Distributed Diagnosis of Networked, Embedded Systems

James Kurien **Xenofon Koutsoukos** **Feng Zhao**

Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 93404
jkurien,koutsouk,zhao@parc.com
Phone: 1-650-812-4340
Fax: 1-650-812-4334

Abstract

Networked embedded systems are composed of a large number of physically distributed nodes that interact with the physical world via a set of sensors and actuators, have their own computational capabilities, and communicate with each other via a wired or wireless network. Monitoring and diagnosis for such systems must address several challenges caused by the distribution of resources, communication limitations, and node and link failures. This paper presents a distributed diagnosis framework that exploits the topology of a physical system to be diagnosed to limit inter-diagnoser communication and compute diagnoses in an anytime and any information manner, making it robust to communication and processor failures. The framework adopts the consistency-based diagnosis formalism and develops a distributed constraint satisfaction realization of the diagnosis algorithm. Each local diagnoser first computes locally consistent diagnoses, taking into account local sensing information only. The local diagnosis sets are reduced to globally consistent diagnoses through pairwise communications between local diagnosers. The algorithm has been successfully demonstrated for the diagnosis of paper path faults for the Xerox DC265 printer.

Introduction

Our diagnostic research is motivated by existing and emerging applications of networked, embedded systems. In such systems the physical plant is composed of a large number of distributed nodes, each of which performs a moderate amount of computation, collaborates with other nodes via a wired or wireless network, and is embedded in the physical world via a set of sensors and actuators. Examples include distributed sensor networks (Chu, Haussecker, & Zhao 2001), complex electromechanical systems with embedded controllers (Zhao *et al.* 2001), data networks, smart matter systems (Jackson *et al.* 2001), and ad-hoc wireless networks of consumer devices. Such systems present a number of interesting new challenges for diagnostic systems. A moderate amount of computation is potentially available, but it is partitioned into embedded chunks that range in size from tiny, in the case of smart dust sensor motes (Kahn, Katz, & Pister 1999) to moderate in the case of consumer devices. Communication between nodes is available, but may involve unreliable delivery, power-constrained wireless networks, or large, complex topologies requiring multiple hops to connect two arbitrary nodes. Finally, nodes might leave the network dynamically and nodes of a previously unseen type might join in their place.

In this paper, we consider how we might apply techniques from model-based diagnosis to these types of problems. In general, traditional model-based techniques are centralized. They assume that the diagnostic algorithm is run on a single processing unit that has access to observations from all sensors in the physical plant. In the next two sections of the paper, we briefly discuss centralized, model-based techniques and discuss how they cause scalability, robustness and reconfigurability problems if employed directly on networked, embedded systems. We then present a set of useful properties for diagnostic algorithms for such systems. In the fourth section, we present a simple formulation for diagnosis of discrete, distributed systems in order to motivate discussion and map the formulation onto distributed constraint satisfaction and distributed constraint optimization. We next propose an algorithmic framework for distributed diagnosis that operates in an anytime manner and is robust to communication and processor failures. We discuss the communications requirements for the framework and compare performance results for one instantiation of the distributed diagnosis framework against a centralized diagnoser. In the related work section, we discuss why existing distributed constraint satisfaction and optimization algorithms are not well suited for distributed diagnosis of networked, embedded systems. We finally discuss two open areas for future work. The contributions of this paper are that it illustrates the interesting features of networked, embedded systems that make them challenging for traditional model-based diagnosis techniques, it presents a simple formulation of the distributed diagnosis problem for these type of systems and relates it to distributed constraint satisfaction and optimization, it presents a class of robust, anytime algorithms for performing diagnosis, and it illustrates preliminary diagnostic results on a model of a real physical system with comparisons to an existing centralized diagnoser.

Model-based Diagnosis

The objective of diagnosis is to determine the state of a physical plant such as a printer, aircraft or network, based upon the current sensor readings from the plant and prior knowledge about the plant's structure and behavior. In order for the diagnosis to be useful for on-line control of the plant,

accurate diagnoses must be generated in a time-critical manner using the available computational resources. In most model-based diagnostic techniques, prior knowledge about the physical plant consists of a description of the behavior of each component of the plant, including normal and faulty behaviors, and the interconnections between components (Hamscher, Console, & de Kleer 1992). Partial observability presents the main challenge of diagnosis. Faults in a component may not be directly observable, and instead may cause changes in the behavior of the plant that propagate through several components before becoming observable at a sensor. To perform diagnosis, the component models are combined into a global store, observations are obtained from the physical plant, and a centralized algorithm is applied to find a system-wide diagnosis. We believe this very abstract description captures many diagnostic formalisms, including logic-based formalisms such as those based upon (de Kleer & Williams 1989) or (Reiter 1987), bond graphs (Mosterman & Biswas 1997) and many others. Throughout this paper we will use a formalism and examples consistent with GDE (de Kleer & Williams 1987) and its descendants, keeping in mind the general properties of centralized, model-based diagnosis that are at issue.

Figure 1 on the next page schematically illustrates a small model for the kind of traditional problem we might attack with a model-based diagnoser. The 24 boxes represent rollers, gears, motors, sensors and other devices in a printer paper path. For example, the acRoll acquires a sheet of paper from the paper tray and transports it to the feedRoll, driven by the acBelt. We have developed a simple diagnostic application for this paper path system using L2 (Kurien & Nayak 2000), a centralized, GDE-style diagnoser developed by NASA. Each component is modeled by finite state machine augmented with finite domain variables that describe its behavior. Arcs between components in Figure 1 represent interactions between components, for example conveying that the acRoll receives an angular velocity from the acBelt. This is represented by a constraint between the corresponding variables. There are five sensors that report the time of arrival of a sheet of paper at various points in the paper path.

