5

Modeling and Design of Distributed Intelligence Systems

Alexander H. Levis
Department of Systems Engineering
George Mason University
Fairfax, VA 22030

Abstract

Elements of a mathematical theory of distributed intelligence systems are presented and used to model and design systems that have to meet stringent structural requirements. The formalism of Petri Nets and Colored Petri Nets is used to depict intelligent nodes and the various types of interconnections between them. Both fixed structure and variable structure architectures are considered. Measures of performance are introduced and various qualitative properties are described.

1. INTRODUCTION

Human organizations are by definition distributed intelligence systems. The characterization applies regardless of which definition is used for the terms distributed and intelligence. A common dictionary defines intelligence as the capacity for reasoning, understanding, or for similar forms of mental activity - clearly, human characteristics. Distributed means that some resource - intelligence in this case - is dispersed through a space or an area.

A more interesting set of definitions is given by Minsky [1]. He defines distributed processes as those in which each function is spread out over a range of locations, so that each part's activity contributes a little to each of several different functions. Substitute "organization member" for "part" and the relevance of the definition is clear. His definition of intelligence, however whimsical, is apt when applied to human organizations - it is all the mental skills that, at any particular moment, we admire but don't yet understand.

In this chapter, the beginnings of a mathematical theory of distributed intelligence systems is presented with emphasis on the concepts, on some of the insights obtained, and on many of the challenges that remain. The point of view taken is that of organization theory. Drenick [2] states in his book on a mathematical organization theory that the objective of such a theory is "to derive certain conclusions from a set

of assumptions by mathematical reasoning." There are two consequences of this statement.

The first is that the assumptions must be stated explicitly and unambiguously. In the beginning, the assumptions are, by necessity, very restrictive since one is dealing with such a complex phenomenon as distributed intelligence in an organization. Then, one carefully and systematically proceeds to weaken the restrictive assumptions so that more aspects of organizational performance can be captured.

The second consequence is that this approach can lead to model-based experimentation: the mathematical theory can give rise to a set of testable hypotheses around which controlled experiments can be designed. With this empirical evidence and the mathematical theory, one can then begin to address the design problem. The nature of the problem is such that a synthesis is required of concepts, methods, and tools from such fields or disciplines as system theory, information theory, computer science, management science, and cognitive and behavioral psychology.

Galbraith [3] describes the various approaches to organization design as follows: In classical management theory, the key concept is that of division of labor. A modern counterpart of that approach is the functional decomposition of tasks and the subsequent allocation of those to the organization members so that some objective function is maximized. Parallel, hierarchical, and mixed structures result. In contrast to this mechanistic approach is the human relations approach in which the style of leadership determines the organizational form. Empirically based, it leads to the consideration of incentives - tangible and intangible - as the means to improve performance. The third approach, which is closest to the conceptual framework for distributed intelligence systems, is based on the view of the organization as an information processing and decision making system. The cognitive limitations of humans (or the memory and information processing limitations of machines) determine the organizational form. This approach admits the allocation of functions to humans and machines. If the decision problems can be formulated analytically and if the proper data are available in the proper form, then the solution can be obtained algorithmically. In the idealized case, classical rationality is used.

But, as more operational problems under more realistic conditions are considered, the classical rationality model does not hold. Even if a clearly defined, commonly agreed upon objective function is identified, there is never enough time to enumerate the possible options, evaluate them, and select the best. Time constraints are usually the most potent reason for violating classical rationality. More subtle reasons also arise. For example, the data available for the evaluation of two competing options may not be consistent and concurrent, thus making comparison not strictly valid. These considerations apply not only to systems where humans are an integral part, such as air traffic control centers; they also apply to flexible manufacturing plants where some robots and flexible manufacturing cells can be considered as intelligent nodes.

As a result, to improve organizational performance in distributed intelligence systems with human decision makers as some of the nodes, decision aids have been introduced that sometimes aim at reducing the decision maker's workload by carrying out mundane, but time consuming tasks such as evaluation of alternatives; sometimes aim at augmenting the decision maker's scope by introducing additional options; and

sometimes aim at reducing human error by providing guidance through a checklist or a step by step procedure. In all cases, they have complicated the organization design problem significantly. The information processing and decision making functions are now distributed not only among the intelligent organization members, but also between humans and machines. This is the design challenge posed by considering human organizations as distributed intelligence systems.

Key assumptions are presented first followed by the model of an intelligent node. In the next section, the node model is used to construct distributed systems. Then measures of performance will be described and the models used in their evaluation outlined. Finally, some of the results obtained to date will be discussed - in particular, consequences of the distributed nature of the cognitive processes that represent distributed intelligence.

2. ASSUMPTIONS

A restricted class of organizations will be considered. It is assumed first that the system contains of at least two intelligent nodes and that the nodes operate as a team. A team is defined as an organization in which the nodes have a common goal and have activities that must be coordinated so as to achieve a higher effectiveness [4]. In the case of humans as nodes, it is further assumed that they have the same interests and same beliefs, are well trained for the tasks that they have to perform, and that they do not learn during the execution of a particular task.

It should be possible to draw a boundary that defines what is included in the distributed intelligence system (DIS) and what is excluded, i.e., what resides in the external environment. Tasks that the DIS must perform are generated in the environment by one or more sources which may or may not be synchronized. The system acts upon these inputs and produces a response, including the null response, that is directed to the environment. Thus, the interface between the system and the environment is composed of the sensors and the effectors. Whether to include the sensors or the effectors or both as parts of the organization or as parts of the environment is a question that must be addressed in each particular case. This issue becomes relevant when alternative organizational designs are evaluated; the comparison must be done on the same basis.

