tag{5.5.30b}$$

Let

$$\tilde{x}(t) = (x^* - x(t)), \tag{5.5.31}$$

where x^* is the vector of true parameters. Then,

$$f^i(x) = G^i\tilde{x} , \tag{5.5.32}$$

so that (5.5.30) becomes

$$\dot{\tilde{x}}(t) = R^{-1}(t)(\Phi^1 G^1 + \Phi^2 G^2)\tilde{x} \triangleq R^{-1}(t)G\tilde{x} , \tag{5.5.33a}$$

$$\dot{R}(t) = (\Phi^1 G^1 + \Phi^2 G^2) - R(t) \triangleq G - R(t) \tag{5.5.33b}$$

where

$$G = \Phi^1 G^1 + \Phi^2 G^2 . \tag{5.5.34}$$

Introducing the function

$$V(\tilde{x},R) = \tilde{x}^T R\tilde{x} , \tag{5.5.35}$$

we can easily check that this is a Lyapunov function which proves that the ODE (5.5.33) is asymptotically stable, with domain of attraction $\{(\tilde{x},R):R>0\}$ and converges to the point $(\tilde{x},R) = (0,G)$, provided that $G>0$.

The condition $G>0$ effectively requires that the input processes $u^1(t)$ and $u^2(t)$ are "sufficiently rich" so that all parameters may be identified. Note that if we have $G^i>0$, for some i, then processor i would be able to identify all unknown parameters by itself. However, it is conceivable that G^1 and G^2 are singular, but $G^1 + G^2 > 0$ (so that $\Phi^1 G^1 + \Phi^2 G^2 > 0$). In that case, no processor may identify the parameters by himself, but the two together can. The condition (5.5.18) introduced in Theorem 5.5.1 is a similar "joint identifiability" condition.

We have effectively shown that the distributed RLS algorithm converges appropriately under certain assumptions, including the assumption $G>0$. However, this result is valid under the assumption that the algorithm returns an infinite number of times to a bounded region. This latter assumption has to be verified using different means, but we conjecture that it holds.

A further issue which suggests itself is the problem of choosing Φ^1, Φ^2 so as to maximize the speed of convergence of the algorithm. Given that the ODE (5.5.33) is an approximate description of the asymptotic behavior of the algorithm, the question becomes: given G^1, G^2 what are the choices of Φ^1, Φ^2 which maximize the rate of convergence of the ODE (5.5.33)?

<u>Example 2: Two Processes Driven by a Common Colored Noise.</u>

Consider two stochastic processes $z^1(t)$, $z^2(t)$ which are generated by

$$A^i(q)z^i(t) = B^i(q)u^i(t) + w(t), \quad i=1,2. \tag{5.5.36}$$

Here $A^i(q)$, $B^i(q)$ are polynomials, q is the unit delay operator, $u^i(t)$ is an input stochastic process and w(t) is a common noise process. We assume that w(t) is not white and is generated according to

$$w(t) = C(q)v(t), \tag{5.5.37}$$

where C is a monic polynomial and v(t) is a white process. (See Figure 5.5.3). Without loss of generality, we assume that the polynomials $A^i(\cdot)$, $B^i(\cdot)$, $C(\cdot)$ all have the same degree, which is denoted by k.

In the single processor case, it is well-known that LMS or RLS algorithms lead to biased estimates. Consistent estimates may be obtained, however, if a processor also tries to identify the parameters of the polynomial C(q). One such algorithm, the Extended Least Squares algorithm has been shown to be globally convergent and a returning condition is not required [Solo, 1979]. (Some confusion may arise due to the fact that different authors use different names to denote the same algorithm. For example, what we call ELS, is called AML by Solo.)

For the example, in Figure 5.5.3, either processor could try to identify $A^i(q)$, $B^i(q)$, C(q) by itself. However, since they both identify the common noise source C(q) they might benefit by exchanging and combining the estimates of the coefficients of C(q). We then obtain the following algorithm (this is the stochastic approximation version of ELS):

Let

$$\theta_0^* = (c_1,\ldots,c_k) \tag{5.5.38}$$

$$\theta_i^* = (a_1^i,\ldots,a_k^i,\ b_0^i,\ldots,b_k^i), \quad i=1,2 \tag{5.5.39}$$

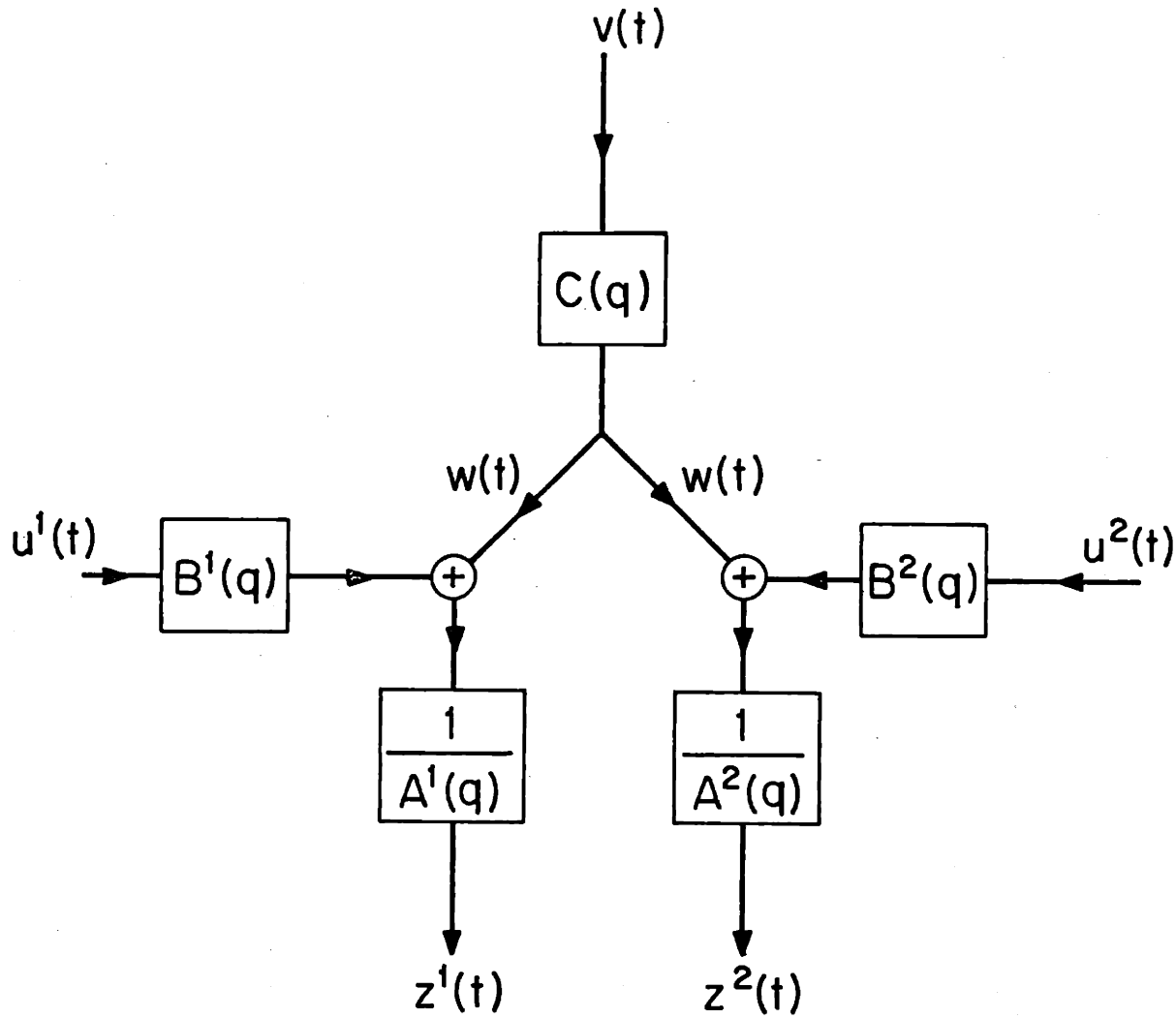

Figure 5.5.3: Two Systems Driven by a Common Colored Noise.

be the vectors of true parameters to be identified. Note that processor i is interested in identifying just the vector (θ_i^*, θ_0^*). Let $\hat{\theta}^i(n) = (\hat{\theta}_i^i(n), \hat{\theta}_0^i(n))$ be the estimate of processor i at time n. In the absence of any communications, the computations performed by processor i, at time n, are the following:

- Processor i has already computed

$$\psi^i(n) = (-z^i(n-1),\ldots,-z^i(n-k),\ u^i(n),\ldots,u^i(n-k),\ \eta^i(n-1),\ldots,\eta^i(n-k)). \quad (5.5.40)$$

- He updates $\hat{\theta}^i$ by

$$\hat{\theta}^i(n) = \hat{\theta}^i(n-1) + \frac{1}{n}\psi^i(n)(y^i(n) - \psi^{iT}(n)\hat{\theta}^i(n-1)) \quad (5.5.41)$$

- He computes the residual

$$\eta^i(n) = y(n) - \psi^{iT}(n)\hat{\theta}^i(n) \quad (5.5.42)$$

  and uses it to compute $\psi^i(n+1)$.

