Diagnosis of Coordination Faults: A Matrix-Based Formulation

Meir Kalech[1] and **Michael Lindner**[2] and **Gal A. Kaminka**[2]
[1]School of Engineering and Applied Sciences, Harvard University, MA USA *kalech@eecs.harvard.edu*
[1]Computer Science Department, Bar Ilan University, Israel *lindnerm@gmail.com, galk@cs.biu.ac.il*

Abstract

One of the key requirements in many multi-agent teams is that agents must coordinate on specific aspects of their joint task. Unfortunately, this coordination may fail due to intermittent failures in sensor readings, communication failures, etc. A key challenge in model-based diagnosis of such coordination faults is to represent the model coordination between the agents in a way that allows efficient detection and diagnosis, based on observations of the agents involved. Previous mechanisms are useful only for small groups as they represent the coordination with binary constraints. This paper presents a model-based diagnosis (MBD) approach to coordination failure in which non-binary constraints are allowed. This model presents two advantages: (1) it appears quite frequently when modeling real problems, (2) it addresses large groups by gathering multiple coordinations in one constraint. To solve the diagnosis problem, we propose a matrix-based approach to represent the basic building blocks of the MBD formalization. This representation is both generic and efficient for large-scale teams.

1 Introduction

With increasing deployment of robotic and agent teams in complex, dynamic settings, there is an increasing need to also be able to respond to failures that occur in multi-agent teams [Tambe, 1997; Kaminka and Tambe, 2000; Kalech and Kaminka, 2006]. One type of failure of particular interest in multi-agent systems is a *coordination fault*, where agents come to disagree on salient aspects of their joint task.

There is thus a particular need to be able to detect and diagnose the causes for coordination faults that may occur, in order to facilitate recovery and reestablishment of collaboration, e.g., by negotiations [Kraus *et al.*, 1998]. This type of diagnosis is called *social diagnosis*, since it focuses on finding causes for failures to maintain social relationships, i.e., coordination failures.

In this paper we focus on a model-based diagnosis approach for coordination failures. Model-based diagnosis (*MBD*) [Reiter, 1987; de Kleer and Williams, 1987] relies on a model of the diagnosed system, which is utilized to simulate the behavior of the system given the operational context (typically, the system inputs). The resulting simulated behavior (typically, outputs) are compared to the actual behavior to detect discrepancies indicating failures. The model can then be used to pinpoint possible failing components within the system.

Previous work presents model-based diagnosis for coordination faults [Kalech and Kaminka, 2005; 2006], however, it models the coordination between every pair of agents as a set of binary constraints between the agents' states. Thus, this representation does not scale well in the group size and in the number of states.

On the contrary, non-binary constraints appear quite frequently when modeling real problems [Bacchus and van Beek, 1998]. Such problems could be naturally defined by non-binary constraints between multiple agents. In addition, there are domains like RoboCup Rescue [Tambe *et al.*, 2005] or ModSAF [Tambe, 1997], in which it may be more efficient to gather multiple coordinations (joint states) in one constraint rather than only one coordination per constraint. For instance, in RoboCup Soccer the players must coordinate the attack and the defense [Matsubara *et al.*, 1998]. It is naturally defining the coordination between multiple attackers and multiple defenders and goalie with non-binary constraints. In addition, by a single constraint we can define the coordination between part of the actions of a defender with partial set of the attacker's actions and the goalie's actions.

In this paper we propose a model-based approach to address this kind of coordination setting. We model the desired behavior of a team, i.e., the allowed coordination among the agents. At runtime the agents are observed and by inferring their states and comparing to the allowed coordination model, we diagnose the faulty agents. To solve the diagnosis problem, we propose to use a matrix-based representation [Kalech *et al.*, 2007] for the fundamental building blocks of the diagnosis problem. This representation has several benefits. First, it provides an easy and intuitive way to define the coordination between teammates. Second, since we do not represent the relations between teammates explicitly, but gather them compactly (joint coordination in the same matrix structure), this approach is scalable in the number of agents and states (unlike the approach proposed in [Kalech and Kaminka, 2005]). Finally, the use of a matrix-based representation, enables the use of the matrix operations and yields interesting information about the agents. To summa-

rize, the matrix representation enables an easy and efficient way to diagnose coordination failures.

2 Related Work

Kalech and Kaminka [Kalech and Kaminka, 2005] present a model-based diagnosis for general framework of coordination faults. In particular, they present consistency- and abductive-based approaches to this problem and propose distributed constraint satisfaction algorithms to solve the diagnosis problem [Kalech and Kaminka, 2006]. However, they model the coordination between the agents in pairs, meaning that their model grows exponentially in the group size and in the number of states.

Horling et al. [Horling *et al.*, 1999] uses a fault-model of failures and diagnoses to detect and respond to multi-agent failures. In this model a set of pre-defined diagnoses are stored in acyclic graph nodes. When a fault is detected a suitable node is triggered and according to the fault characters the node activates other nodes along the graph. However, this work does not address the scale-up issues. In addition, the failure-model approach dictates that all possible failures be analyzed in advance.