The elements of the distributed intelligent system consist of intelligent nodes, such as humans, intelligent machines, or a combination, data bases, processors, and communication systems. A decision aid is defined as any technique or procedure that restructures the methods by which problems are analyzed, alternatives developed, and decisions taken. Decision support systems, a specific form of decision aids, do not automate a specific decision making process, but must facilitate it [5]. Decision support systems are considered here as higher level components that may consist of processors, data bases and communication systems.

Relationships are the links that tie these elements together. These relationships can be considered at three levels: they may describe the physical arrangement of the components - such as the geographical location of the organization members, - or the functional relationship between components - such as the sharing of information

between two nodes, - or the rules and protocols that govern the interactions - such as the conditions under which two nodes may share information. The nature of the relationship determines the type of architecture that is obtained: a physical architecture, a functional architecture, or an operational one. While this demarcation between relationships and components is often hard to justify, it is assumed that it can be done.

3. THE INTELLIGENT NODE

The notion of decomposing a function and assigning its components (or sub-functions) to different nodes is an old one. What the definition of *distributed* makes explicit is that each node contributes to the execution of *several* different functions. Thus, the problem is not solved by doing a simple allocation of a decomposed function to the available resources - human and machine ones. One must allocate several decomposed functions in such a manner that the resulting workload does not exceed the capacity of each node.

The concept of a role forms the basis for modeling the distributed aspects of the system. The role represents the lowest level of functional decomposition for a particular application; a role must be executed in its entirety by a single node. Two types of interactions among roles are defined: (a) those that are among roles within the same node, called internal interactions, and (b) those that are between roles in different nodes, called external interactions. The latter are the ones that determine the architecture of the DIS.

The origins of the model of the role can be traced back to the four stage model of the interacting decision maker with bounded rationality introduced by Boettcher and Levis [6]. Andreadakis and Levis [7] introduced an alternative model that was not based on assigning functions to resources; first, the data flow structure for carrying out a task was determined and then sub-functions were grouped together and assigned to a resource, whether the resource represented a human or a machine. While this was a five stage model, it was very similar to the four stage one in terms of the allowable interactions and led to the formulation of the notion of the role. [8] A role is used to model the execution of a single task by a single resource. The functional decomposition can be carried out to the point that the lowest level tasks must be executed by a single resource.

The basic model of the role can be represented in block diagram form as shown in Fig. 3.1. It consists of three processing stages and two interaction stages for a total of five stages.

A role receives inputs or data x from the external environment (sensors) or from other nodes of a system. The incoming data are processed in the first block marked situation assessment (SA) to obtain the assessed situation z. This variable may be sent to other nodes, as shown by the outgoing arrow. If the role receives data about the assessed situation from other nodes, these data z' are fused together with its own assessment z in the information fusion (IF) stage to obtain the revised assessed situation z". The assessed situation is processed further in the task processing (TP) stage to determine the strategy v to be used to select a response. However, if the

system has embedded functional or organizational hierarchies, the particular role may receive command inputs v' from superordinate nodes (or higher echelon decision making nodes) that may restrict the strategies available for selecting a response. This is depicted by the use of the command interpretation (CI) stage. The output of that stage is the variable w which contains both the revised situation assessment data and the response selection strategy. Finally, the output or response of the role y is generated by the response selection (RS) stage.

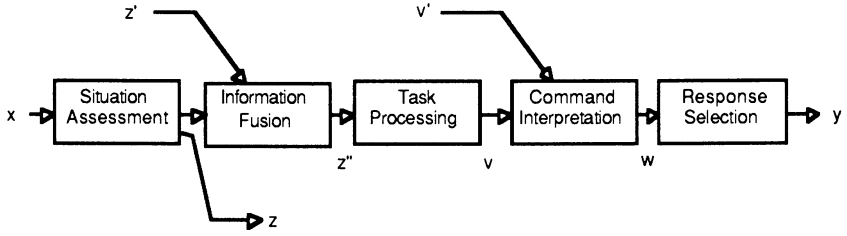

Figure 3.1 The five-stage model of a role.

Note that if there are no situation assessment and command inputs from other nodes, the five stage model reduces to the common two-stage situation assessment and response selection model of the single, non-interacting decision maker. If there is no information fusion, then the IF stage disappears and the SA and TP stages can be merged into a single SA stage. Finally, if there are no command inputs, then the CI stage disappears and the TP and RS stages can be merged into a single RS stage.

If ordinary Petri Net notation is used, the model of Fig. 3.1 takes the form shown in Fig. 3.2. The rounded box enclosing the five stages has no formal meaning; it is used to indicate the internal structure of a role and show explicitly the inputs and outputs of the role. In accordance with Petri Net conventions, [9] transitions are denoted by solid bars and they represent processes or events. Signals or conditions are depicted by places which are shown as circles. Directed arcs indicate the relationships between the two types of nodes. Since Petri Nets are bipartite directed graphs, arcs can exist only from a transition to a place or a place to a transition. A detailed description of Petri Net modeling of decision making organizations is given in Levis [10].

The intelligence in this model is embodied in the algorithms embedded in the transitions of a role; however, even if the algorithms are stochastic, the model is rather mechanistic and does not capture intelligent behavior well. To model the choice inherent in decision making, the decision making entity - a human or an intelligent machine - is assumed to have, at any stage of processing, a set of options: different algorithms processing in different ways the same input to produce the same type of output. Thus, the SA and RS stages of the role of Fig. 3.2 are modeled so as to include a set of U and V algorithms, respectively. The SA and RS stages are represented by a subnet of a Petri Net with switches.