To perform diagnosis with L2 and this model, observations as to when or if the paper arrived at various points in the path would first be obtained from the printer's sensors via its internal data bus and sent to an external processor running L2. The values would be discretized and assigned to the corresponding variables in the constraint system. A constraint optimization algorithm would be applied to the updated constraint system to find assignments to the variables that are consistent with the observations. Such an assignment might represent that the paper was late at the first sensor because the feedMotor is slow, slowing down both the acRoll and the feedRoll. This information could then be used to perform maintenance, or in systems with redundancy, to reconfigure the system for robust control. In addition to this small demonstration, we have applied similar diagnostic techniques to spacecraft (Bernard *et al.* 1998), chemical processing plants (Goodrich & Kurien 2001), scientific instruments, and other electromechanical systems to assist in robust control.

Given a set of component models and a centralized diagnoser C:

1. C combines the component models in a central store
2. Observations are collected from the physical system
3. C computes the system-wide diagnoses

Figure 2: Centralized Diagnosis of a Centralized System

Given a set S of currently connected components and a centralized diagnoser C:

1. $\forall S$, S forwards its component model to C
2. C combines the component models in a central store
3. $\forall S$, S forwards its observations to C
4. C computes the system-wide diagnoses
5. $\forall S$, C projects the variables of interest to S from the diagnoses and forwards them to S

Figure 3: Centralized Diagnosis of a Networked System

Challenges of Monitoring and Diagnosing Networked, Embedded Systems

Suppose we would like to perform diagnosis for a reconfigurable, networked, embedded system. Such systems are constructed such that each component is locally controlled by a small, embedded processor which coordinates with other processors via a potentially unreliable network. In addition, components and their processors might be unplugged and replaced with upgraded versions from time to time. Examples of such systems are ad-hoc wireless networks, modular robots, and more conventional systems such as intranets. Even traditional electro-mechanical systems such as printers and automobiles now contain on-board networks, embedded sensing and tens or hundreds of local controllers.

We can provide diagnostic information to the local controllers of such a system using centralized diagnosis via the process outlined in Figure 3. First, a centralized, global diagnosis problem is created by assembling a global model of the components within a centralized diagnoser. The observations are centrally collected and a diagnosis or set of diagnoses are computed by the centralized diagnoser. Aspects of the centralized, global diagnosis are then be distributed back to the local controllers.

This approach makes several assumptions. First, there must exist a processor large enough to store the global diagnostic model and run the centralized diagnostic algorithm. If this processor fails, it must be acceptable for no further diagnoses to be generated. Second, there must exist a central bus or buses with sufficient capacity to forward all data needed for diagnosis to the central processor. If a bus fails, the data needed to diagnose and recover for the failure must be located on the near side of the bus with respect to the diagnostic processor, or it must be acceptable for no further diagnoses to be generated for the bus and the far side compo-

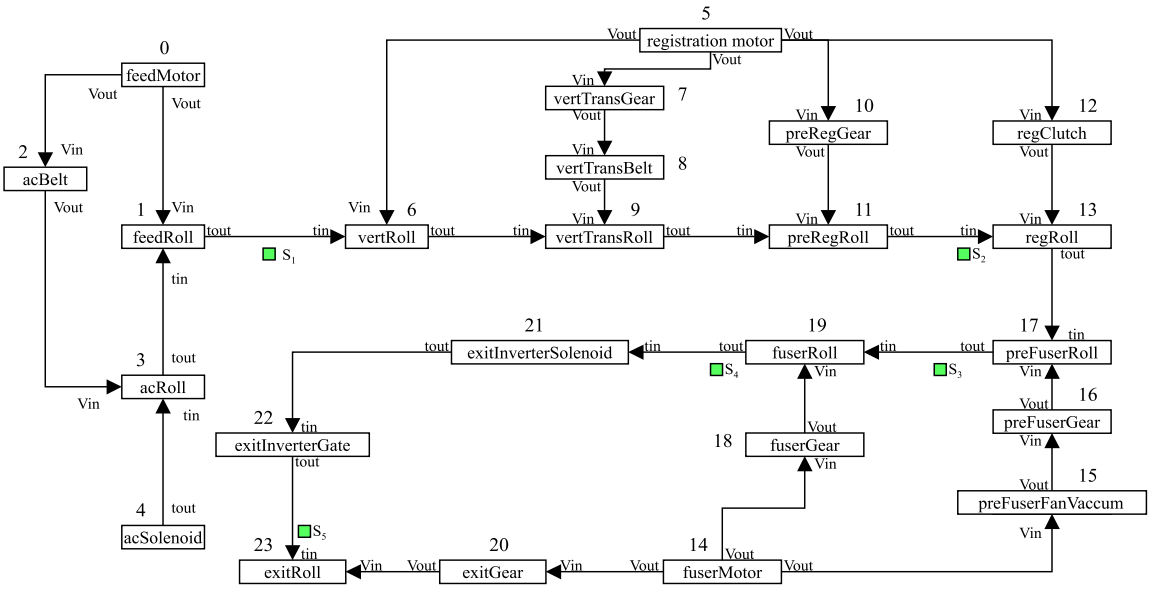

Figure 1: Paper Path Model in Xerox DC265ST Printer

nents. Finally, the set of components to be diagnosed must be represented using the same formalism, and in most applications must be known a priori.