Fröhlich et al. [Fröhlich *et al.*, 1997] suggest dividing a spatially distributed system into regions, each under the responsibility of a diagnosing agent. If the fault depends on two regions the agents that are responsible for those regions cooperate in making the diagnosis. This method is inappropriate for dynamic team settings, where agents cannot pre-select their communication partners. Similarly, Roos et al. [Roos *et al.*, 2004] analyze a model-based diagnosis method for spatially distributed knowledge. But, their method assumes that there are no conflicts between the knowledge of the different agents, i.e., that no coordination failure occurs.

Williams et al. [Williams *et al.*, 2001; Kim *et al.*, 2001] provide a model for cooperation of unmanned vehicles. They coordinate these vehicles by introducing a reactive model-based programming language (RMPL). This model is robust and can detect failures and recover. However, their model-based language addresses only smaller-scale systems.

In previous work [Kalech *et al.*, 2007] we have proposed an approach to representing multi-agent coordination and observations, using matrix structures. This representation facilitates easy representation of coordination requirements, modularity, flexibility and reuse of existing systems. We have demonstrated how in principle, this representation can support detection of coordination faults. In this paper, we build on this work and utilize the matrix-based representation in model-based coordination diagnosis. We show that we can compactly represent joint states using matrix structures, and thus reduce (in part) the exponential complexity of the diagnosis to linear in the number of agents and states.

3 Fundamental Objects

We adopt a model-based diagnosis approach to diagnose the agents and the coordination failures. In model-based diagnosis of a single agent, the diagnoser uses a model of the agent to generate expectations which are compared to the observations, in order to form diagnoses [Reiter, 1987; de Kleer and Williams, 1987]. In model-based multi-agents diagnosis, the diagnoser models the coordination between the agents [Kalech and Kaminka, 2005]. The goal of the diagnosis is to diagnose the failures in the coordination by detecting deviation of the observation from the model's predictions.

3.1 The Agent Model

The most fundamental entity is an *agent*. At any moment, an agent is found in a given *state*. This is a logical, internal representation of the agent status, or belief, at this very moment. Throughout the paper, we will refer to the following sets:

(i) Let A be a set of n agents, $\{a_1, a_2, ..., a_n\}$.

(ii) Let S be set of m states, $\{s_1, s_2, ..., s_m\}$.

For example, consider a management system for a shop consisting of the following six agents (hereinafter this example will be referred as "the shop "): ANNY the manager, BENNY the cashier, two sellers (CANNY and DANNY), ERNY the storekeeper and a guard, FRENNY:

A_{shop}={ANNY, BENNY, CANNY, DANNY, ERNY, FRENNY}

Agents may be in one of eight possible states:

S_{shop}={BREAK, IDLE, NEGOTIATE, SELL, INNERTALK, WATCH, GUARD, EQUIP}

Having the two sets A and S, we can define the environment for a team:

Definition 1 (environment). Let A be a set of agents, and let S be a set of states. The pair $E = \langle A, S\rangle$ is called the *environment* of A over S.

Now that we have the definition of the environment, we can continue to define the relation between an agent and a state. In order to define the basic structures in terms of model-based diagnosis, we will use a first-order logic:

Definition 2 (position). A *position* function over an environment $\langle A, S\rangle$ is a function that *positions* an agent in a particular state: $\gamma : A \rightarrow S$. In terms of first order logic, we define the predicate $\gamma'(a_i, s_j) = true \Leftrightarrow \gamma(a_i) = s_j$. We will use shorthand and denote $\gamma'(a_i, s_j)$ as s^i_j.

As mentioned in the introduction, one of the novelties of this work is the possibility to gather joint coordinations to one structure. To this end, we present a function to set multiple states for an agent. To this end, we will define superposition:

Definition 3 (superposition). A *superposition* function over some environment $E = \langle A,\ S\rangle$ is a function $\Gamma : A \rightarrow \|S\|\backslash\emptyset$ i.e., it positions each agent in a *set* of possible states. Logically, $\Gamma(a_i) = S'^i \subseteq S \Rightarrow (\bigvee_{s_j \in S'^i} s^i_j) \wedge (\bigwedge_{s_j \in S\backslash S'^i} \neg s^i_j)$.

For example, let us refer back to the agents and states presented in the shop . $\gamma(Erny) = Guard$ is a position (s^{Erny}_{Guard}), while $\Gamma(Anny) = \{InnerTalk, Watch\}$ is a superposition. In first order logic:

$$(s^{Anny}_{InnerTalk} \vee s^{Anny}_{Watch}) \wedge$$
$$\neg s^{Anny}_{Break} \wedge \neg s^{Anny}_{Idle} \wedge \neg s^{Anny}_{Negotiate} \wedge \neg s^{Anny}_{Sell} \wedge \neg s^{Anny}_{Guard} \wedge \neg s^{Anny}_{Equip}$$

Figure 1 presents the full superposition function for the shop .

$$\Gamma(a) = \begin{cases} \{\text{INNERTALK, WATCH}\} & a = \text{ANNY} \\ \{\text{BREAK, SELL}\} & a = \text{BENNY} \\ \left\{ \begin{matrix} \text{BREAK, NEGOTIATE,} \\ \text{SELL, EQUIP} \end{matrix} \right\} & a \in \{\text{CANNY, DANNY}\} \\ \{\text{GUARD}\} & a = \text{ERNY} \\ \{\text{BREAK, INNERTALK}\} & a = \text{FRENNY} \end{cases}$$

Figure 1: A superposition function.