Since a switch is really a transition, the firing rules for a switch are identical to the firing rules for a transition: a switch will fire if all its input places contain at least

one token. Unlike regular transitions however, only one of the output places of a switch will receive a token. This place will be chosen by the internal decision rule embedded in the switch. Note that it is necessary to associate attributes with the tokens so that the rule can differentiate among inputs. In the SA stage, this choice is denoted by the variable u, taking its values in {1, 2,..., U}. The rule for determining

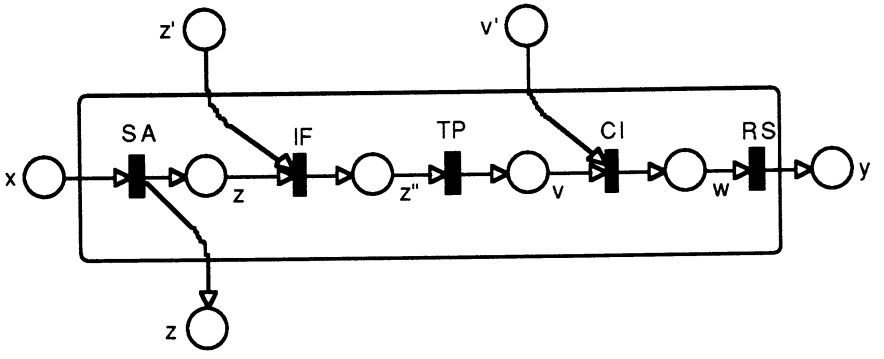

Figure 3.2. Petri Net representation of the five-stage role model.

the value of the decision variable is called the decision strategy of the role for the particular stage. If the rule is characterized by a probability distribution p(u) and if one branch of the switch is always chosen, i.e., if there is an i in {1, ... , U} such that p(u = i) = 1, then the strategy is called pure. Otherwise, it is mixed. The strategy that a rule uses at the RS stage usually depends on the input to that stage. In that case, the probabilities are conditional probabilities p(v = j |z, v). Together, the strategies for the two stages constitute the internal decision strategy of the role. While this is a way of describing the set of strategies that a well trained human decision maker may use, if his bounded rationality threshold is not exceeded, there are no rules to specify how and when any of these strategies will be selected by a specific decision maker at any given time. These rules are assumed to depend on his level of expertise, and on those mental skills that "we admire but don't yet understand," i.e., on his intelligence.

A second way in which intelligence is embedded in a node containing a number of roles is to introduce a switch and a decision rule for the selection of the appropriate role to perform the arriving task. Consider an intelligent node that can instantiate any one of several roles depending on the input it receives. The switch is introduced between the source generating tasks (inputs) and the M roles with a set of rules that directs the particular task to the appropriate role. This is shown schematically in Fig. 3.3 for the case when there are four roles.

Consider the changes that are needed in the modeling of the source. Let the source generate n distinct tasks x; then the set of tasks X can be partitioned into four disjoint subsets X_1 to X_4. One possible rule embedded in the switch may take the form:

If the input $x_i \in X_j$, j = 1 to 4, then Role j is activated.

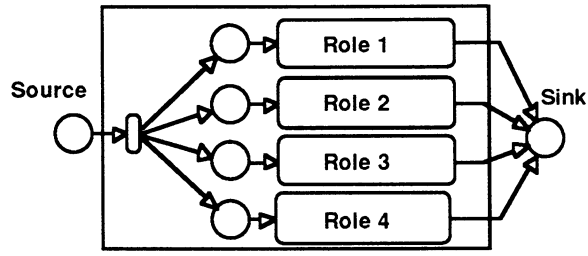

Figure 3.3. Model of Intelligent Node

This particular generalization is straightforward. A task is created by the source, it is directed by the switch to the appropriate role and an output is generated. The complications arise when interactions between intelligent nodes are considered. As Monguillet [11] has shown, the rules governing the switches are not independent. If the interactions between nodes depend on the task (i.e., the type of token that is being processed) then a node i that interacts with node j must know what role node j has assumed so that it can select a compatible one. If this does not happen, then deadlocks can occur. For example, node i chooses a role that does not include the transmission of situation assessment information to node j. On the other hand, node j chooses a role that requires situation assessment information from node i. The IF transition of node j will not be enabled and that node will be deadlocked. To address this problem, a table needs to be created that contains all the admissible combinations (the intercorrelations) of switch settings. The difficulty is that this information is not included as part of the Petri Net formalism; the Petri Net model is then an incomplete description of the DIS.

The answer to this problem is the introduction of Colored Petri Nets, a form of High Level Nets introduced by Jensen [12], tokens are distinguishable. A set of attributes is associated with each token, where each attribute can take a number of values. The color of the token denotes a particular choice of attribute values. All the possible combinations of attribute values (all the colors) constitute the universal color set for a particular problem. There is a Color Set associated with each place in the Petri Net: the Color Set specifies the colors of the tokens that may reside in that place. This Color Set is a subset of the universal color set. Similarly, an Occurrence Color Set is associated with each transition. Arcs are inscribed with expressions of the form:

[Boolean]%Expression

where a Boolean expression is enclosed by the brackets. When this expression evaluates a true, then the arc inscription evaluates to a set of colors according to the normal expression "Expression". A transition is enabled, if there exists one at least one binding for the variables in the inscriptions of the arcs from the input places such that each input place contains at least as many color tokens as specified by the arc inscription. In addition, when the transition contains a guard function, the condition indicated by the guard function must also be satisfied. The following example, Fig. 3.4, demonstrates these definitions.