With networked, embedded systems, all of these assumptions may be false. Each processor in the plant may be quite small. If a processor fails, we may require the components attached to remaining processors to continue operating in a full diagnosis and control cycle. If the network is bifurcated, we may require that each half of the plant continues operations to the extent possible and works to resolve the failure with the locally available information. New components might join into the network at any time by publishing their capabilities such as described by JINI (Sun Microsystems Inc 1999).

These issues suggest an approach wherein we do not artificially centralize the problem but allow a local diagnoser to be associated with each system processor. Each local diagnoser finds a partial diagnostic solution using a model of the locally controlled portion of the plant and the locally available observations. Communication is then required to refine the partial diagnostic solution into a diagnosis, in effect making use of observations and models local to other diagnosers. We next suggest themes for dividing and coordinating the diagnostic process to maximize scalability, robustness and reconfigurability, based upon our experience with both diagnosis and networked, embedded systems.

- **Scalability**

  Dividing the diagnostic problem among local diagnosers allows us to apply multiple processors and potentially address computational scalability problems caused by the small processors we may encounter in some embedded systems. To address communication scalability issues, we seek to exploit the topology of the physical plant. We would like to arrange that two local diagnosers need communicate only if the subsystems of the physical plant they correspond to are physically interconnected or share data. Thus the structure of our diagnostic architecture will mimic the physical topology of the plant being diagnosed. For the type of engineered systems that are typically amenable to diagnosis, physical scalability is accomplished by modularizing subsystems and connecting them through fairly narrow physical interfaces (power, data, physical support). By respecting these interfaces, we expect our communication needs for moving diagnostic data to scale as well as the underlying physical plant.

- **Robustness**

  A diagnostic architecture must be extremely robust to failure and able to operate in an anytime and any information manner. This can be accomplished with refinement. We would like to arrange that each diagnoser locally produce a superset of the diagnoses that a global diagnoser would produce for the local components. Communication with other diagnosers is then used only to prune the local diagnosis set. This yields several important properties. First, the diagnostic process can be interrupted at any time and each diagnoser will contain the true diagnosis plus possible imposters. This is an important safety feature in domains where taking action based upon a false negative can cause serious harm. Second, if diagnosers fail, then the remaining diagnosers will simply produce coarser (more conservative) estimates of the possible states of their components. Third, if the system is bifurcated due to a communication failure, then each half will produce all diagnoses consistent with the reachable diagnosers and any state of the other half of the system.

- **Reconfigurability**

  A side effect of employing local diagnosers that commu-

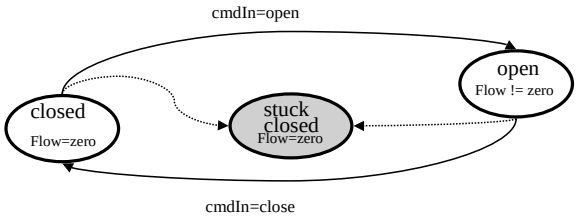

Figure 4: Automaton Representing A Single Valve

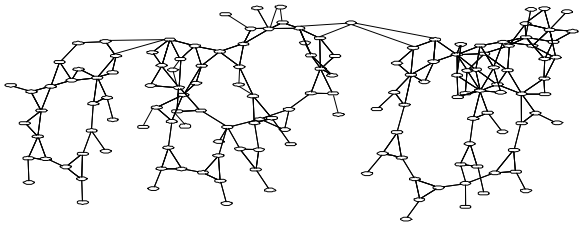

Figure 5: Variable Connectivity In a Global Model

nicate via opaque interfaces defined by the physical plant is natural support for modular or reconfigurable plants. Intuitively, a connected subset of the components of Figure 1 may be disconnected from the plant and replaced by new hardware with a different model, so long as the physical and diagnostic interface at the point of disconnection is maintained. In addition, this opens the possibility of participation by different implementations of the same diagnostic algorithm or even different algorithms participating in a diagnosis. The latter would of course require an interface that is semantically meaningful for all participating diagnosers. However, even the former capability might be useful in allowing vendors of components that are likely to be connected (e.g. data network components or power distribution components) to create diagnosers that can collaborate.

We believe these properties will be of interest as we begin to investigate applications involving very large numbers of embedded processors communicating via networks. In the next section we introduce a simple formalization that will allow us to discuss algorithmic directions for type of problem.

Centralized Formulation

Our approach to distributed qualitative diagnosis follows the centralized diagnostic formalism developed in (de Kleer & Williams 1989) and extended in (Williams & Nayak 1996) and (Kurien & Nayak 2000). To motivate our distributed algorithms, we begin with a brief overview of the centralized technique, summarized from (Kurien & Nayak 2000). Suppose we would like to diagnose the state of a single component, a valve, which is qualitatively modeled via the finite state machine illustrated in Figure 4. We refer to each possible discrete state of a component as a $mode$. A valve v has three modes, $open$, $closed$, and $stuckClosed$. The behavior of the flow of the valve within each mode, which has the discrete domain $\{zero, nonzero\}$, can be captured with the following propositional formulae.

$$\begin{array}{lcl} v = open & \Rightarrow & flow_v = nonzero \\ v = closed & \Rightarrow & flow_v = zero \\ v = stuckClosed & \Rightarrow & flow_v = zero \end{array}$$

If $flow_v$ is observable from the physical plant, we will refer to this variable as an *observation*. In order to represent the non-determinism of the automaton within a propositional framework, the encoding introduces an *assumption* variable a. Intuitively, a_v represents the choice that Nature makes as to whether valve v will behave normally or experience a failure when it is commanded. The transition portion of the automaton can thus be captured by the following formulae.