In the absence of communications, we obtain a different ODE for each processor. Namely:

$$\left.\begin{aligned} \dot{\hat{\theta}}_1^1(t) &= f_1(\hat{\theta}_1^1(t), \hat{\theta}_0^1(t)) \\ \dot{\hat{\theta}}_0^1(t) &= f_{10}(\hat{\theta}_1^1(t), \hat{\theta}_0^1(t)) \end{aligned}\right\} \quad (5.5.43)$$

and similarly for processor 2. In the presence of communications of the values of $\hat{\theta}_0^i$, let use assume that the Assumptions of Theorem 5.4.2 are satisfied and that $\Phi^i(t) = \Phi^i$ (independent of time). We then obtain a joint ODE, which is:

$$\left.\begin{aligned}
\dot{\hat{\theta}}_1(t) &= f_1(\hat{\theta}_1(t),\ \hat{\theta}_0(t)) \\
\dot{\hat{\theta}}_2(t) &= f_2(\hat{\theta}_2(t),\ \hat{\theta}_0(t)) \\
\dot{\hat{\theta}}_0(t) &= \Phi_1 f_{10}(\hat{\theta}_1(t),\ \hat{\theta}_0(t)) + \Phi_2 f_{20}(\hat{\theta}_2(t),\ \hat{\theta}_0(t))
\end{aligned}\right\} \tag{5.5.44}$$

Ljung [1977b] has shown that the ODE (5.5.43) associated with the centralized ELS algorithm is asymptotically stable (and all required conditions are satisfied) provided that the positive real condition

$$\text{Re } C(e^{i\theta}) > 0, \quad \forall\theta \in [0,2\pi] \tag{5.5.45}$$

is satisfied. Moreover, stability may be demonstrated using the Lyapunov function

$$V^i(\theta_i,\theta_0) = ||\theta_i-\theta_i^*||^2 + ||\theta_0-\theta_0^*||^2 \tag{5.5.46}$$

Even though (5.5.43) is stable, for i=1,2, it does not seem to follow that (5.5.44) is stable as well. For this reason, we introduce a small modification. Namely, we assume that processor i updates the first 2k components of $\hat{\theta}^i$, using

$$\hat{\theta}^i(n) = \hat{\theta}^i(n-1) + \frac{\Phi^i}{n}\psi^i(n)(y^i(n) - \psi^{iT}(n)\hat{\theta}^i(n-1)), \tag{5.5.46}$$

instead of (5.5.41). The remaining k components (those corresponding to C(q)) are updated according to (5.5.41). Then, the ODE associated to the decentralized algorithm becomes:

$$\left.\begin{aligned}
\dot{\hat{\theta}}_1(t) &= \Phi_1 f_1(\hat{\theta}_1(t),\ \hat{\theta}_0(t)) \\
\dot{\hat{\theta}}_2(t) &= \Phi_2 f_2(\hat{\theta}_2(t),\hat{\theta}_0(t)) \\
\dot{\hat{\theta}}_0(t) &= \Phi_1 f_{10}(\hat{\theta}_1(t),\ \hat{\theta}_0(t)) + \Phi_2 f_{20}(\hat{\theta}_2(t),\ \hat{\theta}_0(t))
\end{aligned}\right\} \tag{5.5.47}$$

This is a convex combination of two stable ODE's with a common Lyapunov function

$$V(\theta_1, \theta_2, \theta_0) = ||\theta_0 - \theta_0^*||^2 + ||\theta_1 - \theta_1^*||^2 + ||\theta_2 - \theta_2^*||^2 \quad (5.5.48)$$

It follows easily, that the (modified) decentralized ELS algorithm is associated to an asymptotically stable ODE. Consequently, it converges to the true parameters, provided that the returning condition is satisfied.

5.6 DECENTRALIZED GRADIENT ALGORITHM FOR AN ADDITIVE COST FUNCTION

The results of Section 5.3 on deterministic pseudo-gradient algorithms (Theorem 5.3.1) show that, given a bound on the time between consecutive communications and on the communication delay, we can find a small enough step-size so that the algorithm is convergent. Moreover, by tracing the steps in the proof of Theorem 5.3.1, it is possible to actually evaluate a bound γ^* on the step-size, to ensure convergence.

We may also follow a reverse approach: given a step-size, we may evaluate a bound on the time between consecutive communications and communication delays, so that the algorithm converges. However, the bounds to be so obtained are not necessarily tight, much of the structure of the problem may be lost and, finally, they may be not particularly illuminating.

In this section we impose more structure on the nature of the optimization problem and the algorithm under study, so as to obtain more specific results. The conceptual motivation behind this new approach is based on the following statement which seems reasonable, at least on an intuitive basis, from a normative perspective:

If an optimization problem consists of subproblems, each subproblem being assigned to a different agent (processor), then the frequency of communications between a pair of processors should reflect the degree by which their respective subproblems are coupled together.

The above statement is fairly hard to capture mathematically. We believe that this is accomplished, at least to some degree, by the model and the results of this section. An extensive conceptual discussion of these results may be found in Section 5.7.

Let $J:\mathbb{R}^M \to [0,\infty)$ be a cost function to be minimized, which has a special structure:

$$J(x) = J(x_1,\ldots,x_M) = \sum_{i=1}^{M} J^i(x_1,\ldots,x_M), \tag{5.6.1}$$

where $J^i:\mathbb{R}^M \to [0,\infty)$. So far, equation (5.6.1) does not impose any restriction on J; we will be interested, however, in the case where, for each i, J^i depends on x_i and only a few more components; consequently, the Hessian matrix of each J^i is sparse.

We view J^i as a cost directly faced by processor i. This processor is free to fix or update the component x_i, but its cost also depends on a few interaction variables (other components of x) which are under the authority of other processors.

We may visualize the structure of the interactions by means of a directed graph $G = (V,E)$:

(i) The set V of nodes of G is $V = \{1,\ldots,M\}$.

(ii) The set of edges E of the graph is

$$E = \{(i,j): J^j \text{ depends on } x_i\} . \tag{5.6.2}$$

For example, the graph is Figure 5.6.1 corresponds to a cost function of the form $J^1(x_1) + J^2(x_1, x_2, x_3) + J^3(x_2, x_3)$.

Since we are interested in the fine structure of the problem, we quantify the interactions between subproblems by assuming that the following bounds are available:

$$\left|\frac{\partial^2 J}{\partial x_i \partial x_j}\right| \leq K_{ij}, \quad \left|\frac{\partial^2 J^k}{\partial x_i \partial x_j}\right| \leq K_{ij}^k, \quad \forall x \in \mathbb{R}^M \tag{5.6.3}$$

and note that K_{ij} can be chosen so that

$$K_{ij} \leq \sum_{k=1}^{M} K_{ij}^k \tag{5.6.4}$$

A centralized gradient-type algorithm for minimizing J is the following:

$$\lambda_i^j(n) = \frac{\partial J^j}{\partial x_i}(x(n)), \tag{5.6.5}$$

$$x_i(n+1) = x_i(n) - \gamma_i \left(\sum_{j=1}^{M} \lambda_i^j(n)\right). \tag{5.6.6}$$

The summation is (5.6.6) needs to be carried out only for those j's such that $(i,j) \in E$, because otherwise $\lambda_i^j(n)$ is zero. This has to be kept in mind when considering implementations of the algorithm. The above algorithm may be implemented in a <u>synchronous</u> decentralized way as follows:

<u>Algorithm 1:</u>

1. For each $(i,j) \in E$, processor i evaluates $\lambda_i^j(n)$ according to (5.6.5).

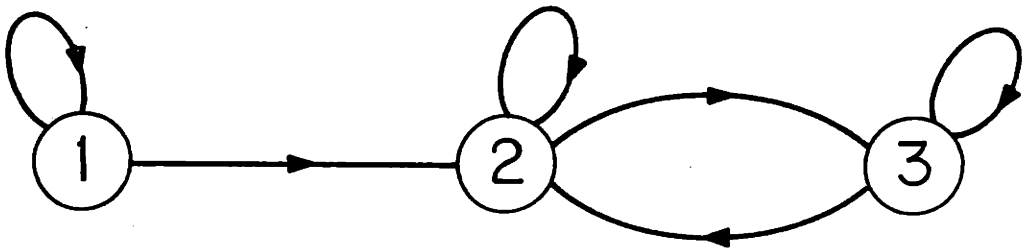

Figure 5.6.1: The Graph Associated to an Additive Cost Function.

2. Each processor i uses (5.6.6) to update x_i.

3. For each i,j,k such that $(i,j) \in E$, $(k,j) \in E$, processor k transmits x_k to processor i.

It is more interesting, however, to assume that the functional form of J^j is stored only in the memory of processor j and no other processor "knows" J^j, i.e. the input is itself decentralized. (Such a configuration has lower memory requirements). In such a case, it is more meaningful to let the first step in Algorithm 1 be executed by processor j (instead of processor i) and then have processor j transmit the result to processor i. We then obtain the following:

Algorithm 2:

1. For each $(i,j) \in E$, processor j evaluates $\lambda_i^j(n)$ according to (5.6.5).

2. For each $(i,j) \in E$, processor j transmits $\lambda_i^j(n)$ to processor i.

3. Each processor i uses (5.6.6) to update x_i.

4. For each $(i,j) \in E$, processor i transmits $x_i(n+1)$ to processor j.

Algorithm 2 is mathematically equivalent to the centralized algorithm (5.6.5) - (5.6.6) but has certain drawbacks: a) There are strict synchronization requirements; b) If there is some pair (i,j) such that communications from i to j are particularly slow, the operation of all processors has to be slowed down. For these reasons we are interested in an asynchronous version of Algorithm 2 which tolerates communication

delays. In fact, the algorithm we present below may also allow us to reduce the number of transmitted messages per stage. The specific advantages are discussed in detail in section 5.7.

We let, as usual, $x^i(n) = (x_1^i(n),\ldots,x_M^i(n))$ denote the x-vector stored in the memory of processor i at time n. We also assume that each processor i stores in its memory another vector $(\lambda_i^1(n),\ldots,\lambda_i^M(n))$ with its estimates of $\frac{\partial J^1}{\partial x_i},\ldots,\frac{\partial J^M}{\partial x_i}$. We do not require that a message be transmitted at each time stage and we allow communication delays. So, let

$p^{ki}(n)$ = the time that a message with a value of x_k was sent from processor k to processor i, and this was the last such message received no later than time n.

$q^{ki}(n)$ = the time that a message with a value of $\frac{\partial J^k}{\partial x_i}$ was sent from processor k to processor i, and this was the last such message received no later than time n.

For consistency of our notation we let

$$p^{ii}(n) = q^{ii}(n) = n, \quad \forall i, \quad \forall n. \tag{5.6.7}$$

With the above definitions, we have:

$$x_k^i(n) = x_k^k(p^{ki}(n)), \quad \forall n, \quad \forall (k,i) \in E, \tag{5.6.8}$$

$$\lambda_i^k(n) = \frac{\partial J^k}{\partial x_i}(x^k(q^{ki}(n))), \qquad \forall n, \forall (i,k) \in E. \tag{5.6.9}$$

Equations (5.6.8), (5.6.9) together with

$$x_i^i(n+1) = x_i^i(n) - \gamma_i \sum_{j=1}^{M} \lambda_i^j(n) \tag{5.6.10}$$

specify completely the asynchronous decentralized algorithm to be studied.

As in Theorem 5.3.1 (for the specialization case), we assume that the time between consecutive communications and the communication delays are bounded. However, we allow the bounds to be different for each pair of processors and type of message:

Assumption 5.6.1: For some constants P^{ik}, Q^{ik},

$$n - P^{ik} \leq p^{ik}(n) \leq n, \qquad \forall (i,k) \in E, \quad \forall n, \tag{5.6.11}$$

$$n - Q^{ik} \leq q^{ik}(n) \leq n, \qquad \forall (k,i) \in E, \quad \forall n . \tag{5.6.12}$$

Note that we may let $P^{ik} = Q^{ik} = 0$, to recover a synchronous algorithm.

The result that follows states that the algorithm converges, if the time between consecutive communications plus communication delays between any two processors is not too large compared to the degree of coupling of their subproblems.