The universal Color Set contains three colors, red, white, and blue. All three places have as their Color Set the universal set; they can hold all types of tokens. A variable x has been defined that can take values in (be bound to) the same universal

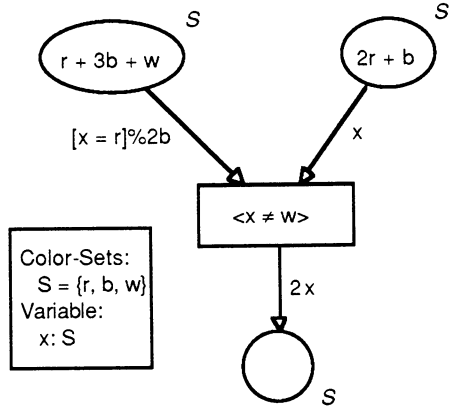

Figure 3.4. Colored Petri Net example

set *S*. The transition has a guard function that restricts the variable x from taking the value w if the transition is to be enabled. The inscriptions translate as follows: If the variable x takes the value r and there is at least one r token in the input place on the right, and if there are two blue tokens in the place on the left, then the transition will fire. Two blue tokens will be removed from the left input place, one red token from the right input place, and two red tokens will be generated in the output place. If x takes the value b, then the right place must have at least one blue token. However, the arc inscription on the left will not evaluate as True and, therefore, there is no enablement condition for that binding of the variable. Finally, because of the guard function, there is no point checking what will happen to the arc inscriptions when x takes the value w.

With this model of the role, it is now possible to formulate the problem of constructing distributed intelligence systems. At this point, a role is the functional element while a node is the physical element that instantiates the role. A node may be able to perform a number of roles, either serially or concurrently.

4. ARCHITECTURES

Interactions Between Roles

It was shown in Figs. 3.2 and 3.3 that a role can only receive inputs at the SA, IF, and CI stages, and produce outputs at the SA and RS stages. These conditions lead to the set of admissible interactions between two roles *in different nodes* that is

shown in Fig. 4.1. For clarity, only the connectors from Role i to Role j are shown; the interactions from j to i are identical.

Consider Role i in Fig. 4.1. The first question to be answered is whether it receives data from the external environment, from the sensors. This is denoted by the coefficient e_i. If e_i is equal to 1, then Role i receives data from sensors, if it is equal to zero, it does not. The coefficient G_{ij} indicates whether the output of Role i is an

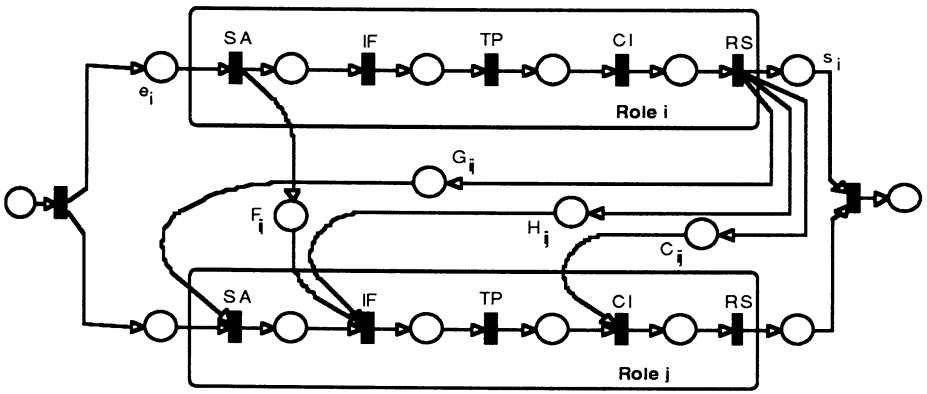

Figure 4.1. Admissible interactions from Role i to Role j

input to the situation assessment stage of Role j. This type of interconnection is needed to represent the tandem (series) connection of roles. The coefficient F_{ij} represents the sharing of the assessed situation z by Role i with Role j. This is one type of information sharing. The second type, the sharing of results, is represented by H_{ij}. In this case, Role i communicates to Role j the output of the response selection stage. The link H_{ij} is used in place of G_{ij} when there is an input from the external environment to Role j. Whether to communicate the assessed situation or to communicate the decision a role has made is an interesting design question that has been addressed by many researchers. The basic trade-off is the amount of data that need to be transmitted - the situation assessment usually requires more bits than the decision. On the other hand, under some rather restrictive conditions, it is possible to reconstruct the assessed situation when the decision is known. The final type of interaction is the issuing of a command from Role i to Role j, as shown by C_{ij} from the response selection stage of i to the command interpretation stage of j. Finally, the coefficient s_i denotes whether Role i produces an output to the environment.

When fixed functional architectures are considered, then each node contains only one role and these six coefficients are constant and take values in {0, 1}. If there are n roles/nodes in a fixed structure DIS, then all the interactions can be represented by a set of six arrays:

$$\Sigma = \{e, F, G, H, C, s\}$$

where e and s are n x 1 and F, G, H, and C are of dimension n x n. From the definition of the arrays (see Fig. 4.1) it is apparent that the diagonal elements of F, G, H, and C are identically zero. Therefore, any possible set of interconnections among n roles can be represented by assigning the value of 0 or 1 to q elements where:

$$q = 2n + (4n^2 - 4n) = 4n^2 - 2n.$$

The number of admissible functional architectures is 2^q. Note that all these structures may not necessarily represent feasible organizations. i.e., organizations that satisfy a number of structural constraints.