3.2 A Model of Coordination

The multi-agent systems of interest to us are composed of several agents, which (by design) are to satisfy certain coordination constraints. We call this type of system a *team*, to distinguish it from general multi-agent systems in which it is possible that no coordination constraints exist.

The states of agents in a team are coordinated. We utilize a coordination primitive to define the coordination constraints. The coordination states a non-binary constraint between agents' states, such that these states must be taken jointly, at the same time.

Definition 4 (coordination(CRD)). A coordination is a constraint between agents' positions, requiring them to be true concurrently. Logically, we represent this constraint as follows: $CRD(s_i^1, ..., s_k^n) \Rightarrow (s_i^1 \wedge, ..., \wedge s_k^n)$

For example, in the shop example above, an allowed coordination could be:

$CRD(s^{\text{ANNY}}_{\text{WATCH}}, s^{\text{BENNY}}_{\text{SELL}}, s^{\text{CANNY}}_{\text{NEGOTIATE}}, s^{\text{DANNY}}_{\text{BREAK}}, s^{\text{ERNY}}_{\text{GUARD}}, s^{\text{FRENNY}}_{\text{INNERTALK}})$

Unlike [Kalech and Kaminka, 2005] that define a binary constraint to represent a coordination only for pair of agents, we define the coordination between multiple agents by a non-binary constraint. In addition, we allow joint coordination concurrently. That means that an agent can be found in one of multiple states while other agents can be found in multiple states. Fundamentally, we can represent the joint coordination as a conjunction statement of coordination constraints. However, it is more efficient to define them using superposition (Definition 3).

Definition 5 (joint coordination). A joint coordination is a constraint between agents' super-position mandates that they must be true concurrently. We represent this constraint as follows: $CRD(A, S) \Rightarrow \bigcup_{a_i \in A}(\Gamma(a_i) = S'^i \subseteq S)$. Logically:

$$CRD(A,S) \Rightarrow \bigwedge_{a_i \in A} ((\bigvee_{s_j \in S'^i} s_j^i) \wedge (\bigwedge_{s_j \in S \setminus S'^i} \neg s_j^i))$$

The corresponding joint coordination for the superposition presented in Figure 1 is (only the true literals for each agent are shown):

$$\begin{aligned} CRD(A,S) = & (s^{Anny}_{InnerTalk} \vee s^{Anny}_{Watch}) \wedge \\ & (s^{Benny}_{Break} \vee s^{Benny}_{Sell}) \wedge \\ & (s^{Canny}_{Break} \vee s^{Canny}_{Negotiate} \vee s^{Canny}_{Sell} \vee s^{Canny}_{Equip}) \wedge \\ & (s^{Danny}_{Break} \vee s^{Danny}_{Negotiate} \vee s^{Danny}_{Sell} \vee s^{Danny}_{Equip}) \wedge \\ & (s^{Erny}_{Guard}) \wedge \\ & (s^{Frenny}_{Break} \vee s^{Frenny}_{InnerTalk}) \end{aligned}$$

This representation allows defining multiple constraints between the agents in the same structure. For example, while ANNY selects state INNERTALK or WATCH, BENNY must select BREAK or SELL and so on for all the agents.

3.3 A Model of Actions

At any given moment, each agent is in a given *state*. As a result of its state, each agent takes some *action*, in order to fulfill its goal. An action is visible, i.e. others might observe it. A state is not necessarily related to one particular action. Rather, it is possible that one of a few given actions will be taken at service of the same state. In the opposite direction, the same action might be taken at service of a few different states. We will annotate the actions as a set $B = \{b_1, b_2, \ldots, b_\ell\}$.

For example, in the shop we define eight states logical positions of the agents and nine actions, which the agents might act upon. State SELL, for example, is when an agent is busy with closing the deal with a customer. Positioned at this state, the agent might act in one of the actions GET (getting the product off the shelf), CARRY (carrying it to the customer) or COUNTER (sitting near the counter). On the other hand, an agent might also CARRY or GET while positioned at state EQUIP, and not only when positioned in SELL.

When designing a multi-agent system, the designer defines which actions might be taken by an agent when positioned in each state. This is called the *latitude* of the agent.

Definition 6 (latitude). Let $E = \langle A, S \rangle$ be an environment, and B be a set of actions, the *latitude* of any agent $a \in A$ is a function $\lambda_a : S \rightarrow \|B\| \backslash \emptyset$.