$$\begin{array}{l} a_v = normal \Rightarrow \\ \quad \begin{array}{lcl} v_t = closed \wedge cmd_t = open & \Rightarrow & v_{t+1} = open \\ v_t = closed \wedge cmd_t \neq open & \Rightarrow & v_{t+1} = closed \\ v_t = open \wedge cmd_t = close & \Rightarrow & v_{t+1} = closed \\ v_t = open \wedge cmd_t \neq close & \Rightarrow & v_{t+1} = open \\ v_t = stuckClosed & \Rightarrow & v_{t+1} = stuckClosed \end{array} \\ a_v = stick \Rightarrow v_{,t+1} = stuckClosed \end{array}$$

Intuitively, the diagnostic task is to find a set of assignments to the assumptions, here $\{a_v\}$, such that the model is consistent with the observations, here $\{flow_v\}$. For example, suppose $v_t = closed$, we command the valve open, represented by $cmd_t = open$. The plant assigns O as $flow_v = zero$. The only consistent assignment to a_v is $a_v = stick$ and we diagnose valve is stuck closed. If we wish to model multiple automata, we introduce a mode and assumption for each automaton and compile all automata into a set of formulae that may share variables. For example, two valves in series share the same flow. Figure 5 visualizes the compilation of the device constraints into a global constraint system model. Each node represents a finite domain variable. Two nodes are connected by an edge if the two variables appear in a constraint together, denoting that the possible values of the variables are related by interacting together in some physical process or the transmission of data. Note that a realistic model such as that of Figure 5 contains many observations and assumptions, and many assignments may be consistent. More formally, let A denote the set of assumptions, O denote the set of observations, and F denote the formulae describing the plant. Given an assignment Ω to O created by observing the plant, a diagnosis D is an assignment to A such that the following propositional formula is consistent:

$$\wedge_{a_i \in A}(a_i = d_i) \ \wedge_{o_j \in O} (o_j = \omega_j) \wedge F.$$

To perform diagnosis over multiple components, we must find an assignment to each a that renders the set of formulae consistent with all observations. Intuitively, we assign the observations reported by the physical plant, Ω to the variables of the graph corresponding to observations, O, then reassign the assumption variables, A until the constraint system illustrated in Figure 5 becomes consistent. Thus in this diagnosis framework, diagnosis can be viewed a constraint satisfaction problem.

A second diagnostic task is to find the most likely diagnoses. For each assumption assignment we can associate the prior probability of the even the assumption represents.

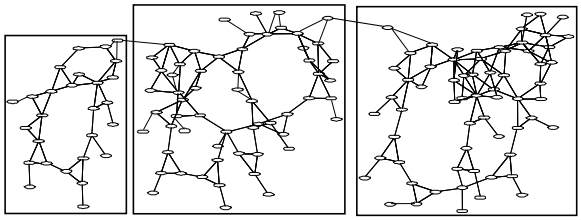

Figure 6: Partition Among Three Diagnosers

Thus, P(a_v=stick) denotes the prior probability of the valve sticking. Assuming conditional independence, the probability of a diagnosis is defined as follows.

$$P(D) = \Pi_{a_i \in D} P(a_i = d_i)$$

Given multiple components, we must find the assignment to each a that renders the set of formulae consistent with all observations such that the probability of the assignment is maximal. Intuitively, we assign the observations reported by the physical plant, Ω to the variables of the graph corresponding to observations, O, then choose among the possible reassignments of assumption values to assumption variables, A, until the constraint system illustrated in Figure 5 becomes consistent. The choice of which assumption to reassign and to which value to assign it is based upon the probability of the possible assignments. In this case, diagnosis can be viewed as a constraint optimization problem.

Distributed Diagnosis

In this paper, we propose splitting the global diagnostic process into a number of cooperating local diagnostic processes. In order to distribute the problem, we divide the global diagnoser which produces assignments to A into a set of local diagnosers which make assignments to subset of A. Intuitively, we partition the edges of Figure 5. If a node is connected to edges in more than one partition, it is replicated and the partitions must reach consensus on its value. More formally, a local diagnoser L is described by (F_L, V_L, A_L, O_L, R_L) where F_L is the subset of F assigned to L, V_L denotes the set of variables that appear in F_L, A_L denotes $A \cap V_L$, O_L denotes $O \cap V_L$ and R_L denotes the union of $V_L \cap V_M$ over all other diagnosers M. Figure 6 illustrates a possible partitioning of the constraint graph of Figure 5. The slightly darker nodes indicate the members of R_L, shared variables that have been replicated. Given a fixed number of diagnosers or the maximum number of constraints a diagnostic processor can accommodate, we can use a graph partitioning algorithm (Sanchis 1989) to find a partitioning of the graph that attempts to minimize R_L for each diagnoser.