The basic structural constraints are:

1. The Petri Net that corresponds to Σ should be connected; a directed path should exist from the single source place to every node of the net and from every node of the net to the single sink place.

The single source and single sink places are modeling artifices used to coordinate the source model and to collect all the outputs of the net into a single node.

2. The net Σ should have no loops; it should be acyclic. Note that this constraint applies only to the basic information flow. Loops can be added to represent resource constraints or coordination conditions.

3. There can be at most one link from the RS stage of node i to another node j, i.e.,

$$G_{ij} + H_{ij} + C_{ij} \leq 1$$

4. Information fusion can take place only at the IF and CI stages. Consequently, the SA stage can receive either inputs from the source model or from a single other node.

5. This last constraint is not necessary; its inclusion eliminates some awkward interactions between nodes.

$$G_{ij} + F_{ij} \leq 1$$

Additional constraints may be introduced to reflect the specifics of a particular application.

Generation of Architectures

The analytical description of the possible interactions between roles in different nodes forms the basis of an algorithm that generates all the architectures that meet the structural constraints 1 - 5 as well as application-specific constraints that may be present. The set of structural constraints that has been introduced rules out a large number of architectures.

The algorithm [13] determines the maximal and minimal elements of the set of designs that satisfy all the constraints by recognizing that each admissible architecture is a union of simple paths - directed paths without loops from the source to the sink. The simple paths are computed using an algorithm based on the s-invariants of Petri Net theory. The solution can be expressed as a set of lattices; the set of greatest and least elements defined by the lattices are the set of maximal and minimal elements of the solution.

Thus, it has become possible to generate a complete class of distributed architectures that can serve as candidate structures. The reduction of the computational complexity of the problem to a tractable level is due to the presence of constraints and on the special structure of the role model.

Variable Structure Architectures

The set of interactions between roles in the model considered thus is fixed. The interactions are always there and, consequently, the structure is fixed. However, in distributed intelligent systems one would expect the structure to be variable so that resources are used efficiently in response to the needs of a specific task. For example, in a human organization, it may be necessary to communicate information to another member only when the latter needs it. Unnecessary communication and the transfer of voluminous irrelevant data can cause severe degradation of performance by increasing the cognitive load of the individual members.

The rules that control the interactions and, therefore, determine which structure is appropriate at any time can be of any kind. However, it is possible to distinguish three types of variability on the basis of the cause for change. The DIS adapts its structure to changes in:

(1) the input it processes.
(2) the environment.
(3) the availability of system resources.

The performance of a system may degrade strongly when individual components are affected; for example, the loss of a communication link in a system with a fixed structure may very well mean that deadlock will occur in the flow of information and the system may cease to function.

These three different types of variability can be related to the properties of *Flexibility* and *Reconfigurability*. Flexibility means that the DIS can adapt to the tasks it has to process, or to their relative frequency of occurrence. Reconfigurability means that it can adapt to changes in its resources or in its objectives. The flexibility and reconfigurability properties are another indication of the intelligence resident in such a system. Clearly, the selection of the proper structure is not done through classical rational decision making.

The Lattice algorithm described earlier was generalized by Demaël and Levis [14] to apply to the class of distributed intelligence systems where the structure varies or adapts in response to the type of input (task) the system is to process. Given, now,

that DIS with fixed structures and variable structures can be generated algorithmically, the question of evaluation or assessment arises.

5. MEASURES OF PERFORMANCE

The input set to a distributed intelligence system has been modeled by a finite alphabet X, where the input x takes its values with a discrete probability distribution p(x). The organization produces a response y, and there is a cost function J for the evaluation of the response. In this context, one can define different measures of performance (MOP's) in order to assess the effectiveness of the DIS in performing its tasks:

Accuracy

This measure of performance, J, evaluates how well the system responses correspond to the desired responses. For each input x, there exists an ideal or desired response y_d. When the system produces y, then $c(y, y_d)$ is a function that assigns a cost to the discrepancy between the desired and the actual response. When the algorithms used are deterministic, there is one response y provided for each input x; otherwise, several responses can be given for the same input x at different times. The actual response also depends on the decision strategy used by each role in each node.

The accuracy of the organization for the given strategy δ is then defined by:

$$J(\delta) = \sum_i p(x_i) \sum_j c(y_j, y_d)\, p(y_j \mid x_i)$$

The evaluation procedure for J is shown in Fig. 5.1.

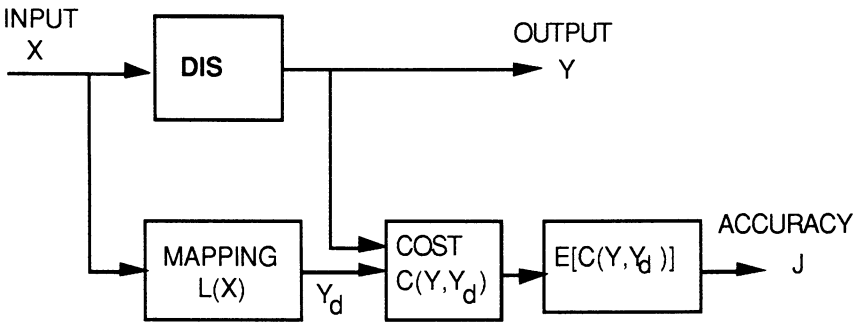

Figure 5.1. Model for evaluating the measure of accuracy.

Timeliness

The timeliness of the response, T, measures the extent to which the responses are provided at the right instants. It is possible to define several measures of timeliness.