This function maps, for any agent $a \in A$ (rather than a certain agent as in definition 2), each state to a subset of actions which the agent is allowed to pick while being in this state. The straight-forward inverse function of λ_a, the function λ_a^{-1}, would map subsets of B to elements in S. While this function is not interesting, we do define a kind of 'inverse' to the latitude function:

Definition 7 (interpretation). Let $E = \langle A, S \rangle$ be an environment, and B be a set of actions, the *interpretation* $\forall a_i \in A$ is the function $\Lambda_{a_i} : B \rightarrow \|S\| \backslash \emptyset$. In terms of first order logic:

$$(\Lambda_{a_i}(b_k) = S' \subseteq S) \Rightarrow (\bigvee_{s_j \in S'} s_j^i) \wedge (\bigwedge_{s_j \in S \setminus S'} \neg s_j^i)$$

Λ_a of a given action b_j, is the set of all states that have b_j in their latitude. Given an action of any agent a', we *interpret* its action as one of a few given states, using this function. Figure 2 presents the latitude and interpretation functions for the shop example. For instance, an action $Phone$ taken by any agent, say BENNY, implies that its states are BREAK or NEGOTIATE, meaning:

$$\begin{aligned} & (s^{Benny}_{Break} \vee s^{Benny}_{Negotiate}) \wedge \\ & \neg s^{Benny}_{Idle} \wedge \neg s^{Benny}_{Sell} \wedge \neg s^{Benny}_{InnerTalk} \wedge \neg s^{Benny}_{Watch} \wedge \neg s^{Benny}_{Guard} \wedge \neg s^{Benny}_{Equip} \end{aligned}$$

This is the first-order representation of the interpretation presented in Figure 2(b):

$$\lambda(s) = \begin{cases} \{\text{TALK, PHONE, STAND, OTHER}\} & \text{BREAK} \\ \{\text{STAND}\} & \text{IDLE} \\ \{\text{TALK, PHONE}\} & \text{NEGOTIATE} \\ \{\text{GET, CARRY, COUNTER}\} & \text{SELL} \\ \{\text{TALK}\} & \text{INNERTALK} \\ \{\text{STAND, WALK, TALK}\} & \text{WATCH} \\ \{\text{STAND, WALK}\} & \text{GUARD} \\ \{\text{WALK, CARRY, PUT, GET}\} & \text{EQUIP} \end{cases}$$

(a) A latitude function

$$\Lambda(b) = \begin{cases} \left\{\begin{matrix}\text{BREAK, NEGOTIATE,} \\ \text{INNERTALK, WATCH}\end{matrix}\right\} & \text{TALK} \\ \{\text{BREAK, NEGOTIATE}\} & \text{PHONE} \\ \left\{\begin{matrix}\text{BREAK, IDLE,} \\ \text{WATCH, GUARD}\end{matrix}\right\} & \text{STAND} \\ \{\text{WATCH, GUARD, EQUIP}\} & \text{WALK} \\ \{\text{SELL}\} & \text{COUNTER} \\ \{\text{EQUIP}\} & \text{PUT} \\ \{\text{SELL, EQUIP}\} & \text{GET} \\ \{\text{SELL, EQUIP}\} & \text{CARRY} \\ \{\text{BREAK}\} & \text{OTHER} \end{cases}$$

(b) An interpretation function

Figure 2: A latitude function for the example of the shop , and its interpretation function.

$$\begin{aligned} \Lambda_{a_i}(Talk) &\Rightarrow s^{a_i}_{Break} \vee s^{a_i}_{Negotiate} \vee s^{a_i}_{InnerTalk} \vee s^{a_i}_{Watch}, \\ \Lambda_{a_i}(Phone) &\Rightarrow s^{a_i}_{Break} \vee s^{a_i}_{Negotiate}, \\ \Lambda_{a_i}(Stand) &\Rightarrow s^{a_i}_{Break} \vee s^{a_i}_{Idle} \vee s^{a_i}_{Watch} \vee s^{a_i}_{Guard}, \\ \Lambda_{a_i}(Walk) &\Rightarrow s^{a_i}_{Watch} \vee s^{a_i}_{Guard} \vee s^{a_i}_{Equip}, \\ \Lambda_{a_i}(Counter) &\Rightarrow s^{a_i}_{Sell}, \\ \Lambda_{a_i}(Put) &\Rightarrow s^{a_i}_{Equip}, \\ \Lambda_{a_i}(Get) &\Rightarrow s^{a_i}_{Sell} \vee s^{a_i}_{Equip}, \\ \Lambda_{a_i}(Carry) &\Rightarrow s^{a_i}_{Sell} \vee s^{a_i}_{Equip}, \\ \Lambda_{a_i}(Other) &\Rightarrow s^{a_i}_{Break} \end{aligned}$$

Now that we have a definition of the joint coordination (5) and the definition of the interpretation function (7), we can define the multi-agent system description (MASD). MASD is a set of implications from the normality of the agents to the correctness of the union of their superposition (based on the joint coordination) and their interpreted states (based on the interpretation). To define the normality of the agent we define the predicate $AB(a_i)$ which represents the abnormality of agent a_i (failing):