Our approach to finding consistent diagnoses in a distributed fashion is refinement based. Intuitively, each local diagnoser finds the diagnoses for the locally modeled component that are consistent with the constraints of the local model and the local observations. This is a superset of the diagnoses for the local components that are consistent will all constraints and observations. Each local diagnoser then communicates with directly other diagnosers to further reduce the set of consistent diagnoses for the local components. We would like that the diagnoses start with a superset of the globally consistent diagnoses and move toward only the globally consistent diagnoses. We define the relationships $conservative$ and $feasible$ between the diagnoses produced by a global diagnoser and the diagnoses produced by a local diagnoser. A local diagnosis set D_L is conservative with respect to the global diagnosis set D_G if

$$\forall \delta_G \in D_G \; \Pi_{A_L}(\delta_G) \in D_L$$

where Π is the projection operator. That is, the assignments made to the assumptions local to L by a global diagnosis must also be made by a local diagnosis. A local diagnosis set D_L is feasible if the assignments made to the local assumptions are contained in a consistent global diagnosis. More formally,

$$\forall \delta_L \in D_L \; \exists \delta_G \in D_G : \Pi_{A_L}(\delta_G) = \delta_L.$$

1. Given observation set Ω, if $o_j \in O_L$, assign $o_j = \omega_j$ in L.
2. $\forall L$, if $O_L \neq \emptyset$, compute all assignments to $A_L \cup R_L$ s.t. $\wedge_{o_j \in O_L} (o_j = \omega_j) \wedge_{a_i \in A_L} (a_i = d_i) \wedge_{r_i \in R_L} (r_i = \rho_i) \models F_L$
3. For each $r \in R_L$, for each other diagnoser M, if $r \in V_M$ send all R_L assignments to M.
4. In each such M, compute all assignments such that $\wedge_{r_i \in R_L} (r_i = \rho_i) \wedge_{a_k \in A_M} (a_k = d_k) \wedge_{r_k \in R_M} (r_k = \rho_k) \models F_M$
5. If the consistent R_M assignments decreased in step 4, return to step 3, substituting M for L.

Figure 7: Consistency-based, Anytime Diagnosis

Incremental Consistency

We next discuss an algorithmic framework for incrementally revising a set of conservative diagnoses into a set feasible diagnoses in a robust, anytime, distributed manner, followed by results from one particular instantiation of this framework. The approach of the algorithmic framework is similar in spirit to Waltz's algorithm (Waltz 1975). Each set of diagnoses is monotonically reduced toward a feasible set as a side effect of spreading consensus on the value of variables shared between diagnosers. The algorithm is illustrated in Figure 7.

The algorithm operates by incrementally reducing the possible assignments to A_L for all L, first by introduction of observations and second by communication between diagnosers. Each local diagnoser begins with a conservative local diagnosis set in A_L. Typically this would be all possible diagnoses, which can be implicitly captured by an appropriate encoding of the constraint set F_L. In Step 1, observations are assigned in every diagnoser which has constraints involving an observation. In Step 2, the observation assignments are used to compute all assignments to $A_L \cup R_L$ that are consistent with F_L and the observations received by L. Note that the projection of A_L from these assignments is a

conservative diagnosis set. Intuitively, suppose an assignment to A_L appears in a global diagnosis but is not computed by L. If it is not computed, it must be inconsistent with F_L and the assignments to O_L. It is therefore inconsistent with F and the assignments to O, and could not appear in a global diagnosis. In Step 3, the assignments to R_L are projected out of the consistent assignments of L and forwarded to each other diagnoser M that references these variables. In Step 4, M eliminates a subset of its assignments that are not feasible. Intuitively, an assignment α to A_M is not feasible if there is no assignment to A containing α that is consistent with F and O. If α constrains a variable in R_L to have a value that was not received from L, then α is inconsistent with all consistent assignments to A_L. Thus, each time Step 4 is performed, infeasible assignments to A_M are eliminated. Each diagnoser begins with a conservative set of assignments to A_L, and as rounds of communication are performed, the local diagnoses are moved toward feasibility in an anytime manner. Per Step 5, the algorithm continues as long as consistent assignments are eliminated. In the worst case, each loop would eliminate one of an exponential number of possible assignments.

Note that we have described the algorithm to propagate sets of assignments that remain consistent in one local diagnoser to to other diagnosers in which the assigned variables appear. More generally, we may propagate any information that allows remote diagnosers to restrict the domain of a variable based upon inference performed in the local diagnoser. Examples include assignments that cannot be made because of constraints within one diagnoser (nogoods), assignments that must be made, or sets of possible assignments to a variable that remain consistent. Note also that this algorithm is not complete with respect to distributed constraint satisfaction. Intuitively, suppose we have two local diagnosers, one containing only the constraint $A \vee B$ and the other containing only the constraint $\bar{A} \vee B$. Neither can constrain and propagate the value of B, though B must be true. This same restriction applies to the centralized constraint satisfaction technique used in L2, so we do not believe it presents a significant drawback. The related work section contains further details on the relationship between distributed diagnosis and distributed constraint satisfaction and why we believe an incomplete algorithm is sufficient.

Communication Requirements

When presented with a networked, embedded system, we may perform centralized diagnosis of the distributed system by transmission of observations or distributed diagnosis of the distributed system by transmission of intermediate results. Choosing distributed diagnosis allows us to trade communication bandwidth for reduced processor requirements, increased robustness and greater reconfigurability. In this section, we examine how the communication requirements of the distributed, incremental diagnosis algorithm compare to a centralized approach. We first consider the communication requirements of the centralized procedure shown in Figure 3. Let n be the number of components and s be the number of components with sensors. In Step 3 of the procedure, each of s components forwards its observations to C. In Step 5, C forwards the diagnostic results to each of n components. Assuming all observations from a single component can be sent in a single message, Figure 3 requires s point to point messages to C and one broadcast message from C to all n components