Theorem 5.6.1: Suppose that for each i

$$\frac{2}{\gamma_i} > \sum_{j=1}^{M} K_{ij} + \sum_{k=1}^{M} \sum_{j=1}^{M} K_{ij}^{k} (P^{ik} + Q^{kj} + P^{jk} + Q^{ki}). \tag{5.6.13}$$

Let $z(n) = (x_1^1(n), x_2^2(n), \ldots, x_M^M(n))$. Then

$$\lim_{n\to\infty} \frac{\partial J}{\partial x_i} (z(n)) = 0, \quad \forall i. \tag{5.6.14}$$

Proof: Let $s_i(n) = \sum_{k=1}^{M} \lambda_i^k(n)$ and note that

$$x_i^i(n+1) = x_i^i(n) - \gamma_i s_i(n) . \tag{5.6.15}$$

From a Taylor series expansion for J we obtain:

$$J(z(n+1)) \leq J(z(n)) - \sum_{i=1}^{M} \gamma_i \frac{\partial J}{\partial x_i}(z(n))s_i(n) +$$

$$+ \frac{1}{2} \sum_{i=1}^{M} \gamma_i^2 (\sum_{j=1}^{M} K_{ij}) |s_i(n)|^2 \leq$$

$$\leq J(z(n)) - \sum_{i=1}^{M} \gamma_i |s_i(n)|^2 + \sum_{i=1}^{M} \gamma_i \left|\frac{\partial J}{\partial x_i}(z(n)) - s_i(n)\right| \cdot |s_i(n)| +$$

$$+ \frac{1}{2} \sum_{i=1}^{M} \gamma_i^2 \left(\sum_{j=1}^{M} K_{ij}\right) |s_i(n)|^2 . \qquad (5.6.16)$$

Using (5.6.9),

$$\left|\frac{\partial J}{\partial x_i}(z(n)) - s_i(n)\right| = \left|\sum_{k=1}^{M} \left[\frac{\partial J^k}{\partial x_i}(z(n)) - \lambda_i^k(n)\right]\right| \leq$$

$$\leq \sum_{k=1}^{M} \left|\frac{\partial J^k}{\partial x_i}(z(n)) - \frac{\partial J^k}{\partial x_i}(x^k(q^{ki}(n)))\right| \leq$$

$$\leq \sum_{k=1}^{M} \sum_{j=1}^{M} K_{ij}^k |x_j^j(n) - x_j^k(q^{ki}(n))| . \qquad (5.6.17)$$

Now note that

$$|x_j^j(n) - x_j^k(q^{ki}(n))| = |x_j^j(n) - x_j^j(p^{jk}(q^{ki}(n)))| \leq$$

$$\leq \sum_{m=n-Q^{ki}-P^{jk}}^{n-1} \gamma_j |s^j(m)| \qquad (5.6.18)$$

Hence,

$$J(z(n+1)) \leq J(z(n)) - \sum_{i=1}^{M} \gamma_i |s_i(n)|^2 + \frac{1}{2} \sum_{i=1}^{M} \gamma_i^2 \left(\sum_{j=1}^{M} K_{ij}\right) |s^i(n)|^2 +$$

$$+ \sum_{i=1}^{M} \gamma_i \sum_{k=1}^{M} \sum_{j=1}^{M} K_{ij}^k \sum_{m=n-Q^{ki}-P^{jk}}^{n-1} \gamma_j |s_j(m)| \cdot |s_i(n)| \qquad (5.6.19)$$

Using the inequality

$$\gamma_i\gamma_j|s_i(n)||s_j(m)| \leq \frac{1}{2}(\gamma_i^2|s_i(n)|^2 + \gamma_j^2|s_j(m)|^2), \tag{5.6.20}$$

and summing (5.6.19) for different values of n, we obtain

$$\begin{aligned} J(z(n+1)) \leq J(z(0)) &- \sum_{\ell=0}^{n}\sum_{i=1}^{M}\gamma_i|s_i(\ell)|^2 + \\ &+ \sum_{\ell=0}^{n}\sum_{i=1}^{M}\frac{1}{2}\gamma_i^2\left(\sum_{j=1}^{M}K_{ij}\right)|s^i(\ell)|^2 + \\ &+ \frac{1}{2}\sum_{\ell=0}^{n}\sum_{i=1}^{M}\sum_{k=1}^{M}\sum_{j=1}^{M}K_{ij}^k\sum_{m=\ell-Q^{ki}-P^{jk}}^{\ell-1}(\gamma_i^2|s_i(\ell)|^2 + \gamma_j^2|s_j(m)|^2) \end{aligned} \tag{5.6.21}$$

Note that

$$\begin{aligned} &\sum_{\ell=0}^{n}\sum_{m=\ell-Q^{ki}-P^{jk}}^{\ell-1}(\gamma_i^2|s_i(\ell)|^2 + \gamma_j^2|s_j(m)|^2) = \\ &= (P^{jk} + Q^{ki})\sum_{\ell=0}^{n}\gamma_i^2|s_i(\ell)|^2 + \sum_{m=0}^{n-1}\sum_{\ell=m+1}^{m+Q^{ki}+P^{jk}}\gamma_j^2|s_j(m)|^2 \leq \\ &\leq (P^{jk} + Q^{ki})\sum_{\ell=0}^{n}[\gamma_i^2|s_i(\ell)|^2 + \gamma_j^2|s_j(\ell)|^2] \end{aligned} \tag{5.6.22}$$

We now use (5.6.22) to rewrite the last iterated sum in (5.6.21) as

$$\begin{aligned} &\frac{1}{2}\sum_{\ell=0}^{n}\sum_{i=1}^{M}\sum_{k=1}^{M}\sum_{j=1}^{M}K_{ij}^k(P^{jk} + Q^{ki})[\gamma_i^2|s_i(\ell)|^2 + \gamma_j^2|s_j(\ell)|^2] = \\ &= \frac{1}{2}\sum_{\ell=0}^{n}\sum_{i=1}^{M}\sum_{k=1}^{M}\sum_{j=1}^{M}K_{ij}^k(P^{jk} + Q^{ki} + P^{ik} + Q^{kj})\gamma_i^2|s_i(\ell)|^2 \end{aligned} \tag{5.6.23}$$

Using (5.6.23) in (5.6.21) and collecting terms, we have

$$J(z(n+1)) \leq J(z(0)) + \sum_{\ell=0}^{n} \sum_{i=1}^{M} |s_i(\ell)|^2 \Big[-\gamma_i + \frac{1}{2} \gamma_i^2 \sum_{j=1}^{M} K_{ij} +$$

$$+ \frac{1}{2} \gamma_i^2 \sum_{k=1}^{M} \sum_{j=1}^{M} K_{ij}^k (P^{jk} + Q^{ki} + P^{ik} + Q^{kj}) \Big] \qquad (5.6.24)$$

We now use (5.6.13) and conclude that

$$\sum_{\ell=0}^{n} |s_i(\ell)|^2 \leq AJ(z(0)), \quad \forall i,n, \qquad (5.6.25)$$

We may finally use inequalities (5.6.17) and (5.6.18) to obtain (5.6.14).■

We close this section with a few remarks:

1. The bounds provided by (5.6.13) are sufficient for convergence but they are not tight, nor necessary. In particular, the first inequality in (5.6.16) is not tight, even if J is a quadratic function, because it does not take into account the signs of the entries in the Hessian matrix of J. It is also conceivable that inequality (5.6.20) could be improved by taking into account the fine structure of the problem.

It is known [Bertsekas, 1983] that a decentralized algorithm of a similar type may converge in certain special cases, even if the γ_i's are held fixed while the bounds P^{ik}, Q^{ik} are allowed to be arbitrarily large. So, the gap between the sufficient conditions (5.6.13) and the necessary conditions may be substantial. Further research might narrow this gap.

2. The convergence rate of the decentralized algorithm should be expected to deteriorate as the bounds P^{ik}, Q^{ik} increase. A characterization of the convergence rate, however, seems to be a fairly hard problem.

5.7 TOWARDS ORGANIZATIONAL DESIGN

We have argued earlier that a central theme in the design of decentralized systems is the question "who should communicate to whom, what and how often". In this section we discuss our results (mainly those of section 5.6) in the context of the above question.

It has been often suggested that the behavior of a boundedly rational human decision maker (or economic agent) may be modeled as a descent-type iterative algorithm. Proceeding along the same lines, we may view distributed descent (e.g. pseudo-gradient) algorithms as a model of adjustment or as a behavioral model in a human organization. This coincides with the models of Arrow and Hurwicz [1960] for adjustment in economic markets and of Meerkov [1979] for societies of animals. Other algorithms from mathematical programming (e.g. the Dantzig-Wolfe decomposition) have been proposed as models of the budget-allocation process in a divisionalized company.

Making the assumption that an iterative algorithm describes the behavior of the members of an organization, let us now place ourselves in the position of the designer of an organization. His task is to create a chart prescribing the flow of information between the decision-makers. Let us assume that the objective of the organization is the minimization of a certain organizational cost which is the sum of the costs faced by each division. To each division, there corresponds a decision-maker who is knowledgeable enough about the structure of the problem he is facing, to the extent that given a tentative decision (or operating point) he is able to change his decision in a direction which results to improvement. We also assume that the divisions are

interacting in some way; so that the decisions of one decision maker may affect the costs of another division. It follows that a decision maker who is interested in the well-being of the entire organization needs to take into account two types of information, coming from other divisions:

(i) "What everybody else is doing". (Because their decisions affect what is a "good" decision for my subproblem.)

(ii) "How do my decisions influence them". (That is, I need to take into account the side-effects of my decisions on other divisions).

Moreover, since the organization is assumed to undergo a process of change and adjustment, this information has to be updated, from time to time, so as to keep up with the changes that have occured. However, this updating occurs in a fairly asynchronous manner, because it is unreasonable to assume that decision makers in an organization make changes in a synchronized manner and that they inform each other each time they make a change.

It should be clear that the above described setting closely resembles the algorithm studied in Section 5.6. In particular, messages of component x_i from processor i carry the information of "what everybody else is doing" and messages λ_j^i carry the information of how processor j influences the i-th division.

Having established this relationship between the organizational problem and the mathematical description, we proceed to the main question, "who should communicate to whom, what, how often". We address each component of this question separately:

(i) Who should communicate to whom? Clearly, a pair of decision makers should communicate if and only if their divisions interact in some way. Mathematically, whether two divisions interact or not may be seen from the graph G that was constructed in Section 5.6. Messages should flow along the edges of that graph.

(ii) What should there be communicated? This is also an easy question (at least in our framework). If $(i,j) \in E$ (that is x_i influences J^j), decision maker i should inform j of his decisions and j should "reply" to i how he is affected by these decisions.