Definition 8 (multi-agent system description (MASD)). Given a set of agents $A = \{a_1, a_2, \ldots, a_n\}$, a set of states $S = \{s_1, s_2, \ldots, s_m\}$ and a set of actions $B = \{b_1, b_2, \ldots, b_\ell\}$, MASD is a set:

$$\begin{aligned} MASD = \quad & \{\neg AB(a_i) \Leftrightarrow (\Gamma(a_i) = S'^i \bigcup \Lambda_{a_i}(b_k) = S" \nvdash \bot) \\ & |S'^i \subseteq S \wedge S" \subseteq S \wedge a_i \in A \wedge b_k \in B\} \end{aligned}$$

This definition enforces the dependency between the perfection, or in terms of model-based diagnosis, the normality of the agents and the correctness of their selected states based on the joint coordination and the interpretation of their states by their actions.

3.4 A Model of Observation

Knowing the exact state of each agent at every time requires that the agent reports its state any time it is changed. This is usually infeasible, since it involves massive communication resources. Our model-based diagnosis approach suggests looking at the action of each agent. Thus the last building block we define is the observation.

Definition 9 (agent-action). Let $A = \{a_1, a_2, \ldots, a_n\}$ be a set of agents and $B = \{b_1, b_2, \ldots, b_\ell\}$ a set of actions, an *agent-action* is a function $\omega : A \rightarrow B$, that maps each agent to a particular action.

Definition 10 (observation (OBS)). A set of agent-actions:

$$OBS = \quad \{(\omega(a_i) = b_k) \quad |b_k \in B \wedge a_i \in A\}$$

In the the shop example, the observation can be:

$$\begin{aligned} OBS = \quad & \{\omega(Anny) \quad = Stand \\ & \omega(Benny) \quad = Stand \\ & \omega(Canny) \quad = Phone \\ & \omega(Danny) \quad = Get \\ & \omega(Erny) \quad = Carry \\ & \omega(Frenny) \quad = Walk\} \end{aligned}$$

4 Diagnosis of Coordination Faults

A fault in the coordination of a multi-agent system may be the result of a faulty agent(s). Given a $MASD$ (Definition 8) it is possible to infer that a fault exists and to generate hypotheses as to the abnormal agents, by checking whether the observed actions of the agents satisfy the $MASD$.

Let us formalize the coordination diagnosis in terms of model based diagnosis:

Definition 11 (Coordination Diagnosis Problem (CDP)). Given $\{A, MASD, OBS\}$ where A is a team of agents $\{a_1 ... a_n\}$, $MASD$ is a multi agent system description defined over A (Definition 8), and OBS is the set of the actions of the agents (Definition 10), then the *coordination diagnosis problem (CDP)* arises when

$$MASD \cup \{\neg AB(a_i) | a_i \in A\} \cup OBS \vdash \bot$$

Given a CDP, the goal of the coordination diagnosis process is to determine a minimal set of abnormal agents whose selection and subsequent setting of the $AB(\cdot)$ clause would eliminate the inconsistency. To this end we define the consistency-based coordination diagnosis:

Definition 12 (consistency-based coordination diagnosis (CBCD)). A minimal set $\Delta \subseteq A$ such that:

$$MASD \bigcup \{AB(a_i) | a_i \in \Delta\} \bigcup \{\neg AB(a_i) | a_i \in A - \Delta\} \bigcup OBS \nvdash \bot$$

In our example, MASD is not consistent with the observation. A diagnosis for this coordination fault can be: $\Delta = \{Erny, Frenny\}$.

The goal now is to find Δ. Consistency-based minimal diagnosis is known as NP-hard problem [de Kleer and Williams, 1987]. In particular, Kalech and Kaminka have

$C^{6\times 8} =$

	BREAK	IDLE	NEGOTIATE	SELL	INNERTALK	WATCH	GUARD	EQUIP
ANNY	0	0	0	0	1	1	0	0
BENNY	1	0	0	1	0	0	0	0
CANNY	1	0	1	1	0	0	0	1
DANNY	1	0	1	1	0	0	0	1
ERNY	0	0	0	0	0	0	1	0
FRENNY	1	0	0	0	1	0	0	0

Figure 3: The coordination-matrix representation of the joint coordination of Figure 1.

proposed algorithm to find consistency-based coordination diagnosis [Kalech and Kaminka, 2005]. However, in their paper the coordination is represented by binary constraints between pair of agents' states. On the other hand, in this paper we represent joint coordination setting (1) by a non-binary constraint between multi-agent, and (2) by joint coordination between multiple states of each agent, rather than single independent state. These two qualities enable an efficient representation of more realistic problems on the one hand, and on the other hand they simplify the representation so the diagnosis can be found even in linear time best case, in the number of agents and states. In the next section we propose a matrix-based representation presented in [Kalech *et al.*, 2007], which uses as the basis for an algorithm for coordination diagnosis in linear time.