One could be, for example, the probability that the response time lies within a certain interval $[T_{min}, T_{max}]$. It can be also the expected delay, which is defined as follows for each strategy δ: for each input x_i, the time delay $r(x_i)$ for processing is computed. The expected delay is then determined by:

$$T(\delta) = \sum_i p(x_i)\, r(x_i)$$

These two measures are appropriate for all types of systems that perform tasks. But when distributed intelligent systems are considered, three more measures can be introduced [4]: *synchronization, information consistency, and coordination.*

Synchronization

The concept of synchronization in DISs is related to the fact that some of the activities that take place are asynchronous and concurrent. Therefore, the various roles may work in parallel and interact at instants that are not predetermined. The protocols of interaction can take various forms: for example, Role 1 may be requested to always wait for instructions from Role 2 before proceeding with any further action; on the other hand, a third role, Role 3, may be allowed to decide whether or not it will wait for these instructions depending on the context of tasks.

Synchronization is related to biases due to the value of the information when the role actually processes it. (The use of old data may cause a degradation of performance; while waiting for the updated information may also cause performance degradation.) Therefore, a DIS is perfectly synchronized when no role has to wait to receive the information that it needs.

A quantitative measure of synchronization is the expected value of the sum of the maximum delays at each process over all the processes in the system for all inputs x. The delays can be computed from the analysis of the Petri Net model of the system: these delays are the extra time that it takes for a transition to be enabled because of the unavailability of a needed token from another role/node.

However, perfect synchronization does not imply necessarily that the delay for the processing of one input will be short. Two roles can be perfectly synchronized but be very slow in executing their tasks; they can also be not perfectly synchronized, but very fast. In the same way, good synchronization will not ensure that the system response will be perfectly accurate: indeed, in order to be well synchronized, one role might use procedures (or algorithms) which introduce important biases in its response.

Synchronization characterizes, in a way, the timeliness of the distributed system. It provides some information on the dynamics of the interactions. Note that the time delays need not be fixed; they can be modeled by random variables. In that case, the Petri Nets are Stochastic Time Petri Nets (STPN). However, all STPN can be considered as a special class of Colored Petri Nets where one of the attributes is time.

Consistency of Information

Consistency of information assesses the extent to which different items of information can be fused together without contradiction. Therefore, it is essentially task dependent. One can evaluate the extent to which two data are inconsistent, if the subsequent processing or actions that will take place for each item of information are not the same. Thus, one cannot speak of the inconsistency of information in general, but must refer to the context and the task at hand.

If the distributed sensors provide different data about the same events, or if different roles arrive at different situation assessments, the inconsistency must be overcome for the overall system to provide a coherent response.

A quantitative measure of information consistency has been developed, again based on the description of the organization by Colored Petri Nets. Since in these nets tokens have identity, they can be marked by their arrival time, the task to which they belong, and some other attribute that characterizes the data they contain - they can belong to different classes. The degree of information consistency for a transition measures the extent to which the tokens enabling that transition are of the same class; the expected value of this measure over all transitions and all inputs is the system measure of information consistency. Clearly, this measure is related to accuracy; the latter, however, considers only the final output.

Coordination

Coordination has been defined as a composite measure. The firing of a transition (or the execution of a process) is coordinated, if it is consistent and synchronized. Consequently, the execution of a task is coordinated if its component processes are coordinated, and a DIS is coordinated, if and only if it is coordinated for all the tasks performed.

All the values of these measures are obtained for a specific decision strategy. As the whole space of decision strategies is spanned, the corresponding values of the measures are obtained. This can be interpreted as a mapping from the strategy space to the performance space. The locus of points in the performance space characterizes the particular design. By comparing qualitatively and quantitatively the loci of different distributed intelligence systems with respect to each other and against requirements, evaluations and trade-off analyses can be carried out. The important aspect is that the locus represents the mapping of all admissible strategies since the rules by which intelligent roles may use to select strategies are not known. The procedure can be refined by allocating different probabilities to the use of different strategies to reflect decision making styles and preferences. However, this has limited usefulness until relevant data are obtained.

6. RESULTS

The value in the development of a mathematical theory of distributed intelligence systems is twofold: to analyze such systems and discover phenomena and behaviors that can be tested in experiments and that can provide insight in the causal

relationships that govern them, and to design DIS that can meet, with predictability, performance requirements.

At this stage in the development of the theory, results have been obtained that have either provided plausible explanations for behavior observed in practice, or are appropriate for the formulation of experimental hypotheses. Some of these results will be described, but only qualitatively. The complete analyses are available in the literature.

Fixed and Variable Structure Organizations

First results on the comparison of fixed and variable structure organizations have confirmed what has been expected. A variable structure organization can perform well over a wide variety of conditions. But, for specific conditions, one can design a fixed structure organization that can perform better. Or, as seen from the requirements point of view, one can design a fixed structure decision making organization (FDMO) that can meet accuracy (J) requirements, or timeliness requirements (T), but one may not be able to design one that meets both types of requirements. However, it is possible to design a variable structure (VDMO) one that meets both requirements. But the flexibility inherent in the VDMO is obtained at a cost. The VDMO may not be able to achieve the best accuracy achievable by an optimized FDMO, or the best speed of response possible by another FDMO. This is shown clearly in Fig. 6.1, where the performance loci of three organizational designs are shown [11].