(iii) How often should they communicate? It is intuitively obvious that the frequency of communications between two decision makers should be proportional to the degree of coupling between their subproblems. While this statement is, in general, hard to make mathematically precise, it may be effectively studied in the framework that we have introduced: the "degree of coupling between divisions" is quantified in terms of the bounds K^k_{ij} on the second derivatives of the costs; the frequency of communications are also quantified in terms of the bounds P^{ij}, Q^{ij}. Finally, these are coupled together, via the sufficient conditions of Theorem 5.6.1, which prescribes the frequency of communications in terms of interconnection strengths and the speed of adjustment γ_i.

We may summarize the above discussion by saying that (for the above assumed model of behavior in cooperative organizations), the approach of Section 5.6 may be used to design an organizational structure, that is, prescribe the nature of communication flows. It must be pointed out, however, that Theorem 5.6.1 defines an entire admissible

set of organizational structures. Choosing a particular element of this set requires more study of the effect of the bounds P^{ij}, Q^{ij} on the convergence rate of the algorithm.

One final remark on organizational design. It is conceivable that the structure of the optimization problem slowly changes with time, and so do the bounds K^k_{ij}, although in a time scale slower than the time scale of the adjustment process. In such a case, the bounds P^{ij}, Q^{ij} should also change. This leads to a natural two-level organizational structure: At the lower level, we have a set of decision makers continuously adjusting their decisions and exchanging messages. At a higher level, we have a supervisor who monitors changes in K^k_{ij} and accordingly instructs the low-level decision makers to adjust their communication rates. Note that the supervisor does not need to know the details of the cost function; he only needs to know the degree of coupling between the divisions. This seems to reflect the actual structure of existing organizations. Low level decision makers are "experts" on the problems directly facing them, while higher level decision makers only know certain structural properties of the overall problem and make certain global decisions, e.g. setting the communication rates.

Another possibility is to let the low-level decision makers monitor the couplings K^k_{ij} and accordingly adjust the communication rates by themselves, without wating for instructions from above. This essentially amounts to another decentralized algorithm (superimposed on the original one) which aims at solving the set of inequalities (5.6.13) of Theorem 5.6.1.

A further topic for research relevant to organization theory is the following: given the mathematical framework we have introduced, are there any generic properties of particular organizational structures? For example, if we have a hierarchy (that is the graph G of section 5.6 is a tree) does the adjustment process have any specific properties which are structurally different than those for general graphs?

5.8 OTHER POSSIBLE APPLICATIONS

Routing in Communication Networks

The problem of (quasi-static) routing of messages in a communication network, may be formulated as a nonlinear programming problem, whose objective is to minimize some performance criterion related to the delays in the transmission of messages through the network [Cantor and Gerla, 1974]. There are plenty of reasons for which one would like to have this problem solved in a decentralized way. Certain decentralized algorithms have been proposed [Gallager, 1977; Gafni and Bersekas, 1983] but most often convergence is proved under an assumption of strict synchronism, which is undesirable and unrealistic. Our approach may be used, however, to prove convergence under more realistic assumptions, tolerating a fair amount of asynchronism. A small modification of our results and proofs would be needed, however, to handle problems with inequality constraints, because such constraints are present in communication network problems.†

As a more specific application one may consider the algorithm of Gafni and Bertsekas [1983] for routing in communication networks utilizing virtual circuits. This algorithm admits a convenient on-line

† A result of this type has been recently obtained for a gradient projection algorithm [Tsitsiklis and Bertsekas, 1984].

decentralized implementation and, under certain assumptions, yields approximately optimal routing strategies, even within the class of dynamic strategies. We strongly conjecture that the same is true for the asynchronous version of this algorithm.

Parallel Computation

Many architectures for parallel computation have been proposed recently, ranging from general purpose multi-processors to special purpose VLSI architectures. Together comes, of course, an effort to develop good parallel algorithms that suit the available architectures.

Consider an optimization problem with the structure assumed in Section 5.6 and suppose that we have available a multi-processor architecture: each processor is assumed to be capable of storing one of the additive terms J^k of the cost function and evaluate its partial derivatives at a given point. We also assume that the processors are arranged in a graph which coincides with the one induced by the structure of the cost function (see Section 5.6).

Given this architecture, we considered, in Section 5.6, both synchronous and asynchronous algorithms for the same problem, which we now compare:

The main disadvantages of the synchronous algorithm are the following:

(i) Synchronism. This requires a synchronization protocol which may introduce substantial overhead and, therefore, be practically undesirable.

(ii) Bottlenecks caused by communication delays. If there is some pair of processors (i,j) such that the delay of messages transmitted

from i to j is excessively large (compared to the delay of other messages), then all processors should remain idle until the message is received by processor j, and only then could they proceed to the next round of computation. Therefore, the speed of the algorithm is determined by the largest communication delay and processors may have to remain idle a large fraction of time.

(iii) Excessive Communication Requirements. At each stage of the algorithm, a message must be transmitted along each arc in the associated graph. This may be problematic, especially if communications capacity is a scarce resource, as is often the case in VLSI architectures. Moreover, if many messages are routed using a common bus, large delays, and therefore bottlenecks, may result.

We now indicate how all the above mentioned problems are alleviated by the asynchronous algorithm that we introduced in Section 5.6: First, it has no synchronization requirements. Moreover, the proof of Theorem 5.6.1 may be easily modified to handle the case in which processors are allowed not to update, once in a while (see Corollary 5.3.1.) This could capture the possibility that certain processors perform computations faster than others, or that their individual clocks do not run at the same speed. Second, and most important, our asynchronous algorithm allows communication delays and infrequent communications, without creating bottlenecks: If processor i has not received the value of some component that it needs for its own computations, it keeps computing using a value for that component it had received some time in the past. So, it will update slightly

in the wrong direction, but this may be substantially better than not updating at all.

Of course, there is a certain tradeoff: if communications become too infrequent and delays too large, while the step-size is held constant, the convergence rate of the algorithm worsens or the algorithm need not converge at all. It is an open research topic how to handle this tradeoff. It is intuitively clear - at least if the Hessian matrix of the cost function is diagonally dominant in some sense and communication delays are not too large - that the asynchronous algorithm will be faster than the synchronous one, but more research is needed, including numerical experimentation.

5.9 SUMMARY OF PROBLEMS FOR FUTURE RESEARCH

In the preceding Sections we have mentioned or alluded to certain problems, related to the ones studied in this chapter, which are potential topics for future research. In this Section we put them together, for easier reference, and to place them into perspective. Since this is mainly a "classification" section, we will be very brief and the reader should consult earlier sections for more details. The problems outlined below fall into three general categories: a) Simple modifications of results in this chapter; b) New research directions of general (theoretical) nature; c) Applications in specific fields.

1. Obtain convergence results for distributed algorithms for constrained optimization, in a setting similar to that of Section 5.3.

2. Obtain convergence results for algorithms in which there are no bounds on the time between consecutive communication, but communications are "event-driven": a message is transmitted whenever a substantial change occurs. Compare, theoretically or through simulation, such algorithms with the old ones, as well as with the corresponding centralized (synchronous) algorithms.

3. Assume that the cost function J to be minimized is convex and see whether something special may be said, in view of the fact that the combining process consists of forming convex combinations.

4. Obtain precise results on the convergence rate of decreasing step-size distributed algorithms.

5. Modify Theorem 5.4.1 so as to be valid even if an associated ODE has only a bounded domain of attraction.

6. Prove analogs of Theorems 2 and 3 of Ljung [1977a] for distributed algorithms.

7. Investigate theoretically and/or with simulations the convergence rate of the gradient algorithm of Section 5.6. Also, obtain convergence conditions tighter than those of Theorem 5.6.1 and possibly narrow the gap between necessary and sufficient conditions for convergence.

8. Run simulations of distributed identification algorithms to evaluate their performance, especially during the initial transient period. Also, find more structures for distributed identification to which our results may be applied.

9. Use results on algorithms for constrained optimization for optimal routing and flow control in data communication networks.

10. Apply the results of Section 5.6 to problems of designing the pattern of communications, either for the purpose of speeding up parallel computation (in the context of Section 5.8) or for the purpose of designing a divisionlized organization.

11. Consider two-level organizations in which the higher level sets the rate of communications, while the lower lovel executes a distributed optimization algorithm (see Section 5.7).

12. Examine whether certain organizational structures (e.g. hierarchies) have any specific properties which reflect themselves on properties of the corresponding distributed algorithms (see Section 5.7).

5.10 SUMMARY AND CONCLUSIONS

A broad class of deterministic and stochastic iterative algorithms admit asynchronous decentralized implementations which retain the desirable convergence properties of their centralized (or synchronous) counterparts. The main requirements for convergence are that the time between consecutive transmission of messages, as well as communication delays, are not too large, when compared to the natural time scale associated with the algorithm. For algorithms with decreasing step-size, this allows communications to become more and more infrequent as the algorithm progresses.

For a class of optimization problems consisting of a set of coupled subproblems, we have shown that communication requirements depend, in a quantifiable way on the degree of coupling between subproblems.

Our results and, more generally, our line of approach may be useful in several different settings. For example, in parallel computation, in organizational design, in decentralized signal processing or in routing for data communication networks.

Some of the advantages of asynchronous algorithms are: there are no synchronization requirements, which makes implementation easier; bottlenecks to the speed of the algorithm, caused by communication delays, are relieved; finally, there may be savings in the total number of exchanged messages, so that overloading of communication channels is avoided.

CHAPTER 6: A GLOBAL VIEW

6.1 OVERVIEW

The results in this report have been already reviewed and discussed, at several places. Instead of an additional review, we discuss in this Section the main conceptual lines which link together the various pieces in our study.

What our results have in common, is that (almost) all refer to static decision problems (estimation or identification problems being a special case). The word "static" "static" is used here to distinguish from those problems in which we are interested in decentralized feedback control of a dynamical system. However, even though the underlying decision problems are static, time enters in our study in two ways: a) It may be the case that the information relevant to the problem is not obtained all at once, but sequentially; so, better decisions are being evaluated as more data become available; b) even if all data become available at once, a solution to the decision problem need not be obtained instantly, but through an iterative process of adjustment.

We first considered the case in which all relevant data become available at time zero and we investigated the question whether decisions may be evaluated instantly, without any communications. If not, we exclude the possibility of centralizing information by direct transmission of all data; rather, we consider sequential schemes for transfering information and evaluating decisions (decentralized protocols). Given that optimal protocols are hard to design, we consider a few specific (ad-hoc chosen) classes of protocols and address two types of questions: a) What happens when a specific protocol and rule for updating decisions is employed and b) What are the communication requirements of such a decentralized scheme in order to guarantee smooth and desirable operation (e.g. convergence).

The class of protocols that we have studied could not be exhaustive. Specific real-world applications might require drastically different ones. However, we have

focussed on fairly general and universally applicable ones, as witnessed by the wide range of possible applications we have suggested in earlier Chapters.