5 Matrix-Based Representation

We will represent the models of the coordination, the actions and the observation by matrices.

Let $A = \{a_1, a_2, \ldots, a_n\}$ be a set of agents and $S = \{s_1, s_2, \ldots, s_m\}$ be a set of states. We represent the joint coordination of the agents (Definition 5) by a Boolean matrix of order $n \times m$.

Definition 13 (coordination-matrix). Let E be the environment $\langle A,\ S\rangle$. A *coordination-matrix* C over E is a Boolean matrix of order $n \times m$ ($C \in \mathbb{B}^{n\times m}$) provides:

$$c_{ij} = \begin{cases} 1 & s_j^i \in \Gamma(a_i) \\ 0 & \text{otherwise} \end{cases}$$

Given a set of states $S = \{s_1, s_2, \ldots, s_m\}$ and a set of actions $B = \{b_1, b_2, \ldots, b_\ell\}$, we can represent the interpretation of the actions to the states (Definition 7) by a Boolean matrix of order $\ell \times m$.

Definition 14 (interpretation-matrix). Let S be a set of states and B a set of actions, an *interpretation-matrix* I from B to S is a Boolean matrix of order $\ell \times m$ ($I \in \mathbb{B}^{\ell\times m}$) provides:

$$i_{ij} = \begin{cases} 1 & s_j \in \Lambda(b_i) \\ 0 & \text{otherwise} \end{cases}$$

Figure 4 presents the corresponding interpretation-matrix to the interpretation function presented in Figure 2(b). The rows represent the actions and the columns represent the states. For example, the second row says that once an agent is observed doing PHONE, then its state is one of {BREAK, NEGOTIATE}.

The last building block we define is the observation-matrix, which is parallel to the observation Definition (10) in the model-based diagnosis formulation.

$I^{9\times 8} =$

	BREAK	IDLE	NEGOTIATE	SELL	INNERTALK	WATCH	GUARD	EQUIP
TALK	1	0	1	0	1	1	0	0
PHONE	1	0	1	0	0	0	0	0
STAND	1	1	0	0	0	1	1	0
WALK	0	0	0	0	0	1	1	1
COUNTER	0	0	0	1	0	0	0	0
PUT	0	0	0	0	0	0	0	1
GET	0	0	0	1	0	0	0	1
CARRY	0	0	0	1	0	0	0	1
OTHER	1	0	0	0	0	0	0	0

Figure 4: The interpretation-matrix for the interpretation function presented in Figure 2(b).

$\Theta^{6\times 9} =$

	TALK	PHONE	STAND	WALK	COUNTER	PUT	GET	CARRY	OTHER
ANNY	0	0	1	0	0	0	0	0	0
BENNY	0	0	1	0	0	0	0	0	0
CANNY	0	1	0	0	0	0	0	0	0
DANNY	0	0	0	0	0	0	1	0	0
ERNY	0	0	0	0	0	0	0	1	0
FRENNY	0	0	0	1	0	0	0	0	0

Figure 5: An observation matrix.

Definition 15 (observation-matrix). Let $A = \{a_1, a_2, \ldots, a_n\}$ be a set of agents and $B = \{b_1, b_2, \ldots, b_\ell\}$ a set of actions, an *observation-matrix* Θ stands for the observation matrix representation:

$$\theta_{ij} = \begin{cases} 1 & \omega(a_i) = b_j \\ 0 & \text{otherwise} \end{cases}$$

Figure 5 presents an example to an observation matrix. The rows represent the agents and the columns the actions. Pay attention that in every row there is exactly a single '1' since every agent is observed in one action.

6 Diagnosis Procedure

A coordination fault occurs when the current agents' positions (Definition 2) do not match the expected coordination given by the coordination-matrix (Definition 13). Thus, if we know the current positions of the agents, we can say for sure whether the system has a fault or not. The exact state of each agent is known only to the agent itself. However, its action is observable. By observing its current action, we can infer the state in which the agent is found. This could be done using the formula:

$$\Omega = \Theta \cdot I \tag{1}$$

Where, Θ is the observation matrix and I is the interpretation matrix. Ω is an $n \times m$ Boolean matrix. Each element j in row i represents whether it is possible that agent a_i is now in state s_j ('1' entry) or not ('0' entry). Note that each element $\omega_{i,j}$ is the sum of multiplying each element k in row i of Θ by element k in column j of I. This multiplication, of course, is '1' iff both of them are '1'. Since each row in Θ has exactly one element which is '1', the value of each element in Ω will be at most '1'.

$\Omega^{6\times 8} = \Theta \cdot I =$

	BREAK	IDLE	NEGOTIATE	SELL	INNERTALK	WATCH	GUARD	EQUIP
ANNY	1	1	0	0	0	1	1	0
BENNY	1	1	0	0	0	1	1	0
CANNY	1	0	1	0	0	0	0	0
DANNY	0	0	0	1	0	0	0	1
ERNY	0	0	0	1	0	0	0	1
FRENNY	0	0	0	0	0	1	1	1

Figure 6: The matrix given by the product between the observation-matrix and the interpretation-matrix.