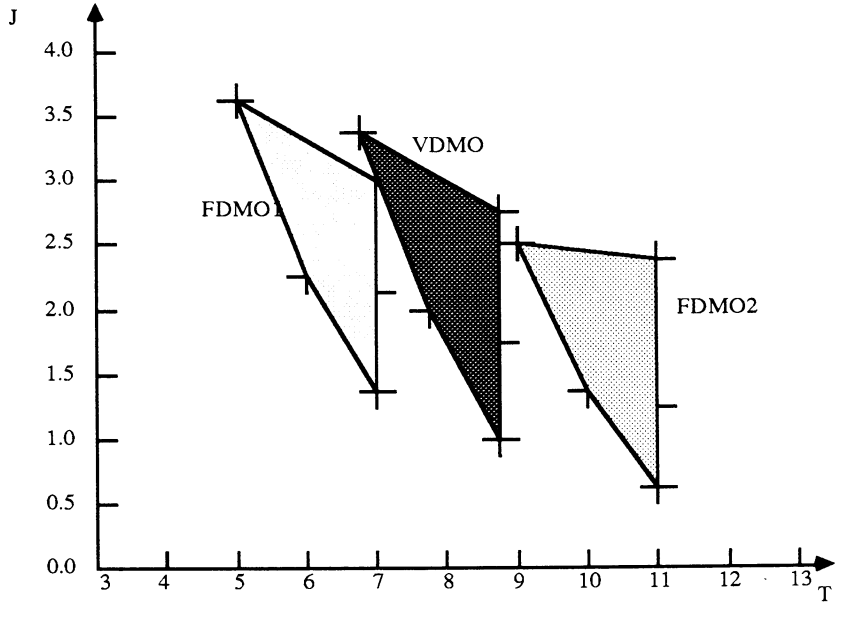

Figure 6.1. Comparison of fixed and variable structure organizations.

FDMO1 optimizes speed of response at a sacrifice in accuracy; FDMO2 optimizes accuracy but at a cost in time. The VDMO achieves intermediate levels of accuracy and speed of response. It can never achieve the highest accuracy possible or the fastest response possible by FDMO2 and FDMO1, respectively, but does not attain the worst values either. Unfortunately, the flexibility inherent in variable structure organizations is achieved through an increase in processing workload. A higher-level cognitive task has been introduced - the management of variability. If this task is performed by humans, then there is the possibility of overload with attendant performance degradation.

This phenomenon that has appeared again and again in the analysis of distributed decision making organizations has two consequences. One is psychological and one is organizational. The first one can be described by the term *metadecision making*. The second leads to the introduction of intelligent decision aids.

Metadecisions

When the roles are carried out by human decision makers, in addition to the decisions that pertain directly to the task, the humans also have to make decisions about how to decide, or, as Einhorn and Hogarth [15] have described it, deciding how to choose. This metadecision making, introduces a cognitive processing "overhead" and has often had a deleterious effect on performance, not only because scarce human cognitive resources are used, but also because it shifts the domain of decisions from the one in which the decision maker is an expert one to one in which he may be less well trained or less expert [16].

Expert Systems

One approach that is often proposed to address this problem is the introduction of decision support systems, especially those based on the expert system model in artificial intelligence. The rationale is obvious: human and artificial (or machine) intelligence are now distributed within the organization with the expected result that higher performance can be achieved or more stringent requirements met.

In some recent work, Perdu and Levis [17] modeled expert system decision aids using the Predicate Transition Net form of High Level Petri Nets. These models were integrated in the Petri Net description of the organization and the overall design could then be evaluated in a consistent manner. The case of two decision makers (DMs) carrying out a data fusion task was considered. Three basic strategies were investigated. DM1 ignores the information provided by DM2; DM1 uses a weighted estimate of the two assessments by taking into consideration the way data used to make these assessments were obtained by each DM; and DM1 uses an expert system. Two implementations of the expert system were modeled; one based on boolean logic and one on fuzzy logic.

The results are shown in Fig. 6.2. Again, it is clear that improved accuracy can be obtained at the cost of longer response time. Therefore, if the response time requirements are too stringent, the expert system will be useless because too much time is needed to perform the data fusion task.

Migration of Control

Another aspect of distributed intelligent system behavior that appears in organizations is the apparent migration of control [18], [19]. In distributed intelligent systems, one can study the effect of the decisions made by each node on the system's performance. This information is readily obtained by considering isoquants on the performance locus: for example, if the decision strategies of all nodes but one are held constant, then the locus of performance when the strategies of the last one only are varied represents the effect that node has on performance. (This is the way the software that has been developed constructs the performance loci). By observing the curvature of the loci, one can determine both qualitatively and quantitatively (by computing partial derivatives) the node whose strategy choice affects the system's performance most.

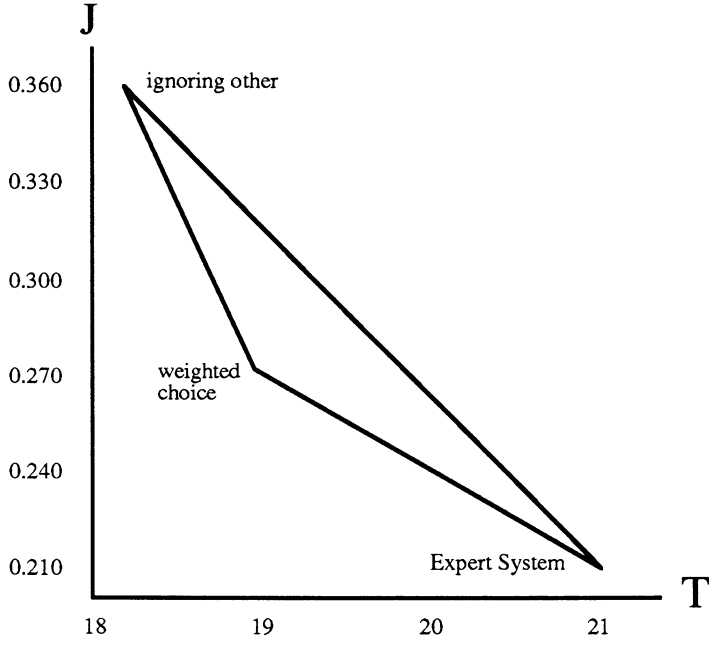

Figure 6.2. Accuracy and timeliness trade-off.