We now consider the communication requirements for the distributed algorithm of Figure 7. This algorithm performs distributed diagnosis by exchanging messages that refine the value of shared variables across local diagnosers. Let v be the number of variables that are shared, and r be the average number of diagnosers that share each variable, and m be the average number of messages exchanged that involve a given variable. For example, if each local diagnoser uses unit propagation, it can send messages specifying that a variable must have a certain value or cannot have a certain value, but no messages specifying disjunctions between assignments. Thus m is bounded by the size of the largest domain of a shared variable. The increase in messages created by moving to the distributed diagnoses technique is given by the ratio

$$\alpha_1 = \frac{vrm}{s+1}.$$

Note that counting the number of messages exchanged is not sufficient to determine the cost of communication. In many applications, such as wireless networks with limited energy or bandwith, the number of packets transmitted is a critical cost measure. Network topology will determine the number of packet transmissions or hops required to deliver a message. In many applications, each node in a network is connected to a small number of neighbors. Point to point communication is implemented by multiple hops between neighbors, and a broadcast is implemented by flooding the network. Let h_c be the average distance in hops between a node with a sensor and the centralized diagnoser. Let h_v be the average number of hops between nodes that share a variable. In general, the change in the total number of packet transmissions required by decentralizing the problem is determined by

$$\alpha_2 = \frac{vrmh_v}{sh_c + n}$$

Intuitively, packet transmission for the centralized diagnoser scales with the size and width of the network, while the decentralized approach scales with the number of constraints that cross network components. Note that if the network topology reflects the physical interactions of the components, it is likely the case that $h_v < h_c$. Thus we can construct wide networks with very localized interactions for which centralized diagnosis requires more packet transmissions than decentralized diagnosis, though we do not expect this to be the case in practice. In addition to total packet transmission, we may further refine our cost measure to include the maximum number of packets transmitted by any link in the network. This determines the minimum bandwidth or power storage a network node must support. The ratio α_2 does not capture that in the centralized case, all messages must pass through network links connected to the central diagnoser. This drives up the minimum capabilities of a network node in relation to distributed diagnosis where message sources and destinations are more evenly distributed

Independent Faults In	L2		Distributed		
	Diag	Time	Spread	Diag	Time
First module	6	0.02	9	21	0
Two modules	12	0.18	14	49	0
Three modules	84	13.28	20	343	0.05
All modules	108	27.08	24	637	0.22

Table 1: Comparison of distributed diagnoser and L2

through the system. We are currently defining a diagnostic model for a distributed sensor network in addition to available models of more traditional electro-mechanical systems in order to better characterize the communication requirements of both distributed and centralized algorithms

Results

To implement the distributed diagnosis algorithm described above, each local diagnoser could represent its conservative diagnosis set as a partial assignment in a GDE-style diagnoser, a relational table, a binary decision diagram and so on, so long as the representation can be efficiently pruned when an observation or neighboring diagnoser decreases the range of a variable. Ideally, we would like to test a centralized diagnoser against a set of local diagnosers that compute and represent diagnoses in the same manner. For these preliminary results, we present the performance of the centralized L2 diagnoser against a distributed diagnoser that takes advantage of the small local model size enabled by distributing the problem. PARC intern Rong Su implemented the distributed algorithm using finite-state automata to prune inconsistent assignments to V_L (Steps 2 and 4 of Figure 7) and a distributed consensus algorithm (Steps 3 and 5) shown to converge to feasible diagnoses (Su *et al.* 2002). Table 1 compares performance with L2 on the paper path model. The first three columns are the name of the diagnostic scenario, the diagnoses found by L2, and the time required. Since the physical plant has few sensors, the number of consistent diagnoses grows with the complexity of the scenario. The fourth column is the number of local diagnosers reached via Step 3 of the algorithm, out of 24. The fifth column lists the number of diagnoses found by the distributed algorithm. Note that the FSA-based algorithm finds more diagnoses than L2. L2 is conflict based, and thus postulates only those failures that can eliminate a discrepancy between an expected observation and the observation received from the plant. The FSA-based algorithm finds all consistent failures, including those that would be indistinguishable from proper operation of the plant. The sixth column is the time to compute the diagnoses, demonstrating the dramatic speed advantage, on this model, of computing feasible local diagnoses via a pre-compiled FSA representation then determining consistent combinations versus global, on-line inference. The current implementation runs each local diagnoser serially on a single processor, and we believe a parallel implementation will provide a greater speed advantage.

Related Work

A diagnoser for a networked, embedded system may be centralized, decentralized or distributed. Work in centralized diagnosis may be applied by collecting models and observations from the networked components of the physical plant and appling a centralized algorithm. As described in the third section of this paper, this raises robustness and scalability issues that must be addressed. Rish, Brodie and Ma, for example, attempt to increase the efficiency of a centralized diagnostic procedure for a distributed network of computers using an approximate representation and carefully designed active probing of the distributed system (Rish, Brodie, & Ma 2002). In decentralized diagnosis, e.g. (Debouk, Lafortune, & Teneketzis 2000), local diagnosers communicate with a coordination process that assembles a global diagnosis. The coordination process of decentralized approaches are still subject to robustness and scalability issues. We are therefore pursuing an approach of distributed diagnosis, similar to (Baroni *et al.* 1999), where there is no centralized control structure or coordination process. Each local diagnoser communicates directly with other diagnosers.