6.2 AN ORGANIZATIONAL VIEW

Recall that in Section 2.2 we had offered an abstract and schematic picture of the operation and evolution of real world organizations. We suggested there that such a picture could guide the selection of research topics. We now indicate the correspondence between some of the issues raised in Section 2.2 and our results.

In small and simple organizations, many decisions are being made without communicating. This corresponds to the problems studied in Chapter 3. We have seen, however, that silent coordination leads to hard problems. Consequently, apart from small and simple decision problems, communications become necessary, even if redundant. A common situation in which a set of decision makers have to communicate is when they have to agree on something. Chapter 4 effectively addresses such situations.

As the organization grows, it ceases being optimal and some decision makers start changing their mode of operation so as to improve performance. As their operation changes, they need- and do- inform those decision makers who should be concerned about such changes. This corresponds to a decentralized adjustment process, of the type studied in Chapter 5. Finally, as the organization becomes even more complex, higher levels of decision making are introduced who know who should be concerned about what and set up the necessary information flows. Such two-level schemes remain to be studied in the future.

6.3 GENERAL AREAS FOR FUTURE RESEARCH

Several problems which are closely related to our study have been already suggested in the main body of this report and will not be repeated here. We will suggest, however, some broader directions which seem to be of particular interest.

From a mathematical perspective, the problem "who should communicate to whom, what, etc." may be viewed as a problem of decentralized complexity theory. A lot of progress in this direction is being made, mainly in the computer science literature. Systems theorists, however, are more often concerned with continuous rather than combinatorial problems. This leads to the question whether a decentralized continuous complexity theory (possibly along the lines of Nemirovsky and Yudin [1983], Traub and Wozniakowsky, [1980]) is feasible. Some interesting issues concern the convergence rate of decentralized algorithms and the tradeoff with the number of communications. Given the possibility of coding an arbitrary amount of information in messages containing real numbers, this line of research will probably encounter some non-trivial issues of modelling of decentralized computation.

A second area of inquiry relates to organizational issues, hierarchical algorithms, whereby the higher levels dictate the information flows, or event-driven algorithms, in which communication protocols are not fixed in advance, but depend on the state of the environment.

A third area of inquiry could concern the nature of the (implicit) information exchange, when a set of decision makers observe the actions of other decision makers, with different information. This would be an extension of the results in Chapter 4 to situations in which the objective of the decision makers is not necessarily to reach consensus. Links with game theory may be investigated, especially when we have decision makers possessing different models of the situation.

APPENDIX A

Proof of Lemma 5.2.1: We only need to prove the Lemma for each component separately, since (5.2.2) corresponds to a decoupled set of linear systems. We may therefore assume that there is only one component; so, the subscript ℓ will be omitted and $\Phi^{ij}(n|k)$ becomes a scalar.

The coefficient $\Phi^{ij}(n|k)$ being the impulse response of the linear system (5.2.2) is determined by the following "experiment": let us fix a processor j and some time k; let $x^h(1)=0$, $\forall h$, $s^h(m)=0$, $\forall h,m$, unless if h=j and m=k in which case we let $\gamma^j(k)s^j(k)=v$, some nonzero element of H. For all times n and for all processors i, $x^i(n)$ will be a scalar multiple of v. This proportionality factor is precisely equal to $\Phi^{ij}(n|k)$. Since this experiment takes place in a one-dimensional subspace of H, we may assume -without loss of generality- that H is one-dimensional and that v=1. We then have, for the above "experiment", $\Phi^{ij}(n|k) = x^i(n)$, $\forall i,n$. A trivial induction based on (5.2.2) and using (5.2.3) shows that $x^i(n)\geq 0$, $\forall i,n$, which proves (5.2.8).

For inequality (5.2.9) we consider a different "experiment": let us fix some time k and let $x^h(1)=0$, $\forall h$, $s^h(m)=0$, $\forall m,h$, unless if m=k in which case we let $\gamma^h(k)s^h(k)=1$, $\forall h$. (H is still assumed one-dimensional). We then use (5.2.2) and (5.2.4) to show by a simple inductive argument that $x^i(n)\leq 1$, $\forall i,n$ which concludes the proof of part (i).

For the proof of the remaining parts of the Lemma we first perform a reduction to a simpler case. It is relatively easy to see that we may assume, without loss of generality, that communication delays are nonzero. For example, we could redefine the time variable so that one time unit for the old time variable

corresponds to two time units for the new one and so that any message that had zero delay for the original description has unit delay for the new description. If any of the Assumptions 5.2.1, 5.2.2, 5.2.3, 5.2.4 holds in terms of the old time variable, it also remains true with the new one.

Next, we perform a reduction to the case where all messages have zero delay, as follows: for any $(i,j)\in E$, we introduce a finite set of dummy processors, between i and j (see Fig. A.1) which act as buffers. Any message from i to j is first transmitted to a buffer processor (with zero delay) which holds it for time equal to the desired delay and then transmits it to j, again with zero delay. (So, whenever a buffer processor h receives a message, it lets $a^{hi}(n)=1$). The buffer processor which is to be employed for any particular message is determined by a round-robin rule. Note that for each pair $(i,j)\in E$ the number of buffer processors that needs to be introduced is equal to the maximum communication delay for messages from i to j, which has been assumed finite.* Note also that the buffer processors are non-computing ones and have in-degree equal to 1.

It is easy to see that if Assumptions 5.2.2, 5.2.3 are valid for the original description of the algorithm, they are also valid for the above introduced augmented description, possibly with a different choice of constants. Assumption 5.2.1 also remains valid except for part c(iii) which may be violated. (If in Figure A.1 processor j had originally in-degree equal to 1, it now has in-degree equal to 4, but there is nothing in our assumptions that guarantees that $a^{jj}(n)\geq\alpha$, $\forall n$). We have, nevertheless, the following condition:

* This procedure is equivalent to a state augmentation for the linear system (5.2.2), as discussed in Section 5.2.1.

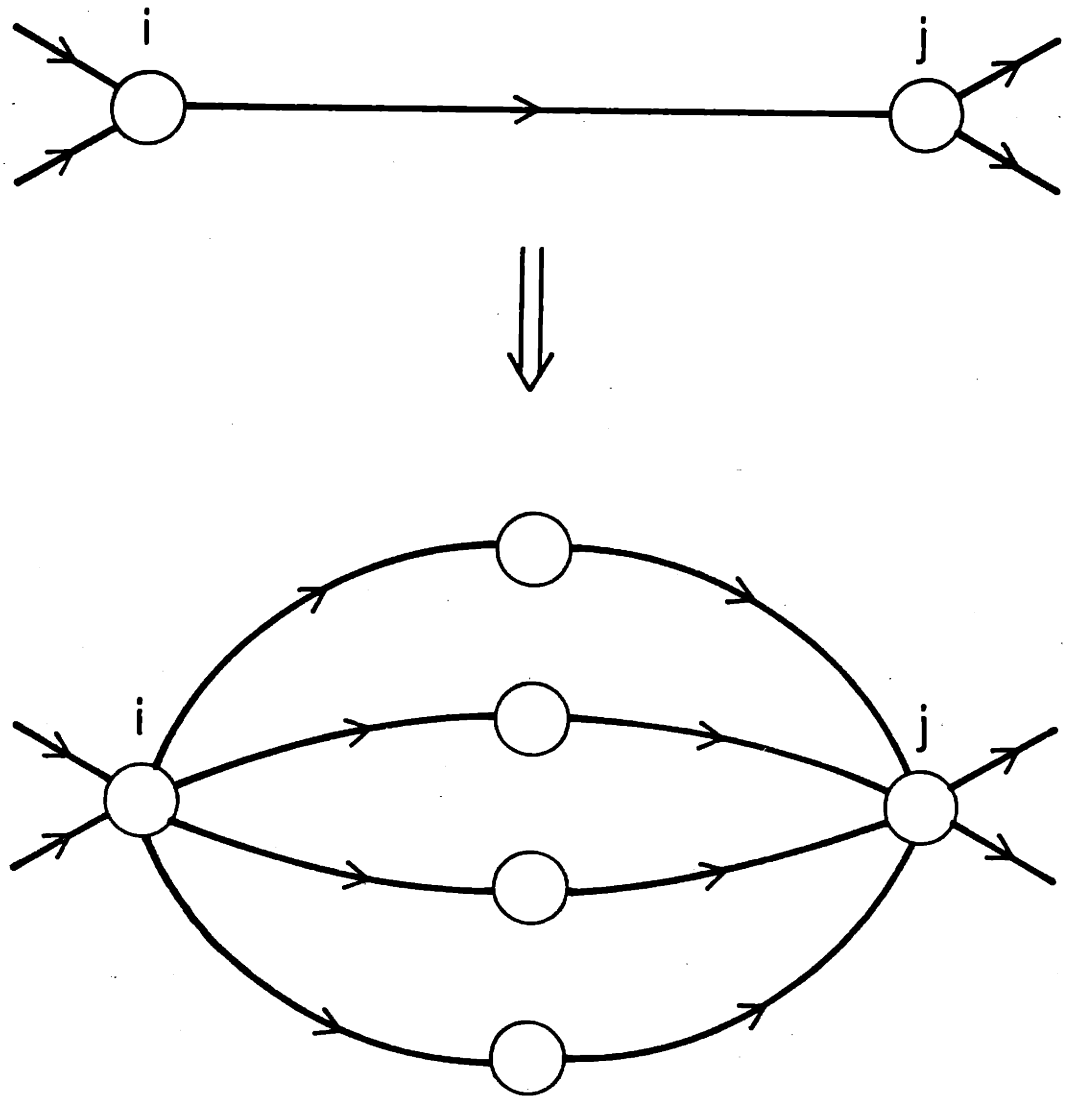

Figure A.1: Reduction to the zero delay case.

<u>Assumption A.1</u>: For any processor i with in-degree larger than 1, either

1. Assumption 5.2.1c(iii) holds, or

2. All predecessors of i are non-computing, have in-degree 1 and a common predecessor.

From now on, we assume, without loss of generality, that communication delays are zero, provided that we replace Assumption 5.2.1c(iii) by Assumption A.1.

Let $A(n)$, $\Phi(n|k)$ be the matrices with coefficients $a^{ij}(n)$, $\Phi^{ij}(n|k)$, respectively. Because of the zero delay assumption it is easy to see that $\Phi(n|k) = \prod_{m=k+1}^{n-1} A(m)$, $n>k+1$.*

We first prove the desired results under Assumptions 5.2.1 and 5.2.2. By combining Assumptions 5.2.1c(i) and 5.2.2, note that there exists a constant B such that, for any $(i,j)\in E$ and any time interval I of length B, there exists some $t\in I$ such that $a^{ji}(t)\geq\alpha> 0$. (These are times that i communicates to j.)