Changes in the structure of the particular DIS through the addition of resources - access to data bases or decision support systems - cause a change in the shape of the locus and, consequently, on the sensitivity of the performance measures to the actions or decisions of the different roles/nodes.

Dynamic Phenomena

The effects that have been described thus far characterize the steady state performance of distributed intelligent systems. Some very interesting dynamic phenomena have

also been observed and analyzed with the use of a simulation system based on Colored Petri Nets [20]. It has been observed in simulations that the introduction of additional resources to aid a specific role in a given node may cause the synchronization between roles to be degraded with an attendant reduction in performance. Or, when different nodes use different protocols, individual tasks may become blocked at a specific node and never be processed, while other later tasks are performed. This is not a system deadlock; the phenomenon appears only when the tokens are given identities or colors that affect the execution of the protocols. At this time, these observations are based on simulation results. Efforts are being made by a number of researchers worldwide to extend invariant theory to Colored Petri Nets and to develop appropriate algorithms for computing the invariants. Efficient methods for constructing the occurrence graphs for Colored Petri Nets would be very helpful.

7. CONCLUSIONS

The study of distributed intelligent systems and the development of mathematical theories for their analysis and design not only is of interest to organization designers, but also to the designers of physical distributed intelligent systems. The theory, even in its early stages of development, is showing that these systems can exhibit a wide variety of not well understood behaviors, both in steady state and in transient state. Furthermore, the understanding of the properties of systems in which both human and machine intelligence co-exist and interact should continue to be a priority, since more and more such systems are being designed and implemented today.

8. ACKNOWLEDGMENT

This work was carried out with support provided by the Office of Naval Research under contract No. N00014-91-J-1584.

REFERENCES

[1] M. Minsky, *The Society of Mind,* New York: Simon and Schuster, 1986.

[2] R. F. Drenick, R. F., *A Mathematical Theory of Organizations.*, New York: North-Holland, 1986.

[3] J. R. Galbraith, *Organization Design.*, Reading, MA: Addison-Wesley, 1977.

[4] J. L. Grevet and A. H. Levis, "Coordination in organizations with decision support systems," in *Proc. 1988 Symp. on C2 Research*, McLean, VA: SAIC, 1988.

[5] P. G. W. Keene and M. S. Scott Morton, *Decision Support Systems: An Organizational Perspective,* Reading, MA: Addison-Wesley, 1978.

[6] K. L. Boettcher and A. H. Levis, "Modeling the interacting decision maker

with bounded rationality," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-12, pp. 334-344, 1982

[7] S. K. Andreadakis and A. H. Levis, "Design methodology for command and control organizations," *Proc. 1987 Symposium on C2 Research,* McLean, VA: SAIC, 1987

[8] A. H. Levis, "A colored Petri Net model of intelligent nodes," *Proc. IMACS Symposium on Modeling and Control of Technological Systems,* 1991.

[9] J. L. Peterson, *Petri Net Theory and the Modeling of Systems,* Englewood Cliffs, NJ: Prentice-Hall, 1981

[10] A. H. Levis, "Quantitative models of organizational information structures," in *Concise Encyclopedia of Information Processing in Systems and Organizations,* A. P. Sage, Ed., Oxford: Pergamon Books Ltd., 1988

[11] J.-M. Monguillet and A. H. Levis, "Modeling and evaluation of variable structure command and control organizations," *Proc. 1988 Symp. on C2 Research,* McLean, VA: SAIC, 1988.

[12] K. Jensen, *Coloured Petri Nets.* Book report. Denmark: Aarhus University, 1990.

[13] P. A. Remy and A. H. Levis, "On the generation of organizational architectures using Petri Nets," in *Advances in Petri Nets 1988,* G. Rozenberg, Ed., Berlin: Springer-Verlag, 1988.

[14] J. J. Demaël and A. H. Levis, "On generating variable structure architectures for distributed intelligence systems," *Proc. 11th IFAC World Congress,* Oxford: Pergamon Press, 1990.

[15] H. J. Einhorn and R. M. Hogarth, "Behavioral decision theory: Processes of judgment and choice," *Annual Review of Psychology,* vol. 32, 1981.

[16] S. T. Weingaertner and A. H. Levis, "Evaluation of decision aiding in submarine emergency decision making," *Automatica,* vol. 25, 1988

[17] D. M. Perdu and A. H. Levis, "Modeling and evaluation of expert systems in decision making organizations," *Automatica,* vol. 27, 1991.

[18] S. Kahne, "Control migration: a characteristic of C^3 systems," *IEEE Control Systems Magazine,* vol. 3, pp. 15-19, 1983.

[19] A. H. Levis and S. L. Skulsky, "Migration of control in decision making organizations," *Information and Decision Technologies,* vol. 15, 1990.

[20] J. L. Grevet, L. Jandura, J. Brode, and A. H. Levis, "Execution strategies for Petri Net simulations," *Proc. 12th IMACS World Congress on Scientific Computation,* 1988.