We have formulated the the distributed diagnostic process as a distributed constraint satisfaction problem (DCSP). Since many problems in scheduling, resource allocation, and hardware design can be formulated as constraint satisfaction problems, the distributed constraint satisfaction problem has received a large amount of attention. Yokoo and Hirayama provide an excellent overview (Yokoo & Hirayama 2000) of algorithms for solving DCSP's. These existing algorithms do not meet our needs for two reasons. First, the great majority of the algorithms are formulated assuming the computational nodes and network connecting the nodes are reliable, and that all messages sent between nodes arrive in the order sent. For diagnosis of networked, embedded systems, we seek specific guarantees of behavior in response to the loss of computing nodes or bifurcation of the network. Second, the majority of DCSP algorithms are designed to solve general discrete constraint satisfaction problems, such as the graph coloring problem. The ability to solve general CSP problems requires features that complicate distribution, such as backtracking on choices for variable assignments. In practice, centralized diagnosers are able to find consistent diagnoses using incomplete, backtrack-free procedures such as unit propagation. This difference arises because the constraints we generate from finite state models such as illustrated in Figure 4 tend to be closer to Horn clauses in structure than general discrete constraints and diagnosis may use observation values asserted by the physical plant to drive constraint processing. We therefore expect a distributed diagnoser acting upon the same models should be able to use less powerful inference methods than full constraint satisfaction. While we have encountered full DCSP algorithms that allow some fault tolerance, such as the Mozart system (Roy 1999), and some simpler constraint processing methods that assume reliable, fully connected networks, such as distributed arc consistency (Nguyen & Deville 1998), we have not yet encountered an algorithm that is sufficiently narrow in scope and robust to failures.

Future Work

A number of issues remain for future work. The issue of how to use knowledge of the prior probability of failures to avoid computing all consistent diagnoses has been explored but not solved. The algorithm of Figure 7 also does not take into account any information about the likelihood of failures. We may of course find the set of globally consistent diagnoses and compute the probability of each by assuming conditional independence of the failures, as described above. However, rather than computing the probabilities of all consistent diagnoses, we might wish to avoid generating unlikely diagnoses given we have generated a sufficient number of consistent, likely diagnoses. Conflict-directed, best-first search (de Kleer & Williams 1989) is a centralized, discrete constraint optimization algorithm that is specialized for diagnosis. It efficiently enumerates consistent assignments to a set of propositional variables in order of their cost, or in this case enumerates diagnoses in order of their prior probability. Intuitively, it operates by starting with the highest probability assignment to the assumptions, the case where no failures have occurred. It substitutes a minimal cost assignment to an assumption with a non-minimal cost assignment only when a conflict between an observation value assigned by the plant and the value predicted by the current assumption assignments occurs. Our current direction in developing a distributed analog is to begins with a maximum likelihood (*e.g.*,no failure) assignment to A_L within each diagnoser L, which in turn constrains the shared variables. When diagnosers L and M disagree on the value of a shared variable r, each performs a local diagnosis to conservatively approximate the maximum probability assignment to the assumptions that would admit a different value for r. This information can then be used to limit propagation of variable changes throughout the system. We have implemented a preliminary version of this system using copies of L2 as the local diagnosers for the purposes of exploration, but we are currently limited to very simple network topologies. Formalizing a reasonably general algorithm for generating a conservative estimate of the most likely diagnoses in a robust, distributed, anytime manner remains future work.

As framed here, the distributed diagnoser never computes complete global diagnoses. Rather, at each local diagnoser it computes feasible local diagnoses. These are projections of the global diagnoses that are relevant to that diagnoser. In the case that control of the plant is distributed, we believe this is appropriate. Each processing node uses the possible states of its components, as determined by the feasible local diagnoses, to inform its control. However, even when performing distributed diagnosis of a distributed system, computation of the global diagnoses may be of interest for purposes such as centralized, supervisory control or display to a user. We note that simply taking the cross-product of the feasible diagnoses produced by each local diagnoser will result in a superset of the global diagnoses. Some combinations of the cross-product may not appear in any consistent global diagnosis. If the consistent global diagnoses are needed, we may compute them by checking combinations of local feasible diagnoses from multiple diagnosers against a combined model using a linear-time technique such as unit propagation. This can be done hierarchically and in parallel, allowing us to rule out inconsistent partial combinations of local diagnoses in order to avoid explicitly checking all combinations. Intuitively and from initial experiments, we suspect for many problems this technique would be a competitive method for producing all consistent global diagnoses. In fact, the performance numbers for the FSA-based distributed algorithm shown in Table 1 are for both computing the conservative and feasible local diagnoses for each local diagnoser and then computing the globally consistent combinations of these local diagnoses. Formalizing this technique and more thoroughly investigating its effectiveness remain future work.

Conclusion

We have developed a distributed diagnosis framework that leverages the topology of the physical plant to limit inter-diagnoser communication and compute consistent diagnoses in an anytime and any information manner, making it robust to communication and processor failures. The framework is conservative, in that it avoids false negatives in favor of false positives in the case where computation cannot be completed due to limited time or communication failure. This property can be vital in applications where safety is critical. In addition to being anytime and conservative, our approach allows a very small granularity for the local diagnosers. We can potentially create a diagnoser per physical component if desired. This flexibility allows us to consider time/space/communication tradeoffs that implement each local diagnoser as an exponentially large (in the small local model size) structure that enables diagnosis to be performed collaboratively on very weak networked processors. One implementation of the distributed algorithm for finding consistent diagnoses has been implemented using a discrete-event formulation and tested on one model. Our future work includes implementations of the algorithm using binary decision diagrams and the unit propagation implementation of L2 to compute locally consistent assignments. The latter will allow direct comparison of centralized and distributed implementations of the same diagnostic technique on a variety of problems modeled for L2. We are also continuing to extend the formulation to include optimization-based distributed diagnosis.

Acknowledgment This work is supported in part by the Defense Advanced Research Projects Agency (DARPA) under contract F33615-99-C3611. Rong Su implemented the distributed diagnoser as a PARC intern. NASA Ames Research Center provided the L2 diagnosis engine.