Let us fix a computing processor j and some time k. By relabelling, let us assume that j=1. We will show by induction (with respect to a particular numbering of the processors) that for any processor i, there exists some $\alpha_i>0$, independent of k, such that $\Phi^{i1}(n|k)\geq\alpha_i$, for all $n\in[k+(i-1)B+1,\ k+MB]\triangleq I_i$.

To start the induction, consider first the case i=1 and notice that

$$\Phi^{11}(n|k)\geq \prod_{t=k+1}^{n-1} a^{11}(t)\geq \alpha^{n-k-1}\geq \alpha^{MB}\triangleq \alpha_1>0, \quad \forall n\in I_1 .$$

*Since each A(n) is a "stochastic matrix (nonnegative entries, each row sums to 1), questions of convergence of $\Phi(n|k)$ are equivalent to questions about the long-run behavior of a finite (time-varying) Markov chain.

Suppose now that a subset S of the processors has been numbered $\{1,\ldots,i-1\}$, for some $i\geq 2$, and that the induction hypothesis has been proved for all processors in S. We show that it is always possible to find a new processor in $V\backslash S$, rename it to i and prove the induction hypothesis for i as well.

Consider the set Q of processors $q\notin S$ such that $(p,q)\in E$, for some $p\in S$. If $Q=\phi$, then S is the set of all processors (because of Assumption 5.2.1b) and we are done. If not, we choose one processor from Q, and rename it to i, subject to the following restriction: we choose a processor with in-degree more than one only if no processor with in-degree equal to one belongs to Q.

We now prove the induction hypothesis for processor i. Let $h\in S$ be some predecessor of i, belonging to S. Then, $h<i$ and, by the induction hypothesis, $\Phi^{hl}(n|k)\geq\alpha_h>0$, $\forall n\in I_h\supset I_{i-1}$. Moreover, for some $t\in[k+(i-2)B+1,\ldots,k+(i-1)B]$ we have $a^{ih}(t)\geq\alpha$ and consequently, $\Phi^{il}(t+1|k)\geq\alpha\alpha_h$.

We first suppose that i has in-degree 1. We prove by induction on n, for $n\in[t+1,\ldots,k+MB]\supset I_i$ that $\Phi^{il}(n|k)\geq\alpha\alpha_h \triangleq \alpha_i>0$. Indeed,

$$\Phi^{il}(n+1|k) = a^{ih}(n)\Phi^{hl}(n|k) + a^{ii}(n)\Phi^{il}(n|k)\geq$$

$$\geq \min\{\Phi^{hl}(n|k),\Phi^{il}(n|k)\}\geq \min\{\alpha_h,\alpha\alpha_h\} = \alpha\alpha_h .$$

Suppose now that i has in-degree more than 1 and that Assumption 5.2.1c(iii) holds. Then,

$$\Phi^{il}(n|k)\geq\left[\prod_{m=t+1}^{n-1} a^{ii}(m)\right]\Phi^{il}(t+1|k)\geq\alpha^{n-t}\alpha_h\geq\alpha^{MB}\alpha_h \triangleq \alpha_i>0,\quad \forall n\in I_i .$$

The last possibility is that i has in-degree more than 1 (hence all processors in Q have in-degree more than 1) and Assumption 5.2.1c(iii) fails. Then the set of predecessors of i (denoted by U) has a single common predecessor, denoted by j. Since $i \in Q$, some $h \in U$ must belong to S. Since any $h \in U$ is non-computing, we have $h \neq 1$ and its predecessor j must belong to S. Now, for any $p \in U$, p does not belong to Q (since it has in-degree 1) and therefore, $p \in S$. We conclude that all predecessors of i belong to S ($U \subset S$). We now perform an easy induction on n, for $n \in [t+1,\ldots,k+MB]$ to show that $\Phi^{i1}(n|k) \geq \alpha \min_{h \in U} \{\alpha_h\} \triangleq \alpha_i > 0$. Indeed,

$$\Phi^{i1}(n+1|k) = \sum_{h \in U \cup \{i\}} a^{ih}(n)\Phi^{h1}(n|k) \geq \min_{h \in U \cup \{i\}} \Phi^{h1}(n|k) \geq$$

$$\geq \min\{\alpha_i, \min_{h \in U} \{\alpha_h\}\} = \alpha_i \quad .$$

This completes our inductive argument.

We may now conclude that $\Phi(k+MB|k)$ is a stochastic matrix with the property that all entries in some column (corresponding to any computing processor) are positive and bounded away from zero by a constant $\bar{\alpha} > 0$ which does not depend on k. We combine this fact with Lemma A.1 below to conclude that the assertions of Lemma 5.2.1 are true.

<u>Lemma A.1</u>: Consider a sequence $\{D_n\}$ of nonnegative matrices with the properties that:

(i) Each row sums to 1.

(ii) For some $\bar{\alpha} > 0$ and for some column (say the first one), all entries of D_n, for any n, in that column are larger or equal than $\bar{\alpha}$.

Then,

a) $\bar{D} = \lim_{n\to\infty} \prod_{k=1}^{n} D_k$ exists.

b) All rows of $\bar{D}$ are identical.

c) The entry in the first column of $\bar{D}$ is bounded below by $\bar{\alpha}$.

d) Convergence to $\bar{D}$ takes place at the rate of a geometric progression.

<u>Proof of Lemma A.1:</u> Given any vector $x=(x_1,\ldots,x_M)$ we decompose it as $x=y+ce$, where c is a scalar, e is the vector with all entries equal to 1 and y has one zero entry and all other entries are nonnegative. (So c equals the minimum of the components of x.

Let $x(n) = \prod_{k=1}^{n} D_k x(0)$, $\forall n$ and $x(n)=y(n)+c(n)e$. It is easy to see that $||y(n+1)||_\infty \leq (1-\bar{\alpha})||y(n)||_\infty$, where $||\cdot||_\infty$ denotes the max-norm on R^M, which shows that $y(n)$ converges geometrically to zero. Moreover, $c(n)\leq c(n+1)\leq c(n) + ||y(n)||_\infty$, which shows that $c(n)$ also converges geometrically to some $\bar{c}$. Hence, $x(n)$ converges geometrically to $\bar{c}e$. Since this is true for any $x(0)\in R^n$, parts (a),(b),(d) of the Lemma follow. Part (c) is proved by an easy induction, for the finite products of the D_k's and, therefore, it holds for $\bar{D}$ as well. □

We now consider the case where Assumption 5.2.2 is replaced by 5.2.3. The key observation here is that during an interval of the form $[B_1 n^\beta, B_1(n+1)^\beta]$ a bounded number of messages is transmitted; hence, $A(k)=I$, except for a bounded number of times in that interval. If we redefine the time variable so that time is incremented only at communication times, we have reduced the problem to the case of Assumption 5.2.2. The only difference, due to the change of the time variable,

is in the rate of convergence. Under Assumption 5.2.2, $||\Phi(n|k)-\Phi(k)||$ decreases by a constant factor during intervals of constant length. This implies that, under Assumption 5.2.3, $||\Phi(n|k)-\Phi(k)||$ decreases by a constant factor during intervals of the form $[B_1 t^{\beta}, B_1(t+c)^{\beta}]$, for some appropriate constant c. Therefore, if $B_1 t^{\beta}=k$ and $B_1(t+m)^{\beta}=n$, we have $||\Phi(n|k)-\Phi(k)|| \leq Bd_o^{m/c}$, for some $B \geq 0$, $d_o \in [0,1)$. Eliminating t and solving for m, we obtain

$$\left(\frac{k}{B_1}\right)^{1/\beta} + m = \left(\frac{n}{B_1}\right)^{1/\beta} ,$$

which finally yields

$$||\Phi(n|k)-\Phi(k)|| \leq Bd_o^{\left[\left(\frac{n}{B_1}\right)^{1/\beta} - \left(\frac{k}{B_1}\right)^{1/\beta}\right]/c} .$$

Let $\delta=1/\beta$ and $d=d_o^{\frac{1}{cB_1^{1/\beta}}}$, to recover the desired result. ■

REFERENCES

Aho, A.V., J.D. Ullman, M. Yannakakis, (1983), "On Notions of Information Transfer in VLSI circuits," Proc. Fifteenth Annual ACM Symposium on the Theory of Computation, pp. 133-139.

Anderson, B.D.O., D.J. Clements, (1981), "Algebraic Characterization of Fixed Modes in Decentralized Control," Automatica, Vol. 17, No. 5, pp. 703-712.

Anderson, B.D.O., J.B. Moore, (1981), "Time-Varying Feedback Laws for Decentralized Control," IEEE Transactions on Automatic Control, Vol. AC-26, No. 5, pp. 1133-1138.

Aoki, M., (1973), "On Decentralized Linear Stochastic Control Problems with Quadratic Cost," IEEE Transactions on Automatic Control, Vol. AC-18, pp. 243-250.

Arrow, K.J., L. Hurwicz, (1960), "Decentralization and Computation in Resource Allocation," Essays in Economics and Econometrics, R.W. Pfouts, Ed., Univ. of North Carolina Press, Chapel Hill, NC, pp. 34-104.

Arrow, K.J., R. Radner, (1979), "Allocation of Resources in Large Teams," Econometrica, Vol. 47, No. 2, pp. 361-385.

Astrom, K.J., P. Eykhoff, (1971), "System Identification - A Survey," Automatica, Vol. 7. pp. 123-162.

Aumann, R.J., (1976), "Agreeing to Disagree", The Annals of Statistics, Vol. 4, No. 6, pp. 1236-1239.

Avriel, M., (1976), Nonlinear Programming, Prentice-Hall, Englewood Cliffs, NJ.

Bagchi, A., T. Basar, (1980), "Team Decision Theory for Linear Continuous Time Systems," IEEE Transactions on Automatic Control, Vol. AC-25, No. 6, pp. 1154-1161.

Barta, S.M., (1978), "On Linear Control of Decentralized Stochastic Systems," Ph.D. Thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology.

Baudet, G.M., (1978), "Asynchronous Iterative Methods for Multiprocessors," Journal of the ACM, Vol. 25, pp. 226-244.

Bertsekas, D.P., (1982), "Distributed Dynamic Programming," IEEE Transactions on Automatic Control, Vol. AC-27, No. 3, pp. 610-616.

Bertsekas, D.P., (1983), "Distributed Computation of Fixed Points," Mathematical Programming, Vol. 27, pp. 107-120.

Bertsekas, D.P., J.N. Tsitsiklis, M. Athans, (1984), "Convergence Theories of Distributed Iterative Processes: A Survey," submitted to SIAM Review.

Borkar, V., P. Varaiya, (1982), "Asymptotic Agreement in Distributed Estimation," IEEE Transactions on Automatic Control, Vol. AC-27, No. 3, pp. 650-655.

Borodin, A., J.E., Hopcroft, (1982), "Routing, Merging and Sorting on Parallel Modes of Computation," Proceedings of the Fourteenth Annual ACM Symposium on the Theory of Computation, pp. 338-344.

Burton, R.M., B. Obel, (1980), "The Efficiency of the Price, Budget, and Mixed Approaches under Varying A Priori Information Levels for Decentralized Planning," Management Science, Vol. 26, No. 4, pp. 401-417.

Cantor, D.G., M. Gerla, (1974), "Optimal Routing in a Packet Switched Computer Network," IEEE Transactions on Computation, Vol. C-23, pp. 1062-1069.

Chazan, D., W. Miranker, (1969), "Chaotic Relaxation," Linear Algebra and Applications, Vol. 2, pp. 199-222.

Chang, T.S., (1980), "Comments on 'Computation and Transmission Requirements for a Decentralized Linear-Quadratic-Gaussian Control ", IEEE Transactions on Automatic Control, Vol. AC-25, No. 3, pp. 609-610.

Chong, C.Y., (1979), "Hierarchical Estimation," Proceedings of the Second MIT/ONR Workshop on Distributed Communication and Control Problems Motivated by Naval C^3 Systems, Monterey, CA, Vol. 1, pp. 205-220.

Chong, C.Y., M. Athans, (1971), "On the Stochastic Control of Linear Systems with Different Information Sets," IEEE Transactions on Automatic Control, Vol. AC-16, No. 5, pp. 423-430.

Chong, C.Y., M. Athans, (1976), "On the Periodic Coordination of Linear Stochastic Systems," Automatica, Vol. 12, pp. 321-335.

Chu, K.C., (1972) "Team Decision Theory and Information Structures Optimal Control Problems -Part II", IEEE Transactions on Automatic Control, Vol. AC-17, No. 1, pp. 22-28.

Chu, K.C., (1978), "Designing Information Structures for Quadratic Decision Problems," Journal of Optimization Theory and Applications, Vol. 25, No. 1, pp. 139-160.

Davison, E.J., U. Ozguner, (1983), "Characterizations of Decentralized Fixed Modes for Interconnected Systems," Automatica, Vol. 19, No. 2, pp. 169-182.

Delebecque, F., J.P. Quadrat, (1981), "Optimal Control of Markov Chains Admitting Strong and Weak Interactions," Automatica, Vol. 17, pp. 281-296.

Ekchian, L.K., R.R. Tenney, (1982), "Detection Networks," Technical Report LIDS-P-1191, Laboratory for Information and Decision Systems, Massachusetts Institute of Technology.

Ekchian, L.K., (1982), "Distributed Detection and Communication Problems," Ph.D. Thesis, Dept. of Electrical Engineering and Computer Science, MIT, Cambridge, MA.

Findeisen, W., (1982), "Decentralized and Hierarchical Control Under Consistency or Disagreement of Interest," Automatica, Vol. 18, No. 6, pp. 647-664.

Forestier, J.P., P. Varaiya, (1978), "Multilayer Control of Large Markov Chains," IEEE Transactions on Automatic Control, Vol. AC-23, No. 2, pp. 298-304.

Gafni, E.M., D.P. Bertsekas, (1983), "Asymptotic Optimality of Shortest Path Routing Algorithms," Technical Report LIDS-P-1307, Laboratory for Information and Decision Systems, Massachusetts Institute of Technology.

Galbraith, J.R., (1977), Organizational Design, Addision-Wesley.

Gallager, R.G., (1977), "A Minimum Delay Routing Algorithm Using Distributed Computation," IEEE Transactions on Communications, Vol. COM-25, No. 1, pp. 73-85.

Gallager, R.G., (1983), "A New Distributed Shortest Path Algorithm," Technical Report LIDS-P-1276, Laboratory for Information and Decision Systems, Massachusetts Institute of Technology.

Garey, M.R., D.S. Johnson, (1979), Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman and Co., San Francisco.

Garey, M.R., D.S. Johnson, H.S. Witsenhausen, (1982), "The Complexity of the Generalized Lloyd-Max Problem," IEEE Transactions on Information Theory, Vol. IT-28, No. 2, pp. 255-256.

Geanakopoulos, J.D., H.M. Polemarchakis, (1978), "We Can't Disagree Forever," Institute for Mathematical Studies in the Social Sciences, Technical Report, No. 277, Stanford University, Stanford, CA.

Geromel, J.C., J. Bernussou, (1979), "An Algorithm for Optimal Decentralized Regulation of Linear Quadratic Interconnected Systems," Automatica, Vol. 15, pp. 483-491.

Goodwin,G.C., R.L. Payne, (1977), Dynamic System Identification: Experiment Design and Data Analysis, Academic Press.

Groves, T. (1973), "Incentives in Teams," Econometrica, Vol. 41, No. 4, pp. 617-631.

Groves, T., M. Loeb, (1979), "Incentives in a Divisionalized Firm," Management Science, Vol. 25, No. 3, pp. 221-230.

Haddad, A.H., (1976), "Linear Filtering of Singularly Perturbed Systems," IEEE Transactions on Automatic Control, Vol. AC-21, No.4.

Harsanyi, J.C., (1967), "Games with Incomplete Information Played by 'Bayesian' Players; Part I, The Basic Model," Management Science, Vol. 14, No.3, pp. 159-182.

Harsanyi, J.C., (1968a), "Games with Incomplete Information Played by 'Bayesian' Players: Part II, Bayesian Equilibrium Points," Management Science, Vol. 14, No. 5, pp. 320-334.

Harsanyi, J.C., (1968b), "Games with Incomplete Information Played by 'Bayesian' Players,"; Part III, The Basic Probability Distribution of the Game," Management Science, Vol. 14, No. 7, pp. 486-502.

Hassan, M.F., G. Salut, M.G. Singh, A. Titli, (1978), "A Decentralized Computation Algorithm for the Global Kalman Filter," IEEE Transactions on Automatic Control, Vol. AC-23, No. 2, pp. 262-268.

Ho, Y.C., (1980), "Team Decision Theory and Information Structure," IEEE Proceedings, Vol. 68, No. 6, pp. 644-654.

Ho, Y.C., T.S. Chang, (1980), "Another Look at the Nonclassical Information Problem," IEEE Transactions on Automatic Control, Vol. AC-25, No. 3, pp. 537-540.

Ho, Y.C., K.C. Chu, (1972), "Team Decision Theory and Information Structures in Optimal Control Problems-Part I," IEEE Transactions on Automatic Control, Vol. AC-17, No. 1, pp. 15-22.

Ho, Y.C., M.P. Kastner, E. Wong, (1978), "Teams, Signalling, and Information Theory," IEEE Transactions on Automatic Control, Vol. AC-23, No. 2, pp. 305-312.

Ho, Y.C., N. Papadopoulos, (1979), "Further Notes on Redundancy in Teams," IEEE Transactions on Automatic Control, Vol. AC-24, No. 2, pp. 323-325.

Ho, Y.C., P.B. Luh, R. Muralidharan, (1981), "Information Structure, Stackelberg Games and Incentive Controllability," IEEE Transactions on Automatic Control, Vol. AC-26, No. 2, pp. 454-460.

Jennergren, L.P., (1980), "On the Design of Incentives in Business Firms - A Survey of Some Research," Management Science, Vol. 26, No. 2, pp. 180-201.

Ioannou, P., P. Kokotovic, (1983), Adaptive Control with Reduced Models, Springer Verlag.

Khalil, H.K., P.V. Kokotovic, (1978), "Control Strategies for Decision Makers Using Different Models of the Same System," IEEE Transactions on Automatic Control, Vol. AC-23, No. 2, pp. 289-297.

Kokotovic, P.V., R.E. O'Malley, Jr., P. Sannuti, (1976), "Singular Perturbations and Order Reduction in Control Theory - An Overview," Automatica, Vol. 12, pp. 123-132.

Krainak, J.C., J.L.Speyer, S.I. Marcus, (1982a), "Static Team Problems - Part I: Sufficient Conditions and the Exponential Cost Criterion," IEEE Transactions on Automatic Control, Vol. AC-24, pp. 839-848.

Krainak, J.C., J.L. Speyer, S.I. Marcus, (1982b), "Static Team Problems-Part II: Affine Control Laws, Projections, Algorithms, and the LEGT Problem," IEEE Trans. Automatic Control, Vol. AC-27, No. 4, pp. 848-859.

Krainak, J.C., F.W. Machell, S.I. Marcus, J.L. Speyer, (1982c), "The Dynamic Linear Exponential Gaussian Team Problem," IEEE Transactions on Automatic Control, Vol. AC-27, No. 4, pp. 860-869.

Kung, H.T., (1976), "Synchronized and Asynchronous Parallel Algorithms for Multiprocessors," in Algorithms and Complexity, Academic Press, pp. 153-200.

Kurtaran, B.Z., R. Sivan, (1974), "Linear-Quadratic-Gaussian Control with One-Step-Delay Sharing Pattern," IEEE Transactions on Automatic Control, Vol. 19, pp. 572-574.

Kushner, H.J., D.S. Clark, (1978), Stochastic Approximation Methods for Constrained and Unconstrained Systems, Applied Math. Series, No. 26, Springer-Verlag, Berlin.

Kushner, H.J., A. Pacut, (1982), "A Simulation Study of a Decentralized Detection Problem," IEEE Transactions on Automatic Control, Vol. AC-27, No. 5, pp. 1116-1119.

Lauer, G.S., N.R. Sandell, Jr., (1983), "Distributed Detection of Signal Waveforms in Additive Gaussian Observation Noise," Technical Paper 160, Alphatech Inc., Burlington, MA.

Levy, B.C., D.A. Castanon, G.C. Verghese, A.S. Willsky, (1983), "A Scattering Framework for Decentralized Estimation Problems," Automatica, Vol. 19, No. 4, pp. 373-384.

Ljung, L., (1977a), "Analysis of Recursive Stochastic Algorithms," IEEE Transactions on Automatic Control, Vol. AC-22, No. 4, pp. 551-575.

Ljung, L., (1977b), "On Positive Real Transfer Functions and the Convergence of Some Recursive Schemes," IEEE Transactions on Automatic Control, Vol. AC-22, No. 4, pp. 539-551.

Ljung, L., (1981), "Recursive Identification," in Stochastic Systems: The Mathematics of Filtering and Identification and Applications, M. Hazewinkel, J.C. Willems (eds), D. Reidel, pp. 247-281.

Ljung, L., T. Soderstrom, (1983), Theory and Practice of Recursive Identification, MIT Press, Cambridge, MA.

Luce, R.D., H. Raiffa, (1957), Games and Decisions, Wiley, NY, 1957.