References

Baroni, P.; Lamperti, G.; Pogliano, P.; and Zanella, M. 1999. Diagnosis of large active systems. *Artificial Intelligence* 110(1):135–183.

Bernard, D. E.; Dorais, G. A.; Fry, C.; Jr., E. B. G.; Kanefsky, B.; Kurien, J.; Millar, W.; Muscettola, N.; Nayak, P. P.; Pell, B.; Rajan, K.; Rouquette, N.; Smith, B.; and Williams, B. C. 1998. Design of the remote agent experiment for spacecraft autonomy. In *Procs. IEEE Aerospace*.

Chu, M.; Haussecker, H.; and Zhao, F. 2001. Scalable information-driven sensor querying and routing for ad hoc heterogeneous sensor networks. *Int'l J. High Performance Computing Applications*. To appear. Also, Xerox Palo Alto Research Center Technical Report P2001-10113, May 2001.

Collin, Z.; Dechter, R.; and Katz, S. 1999. Self-stabilizing distributed constraint satisfaction. *Chicago Journal of Theoretical Computer Science*.

de Kleer, J., and Williams, B. C. 1987. Diagnosing multiple faults. *Artificial Intelligence* 32(1):97–130. Reprinted in (Hamscher, Console, & de Kleer 1992).

de Kleer, J., and Williams, B. C. 1989. Diagnosis with behavioral modes. In *Proceedings of IJCAI-89*, 1324–1330. Reprinted in (Hamscher, Console, & de Kleer 1992).

Debouk, R.; Lafortune, S.; and Teneketzis, D. 2000. Coordinated decentralized protocols for failure diagnosis of discrete event systems. *Discrete Event Dynamic System: Theory and Applications* 10(1/2):33–86.

Goodrich, C., and Kurien, J. 2001. Continous measurements and quantitative constraints - challenge problems for discrete modeling techniques. In *Proceedings of iSAIRAS-2001*.

Hamscher, W.; Console, L.; and de Kleer, J. 1992. *Readings in Model-Based Diagnosis*. San Mateo, CA: Morgan Kaufmann.

Jackson, W.; Fromherz, M.; Biegelsen, D.; Reich, J.; and Goldberg, D. 2001. Constrained optimization based control of real time large scale systems: Airject movement object system. In *Proceedings of the 40th IEEE Conference on Decision and Control*, 4717–4720.

Kahn, J. M.; Katz, R. H.; and Pister, K. S. J. 1999. Mobile networking for smart dust. In *ACM/IEEE Intl. Conf. on Mobile Computing and Networking (MobiCom 99)*.

Kurien, J., and Nayak, P. P. 2000. Back to the future with consistency based trajectory tracking. In *Proceedings of AAAI-00*.

Mosterman, P., and Biswas, G. 1997. Monitoring, prediction and fault isolation in dynamic physical systems. In *Proceedings of AAAI-97*, 100–105.

Nguyen, T., and Deville, Y. 1998. A distributed arc-consistency algorithm. *Science of Computer Programming* 30(1-2):227–250.

Reiter, R. 1987. A theory of diagnosis from first principles. *Artificial Intelligence* 32(1):57–96. Reprinted in (Hamscher, Console, & de Kleer 1992).

Rish, I.; Brodie, M.; and Ma, S. 2002. Efficient fault diagnosis using probing. In *Proceedings of the AAAI Spring Symposium on Information Refinement and Revision for Decision Making: Modeling for Diagnostics, Prognostics and Prediction*.

Roy, P. V. 1999. The separation of concerns in distributed programming: Application to distribution structure and fault tolerance in mozart.

Sanchis, L. A. 1989. Multiple-way network partitioning. *IEEE Transactions on Computers* 38(1):62–81.

Su, R.; Wonham, W. M.; Kurien, J.; and Koutsoukos, X. 2002. Distributed diagnosis for qualitative systems. Technical Report SPL-01-071, Palo Alto Research Center. Submitted to WODES 2002.

Sun Microsystems Inc. 1999. Jini architectural overview.

Waltz, D. L. 1975. Understanding line drawings of scenes with shadows. In Winston, P. H., ed., *The Psychology of Computer Vision*. McGraw-Hill. 19–91.

Williams, B. C., and Nayak, P. P. 1996. A model-based approach to reactive self-configuring systems. In *Procs. AAAI-96*, 971–978.

Yokoo, M., and Hirayama, K. 2000. Algorithms for distributed constraint satisfaction: A review. *Autonomous Agents and Multi-Agent Systems* 3(2):185–207.

Yokoo, M.; Durfee, E. H.; Ishida, T.; and Kuwabara, K. 1992. Distributed constraint satisfaction for formalizing distributed problem solving. In *International Conference on Distributed Computing Systems*, 614–621.

Yokoo, M.; Durfee, E. H.; Ishida, T.; and Kuwabara, K. 1998. The distributed constraint satisfaction problem: Formalization and algorithms. *Knowledge and Data Engineering* 10(5):673–685.

Zhang, Y., and Mackworth, A. K. 1992. Parallel and distributed finite constraint satisfaction: Complexity, algorithms and experiments. Technical Report TR-92-30, Department of Computer Science, The University of British Columbia.

Zhao, F.; Koutsoukos, X.; Haussecker, H.; Reich, J.; Cheung, P.; and Picardi, C. 2001. Distributed monitoring of hybrid systems: A model-directed approach. In *Proc. IJCAI'2001*, 557–564.