Looze, D.P., P.K. Houpt, N.R. Sandell,Jr., M. Athans, (1978), "On Decentralized Estimation and Control with Application to Freeway Ramp Metering", IEEE Trans. on Automatic Control, Vol. AC-23, No. 2, pp. 268-275.

Mahmoud, S.M., (1977), "Multilevel Systems Control and Applications: A Survey," IEEE Transactions on Systems, Man and Cybernetics, Vol. SMC-7, pp. 125-143.

Marschak, J., R. Radner, (1972), The Economic Theory of Teams, Yale University Press, New Haven, 1972.

Meerkov, S.M., (1979), "Mathematical Theory of Behavior-Individual and Collective Behavior of Retardable Elements," Mathematical Biosciences, Vol. 43, pp. 41-106.

Mesarovic, M.D., D. Macko, Y. Takahara, (1970), Theory of Hierarchical Multilevel Systems, Academic Press.

Metivier, M., P. Priouret, (1984), "Applications of a Kushner and Clark Lemma to General Classes of Stochastic Algorithms," IEEE Transactions on Information Theory, Vol. IT-30, No. 2, pp. 140-151.

Meyer, P.A., Probability and Potentials, Blaisdell, Waltham, MA.

Miranker, W.L., (1971), "A Survey of Parallelism in Numerical Analysis," SIAM Review, Vol. 13, No. 4, pp. 524-547.

Moore, J.H., (1979), "Effects of Alternative Information Structures in a Decomposed Organization: A Laboratory Experiment," Management Science, Vol. 25, No. 5, pp. 485-497.

Nemirovsky, A.S., D.B. Yudin, (1983), Problem Complexity and Method Efficiency in Optimization, J. Wiley, New York.

Papadimitriou, C.H., Sipser, M., (1982), "Communication Complexity," Proceedings of the 14th STOC, pp. 196-200.

Papadimitriou, C.H., Steiglitz, K., (1982), Combinatorial Optimization: Algorithms and Complexity, Prentice-Hall.

Papadimitriou, C.H., J.N. Tsitsiklis, (1982), "On the Complexity of Designing Distributed Protocols," Information and Control, Vol. 53, No. 3, pp. 211-218.

Papadimitriou, C.H., J.N. Tsitsiklis, (1984), "Intractable Problems in Control Theory," submitted to the SIAM Journal on Optimization and Control.

Papanicolaou, G., D. Stroock, S. Varadhan, (1977), "Martingale Approach to Some Limit Theorems," Proceedings Conf. on Turbulence, Duke University Press.

Papavassilopoulos, G.P., (1983), "On the Optimal Choice of Measurements in Linear Quadratic Gaussian Team Problems," Proceedings of the 1983 American Control Conf., San Francisco, CA, pp. 693-701.

Phillips, R.G., P.V. Kokotovic, (1981), "A Singular Perturbation Approach to Modeling and Control of Markov Chains," IEEE Transactions on Automatic Control, Vol. AC-26, No. 5, pp. 1087-1094.

Poljak, B.T., Z. Tsypkin, (1973), "Pseudogradient Adaptation and Training Algorithms," Avtomatika i Telemekhanika, No. 3, pp. 45-68.

Poljak, B.T., (1976), "Convergence and Convergence Rate of Iterative Stochastic Algorithm. I. General Case," Avtomatika i Telemekhanika, No. 12, pp. 83-94.

Poljak, B.T., (1977), "Convergence and Convergence Rate of Iterative Stochastic Algorithms. II. The Linear Case," Avtomatika i Telemekhanika, No. 4, pp. 101-107.

Poljak, B.T., (1978), "Nonlinear Programming in the Presence of Noise," Mathematical Programming, Vol. 14, pp. 87-97.

Radner, R., (1962), "Team Decision Problems," Annals of Mathematical Statistics, Vol. 33, pp. 857-881.

Radner, R., (1979), "Rational Expectations Equilibrium: Generic Existence and the Information Revealed by Prices," Econometrica, Vol. 47, No. 3, pp. 655-678.

Roth, A.E., (1979), Axiomatic Models of Bargaining, Springer-Verlag, N.Y.

Sandell, N.R.,Jr., M. Athans, (1974), "Solutions of Some Nonclassical LQG Decision Problems," IEEE Transactions on Automatic Control, Vol. AC-19, No. 2, pp. 108-116.

Sandell, N.R., Jr., (1976), "Decomposition versus Decentralization in Large Scale System Theory," Proceedings of the 15th Conference on Decision and Control, pp. 1043-1046.

Sandell, N.R., Jr., P. Varaiya, M. Athans, M. Safonov, (1978), "Survey of Decentralized Control Methods for Large Scale Systems," IEEE Transactions on Automatic Control, Vol. AC-23, No. 2, pp. 108-128.

Sanders, C.W., E.C. Tacker, T.D. Linton, (1974), "A New Class of Decentralized Filters for Interconnected Systems," IEEE Transactions on Automatic Control, Vol. AC-19, pp. 259-262.

Sanders, C.W., E.C. Tacker, T.D. Linton, R.Y.-S. Ling, (1978), "Specific Structures for Large Scale State Estimation Algorithms Having Information Exchange," IEEE Trans. on Automatic Control, Vol. AC-23, No. 2, pp. 255-261.

Sezer, M.E., D.D. Siljak, (1981), "Structurally Fixed Modes," Systems and Control Letters, Vol. 1, No. 1, pp. 60-64.

Siljak, D.D., (1983), "Complex Dynamic Systems: Dimensionality, Structure and Uncertainty," Large Scale Systems, Vol. 4, pp. 279-294.

Simon, H.A., A. Ando, (1961), "Aggregation of Variables in Dynamic Systems," Econometrica, Vol. 29, pp. 111-138.

Simon, H.A., (1980), The Sciences of the Artificial, MIT Press, Cambridge, MA.

Singh, M.G., (1981), Decentralized Control, North Holland, Amsterdam.

Solo, V., (1979), "The Convergence of AML," IEEE Transactions on Automatic Control, Vol. AC-24, pp. 958-963.

Speyer, J.L., (1979), "Computation and Transmission Requirements for a Decentralized Linear-Quadratic-Gaussian Control Problem," IEEE Transactions on Automatic Control, Vol. AC-24, No. 2, pp. 266-269.

Speyer, J.L., S.I. Marcus, J.C. Krainak, (1980), "A Decentralized Team Decision Problem with an Exponential Cost Criterion," IEEE Transactions on Automatic Control, Vol. AC-25, No. 5, pp. 919-924.

Teneketzis, D., N.R. Sandell,Jr., (1977), "Linear Regulator Design for Stochastic Systems by a Multiple Time Scales Method," IEEE Trans. on Automatic Control, Vol. AC-22, pp. 615-621.

Teneketzis, D., (1982), "The Decentralized Quickest Detection Problem," Proceedings of the 22nd Conference on Decision and Control.

Teneketzis, D., (1983), "The Decentralized Wald Problem," Proceedings of the 1983 American Control Conference, San Francisco, CA.

Teneketzis, D., P. Varaiya, (1984), "Consensus in Distributed Estimation with Inconsistent Beliefs," submitted to Systems and Control Letters.

Tenney, R.R., N.R. Sandell,Jr., (1981), "Detection with Distributed Sensors," IEEE Transactions on Aerospace and Electronic Systems, Vol. AES-17, No. 4, 501-509.

Tenney, R.R., (1983), "Optimal Decentralized Control of Finite Nondeterministic Systems," Proceedings of the 1983 American Control Conference.

Traub, J.F., Wozniakowski, H., (1980), A General Theory of Optimal Algorithms, Academic Press.

Tsitsiklis, J.N., M. Athans, (1984), "Convergence and Asymptotic Agreement in Distributed Decision Problems," IEEE Transactions on Automatic Control, Vol. AC-29, No. 1, pp. 42-50.

Tsitsiklis, J.N., M. Athans, (1985), "On the Complexity of Decentralized Decision Making and Detection Problems," to appear in the IEEE Trans. on Automatic Control.

Tsitsiklis, J.N., D.P. Bertsekas, M. Athans, (1984), "Distributed Asynchronous Deterministic and Stochastic Gradient Optimization Algorithms," submitted to the IEEE Transactions on Automatic Control.

Tsitsiklis, J.N., D.P. Bertsekas, (1984), "Distributed Asynchronous Optimal Routing in Data Networks," Proceedings of the 23d Conference on Decision and Control, Las Vegas, Nevada.

Ullman, J.D., (1984), Computational Aspects of VLSI, Computer Science Press, Rockville, MD.

Varaiya, P., (1972), "Review of Theory of Hierarchical Multilevel Systems," IEEE Trans. on Automatic Control, Vol. AC-17, pp. 280-281.

Varaiya, P., J. Walrand, (1978), "On Delayed Sharing Patterns," IEEE Trans. on Automatic Control, Vol. AC-23, No. 3, pp. 443-445.

Von Neumann, J., O. Morgenstern, (1953), Theory of Games and Economic Behavior, Princeton University Press, Princeton, NJ.

Vorobev, N.N., (1977), Game Theory: Lectures for Economists and Systems Scientists, Springer-Verlag, NY.

Wang, S.H., E.J. Davison, (1973), "On the Stabilization of Decentralized Control Systems," IEEE Transactions Control, Vol. AC-18, pp. 473-478.

Washburn, R.B., D. Teneketzis, (1984), "Asymptotic Agreement among Communicating Decisionmakers," Stochastics, Vol. 13, pp. 103-129.

Willsky, A.S., M. Bello, D.A. Castanon, B.C. Levy, G. Verghese, (1982), "Combining and Updating of Local Estimates and Regional Maps along Sets of One-Dimensional Tracks," IEEE Trans. on Automatic Control, Vol. AC-27, No. 4, pp. 799-813.

Witsenhausen H.S., (1968), "A Counterexample in Stochastic Optimum Control," SIAM J. Control, Vol. 6, No. 1, pp. 138-147.

Witsenhausen, H.S., (1971), "On Information Structures, Feedback and Causality," SIAM J. Control, Vol. 9, No. 2, pp. 149-160.

Witsenhausen, H.S., (1976), "The Zero Error Side Information Problem and Chromatic Numbers," IEEE Trans. on Information Theory, Vol. IT-22, No. 5, pp. 592-593.

Witsenhausen, H.S., (1981), "On Team Guessing with Independent Information," Mathematics of Operations Research, Vol. 6, No. 2, pp. 293-304.

Yao, A.C.-C., (1979), "Some Complexity Questions Related to Distributed Computing," Proceedings of the 11th STOC, pp. 209-213.

Yao, A.C.-C., (1981), The Entropic Limitations of VLSI Computations, Proceedings of the 13th STOC, pp. 308-311.

Yosida, K., (1980), Functional Analysis, Springer-Verlag.