$R = \Omega \wedge C =$

	BREAK	IDLE	NEGOTIATE	SELL	INNERTALK	WATCH	GUARD	EQUIP
ANNY	0	0	0	0	0	1	0	0
BENNY	1	0	0	0	0	0	0	0
CANNY	1	0	1	0	0	0	0	0
DANNY	0	0	0	1	0	0	0	1
ERNY	0	0	0	0	0	0	0	0
FRENNY	0	0	0	0	0	0	0	0

Figure 7: The matrix given by boolean 'and' operation between the coordination-matrix C and Ω.

For example, Figure 6 presents the matrix given by the product between the observation-matrix (given in Figure 5) and the interpretation-matrix (given in Figure 4). Our observation may lead us to conclude that CANNY's state is either BREAK or NEGOTIATE.

We can now explain the diagnosis algorithm. Failure is defined as a situation wherein none of an agent's possible assigned state (according to Ω) appears on the 'allowed coordination', designated as C (the coordination-matrix). In order to examine possible matches we will operate a logical 'and' between C and Ω in an element-by-element process, to get the results matrix, $R^{n\times m}$:

$$r_{i,j} = c_{i,j} \wedge \omega_{i,j} \tag{2}$$

R represents all the agents-assigned combinations that satisfy C according to interpreted states by the observation. The combinations represented by R are all those that agent a_i is found in one of the states s_j that match '1' element in row R_i. Thus, if in each row i in R there is at least one '1' element, it implies that at least one combination exists. In this case, we may assume that the agents will be found in one of those joint states. If, however, R defines an all-zero row exists, then the assigned agents' states are definitely forbidden. In this case, a failure alert is warranted, and the diagnosis is that the agents that are represented by these all-zero rows are abnormal. This operation takes only $O(nm)$ operations (counting the '1's for m elements on each of R's n rows).

Returning to the shop example, matrix R in Figure 7 is the result of an element-by-element 'and' operation between C (Figure 3) and Ω (Figure 6). In this coordination-matrix, the two bottom lines, representing ERNY and FRENNY, are all-zero. No desired combination can explain their actions. A failure has been detected and the diagnosis is $\Delta = \{Erny, Frenny\}$.

In order to prove that the algorithm finds complete and sound diagnosis we will prove that all-zero row entails the abnormality of the agent represented by that row and vice versa. To prove this statement we should prove first two logical lemmas related to the consistency of the sets given by the superposition and the interpretation functions. To simplify the proof we define a set of states $S = \{s_1, s_2 \dots s_p\}$, and two subsets $S', S" \subseteq S$ ($S' \neq \emptyset$, $S" \neq \emptyset$), where S' represents the set given by the superposition function and $S"$ represents the set given by the interpretation function. We define the following statements:

1. $ST_1 : (\bigvee_{s_j \in S'} s_j) \wedge (\bigwedge_{s_j \in S \setminus S'} \neg s_j)$
2. $ST_2 : (\bigvee_{s_j \in S"} s_j) \wedge (\bigwedge_{s_j \in S \setminus S"} \neg s_j)$

Lemma 1. $S' \bigcap S" = \emptyset \Rightarrow ST_1 \wedge ST_2 \vdash \perp$

Proof: Without loss of generality, $ST_1 \Rightarrow \exists s_j \in S' = true$, but $S' \bigcap S" = \emptyset \Rightarrow s_j \in S \setminus S"$,
$ST_2 \Rightarrow s_j = false$.
Consequently, $ST_1 \wedge ST_2 \vdash \perp$. □

Lemma 2. $S' \bigcap S" \neq \emptyset \Rightarrow ST_1 \wedge ST_2 \nvdash \perp$

Proof: To prove consistency we need to show a truth assignment. Without loss of generality, assume $S' \bigcap S" = s_1$,
$\Rightarrow ST_1 = true \wedge ST_2 = true$.
Consequently, $ST_1 \wedge ST_2 \nvdash \perp$. □

Theorem 1. Given a coordination-matrix representation: $\exists i,\ 1 \leq i \leq n :\ \bigwedge_{j=1}^{m} r_{ij} = 0 \Leftrightarrow AB(a_i)$

Proof:

1. $\exists i,\ 1 \leq i \leq n :\ \bigwedge_{j=1}^{m} r_{ij} = 0 \Rightarrow AB(a_i)$ (soundness):
   Without loss of generality, assume $\bigwedge_{j=1}^{m} r_{1j} = 0$ and prove that $AB(a_1)$.
   $\bigwedge_{j=1}^{m} r_{1j} = 0 \Rightarrow \forall j :\ c_{1,j} \wedge \omega_{1,j} = 0$ (equation 2).
   (a) $c_{1,j}$:
      i. $c_{1,j} = \Gamma(a_1) = S' \subseteq S$ ($S' \neq \emptyset$) (Definition 13).
      ii. $\Gamma(a_1) = S' \Rightarrow ST_1 = (\bigvee_{s_j \in S'} s_j^1) \wedge (\bigwedge_{s_j \in S \setminus S'} \neg s_j^1)$ (Definition 3).
   (b) $\omega_{1,j}$:
      i. $\omega_{1,j} = \Lambda(\omega(a_1)) = S"$ ($S" \neq \emptyset$) (equation 1, Definitions 14, 15).
      ii. $\Lambda(\omega(a_1)) = S" \Rightarrow ST_2 = (\bigvee_{s_j \in S"} s_j^1) \wedge (\bigwedge_{s_j \in S \setminus S"} \neg s_j^1)$ (Definition 7).
   By (a) and (b): $\forall j :\ c_{1,j} \wedge \omega_{1,j} = 0 \Rightarrow S' \bigcap S" = \emptyset$
   By Lemma 1: $\Rightarrow ST_1 \wedge ST_2 \vdash \perp$
   Consequently by Definition 8: $AB(a_1)$.
2. $\nexists i,\ 1 \leq i \leq n :\ \bigwedge_{j=1}^{m} r_{ij} = 0 \Rightarrow \neg AB(a_i)$ (completeness):
   Without loss of generality, assume $r_{1,1} \neq 0$ and prove that $\neg AB(a_1)$.
   $r_{1,1} \neq 0 \Rightarrow c_{1,1} \wedge \omega_{1,1} = 1$ (equation 2).
   (a) $c_{1,1}$:
      i. $c_{1,1} = 1 \Rightarrow\ s_{1,1} \in \Gamma(a_1)$ (Definition 13).

ii. $\Gamma(a_1) = S' \Rightarrow ST_1 = (\bigvee_{s_j \in S'} s_j^1) \wedge (\bigwedge_{s_j \in S \setminus S'} \neg s_j^1)$.

iii. $\Rightarrow s_1^1 \in S'$

(b) $\omega_{1,1}$:

i. $\omega_{1,1} = 1 \Rightarrow s_{1,1} \in \Lambda(\omega(a_1)) = S$" (equation 1, Definitions 14, 15).

ii. $\Lambda(\omega(a_1)) = S$" $\Rightarrow ST_2 = (\bigvee_{s_j \in S"} s_j^1) \wedge (\bigwedge_{s_j \in S \setminus S"} \neg s_j^1)$ (Definition 7).

iii. $\Rightarrow s_1^1 \in S$"

From (a) and (b): $S' \bigcap S" \neq \emptyset$ $(s_1^1 \in (S' \bigcap S"))$

By Lemma 2: $\Rightarrow ST_1 \wedge ST_2 \nvdash \perp$

Consequently by Definition 8: $\neg AB(a_1)$.□

In order to detect failures by observations only, we define two policies of decision [Kaminka and Tambe, 2000]. The *optimistic policy* assumes that as long as the system is not proven to be faulty, no fault should be reported. Using this policy, one can never get a false alarm. If it reports a fault, then a fault has certainly occurred. The other policy is the *pessimistic policy*. This policy reports a fault in the system, unless it is completely confident that no fault has occurred. Using this policy, one can never get to a situation of an unreported fault. We have adopted here an optimistic policy, thus in matrix Ω we inferred *all* the possibilities of the states that could be taken by the observed agents. By generating the result matrix (R) we check if at least one of the interpreted joint–states of the observed agents is consistent with the desired coordination.

Sometimes, an agent cannot detect the exact action of one of its teammates. In this case, we can still provide a partial solution; the agent may assume that the teammate row is 'all-ones' (i.e., its action might be any action in the system). Although in this case we are likely to miss faults, we still keep the property of the optimistic policy, that is, report no false-alarms. If the system principally allows communication between agents, the agent may better solve the problem by explicitly communicate agents whose action are not observable for it.

7 Complex Coordination

One of the advantages of the matrix representation is the possibility to define complex coordinations [Kalech *et al.*, 2007]. One coordination-matrix will usually not suffice for a full desired coordination definition. Thus, the coordination-matrix we introduced earlier (Definition 13), may only partially define the allowed combinations in a desired coordination. For instance, in the shop example, assume ERNY could replace FRENNY in GUARD duty, the coordination-matrix in Figure 3 does not deal with this new relation. Moreover, we cannot add another state to C, by just changing $c_{6,7}$ $\langle$FRENNY, GUARD$\rangle$ from '0' to '1'. This would allow undesired combinations, such as ERNY and FRENNY guarding simultaneously. In this section we will briefly present the complex coordinations and then focus on the diagnosis aspects.

The most important operator used to join a few coordination-matrices is the 'or', notated as ' $\sqcup$ '. Defining two sets of coordination $C_1 \sqcup C_2$, means that the set of allowed combinations in the system is the union of all the combinations defined by C_1 and all the combinations defined by C_2. This operator may be extended to expressions of the kind $C_1 \sqcup C_2 \sqcup \cdots \sqcup C_p$.

There may be cases in which the use of $\sqcup$ is more difficult for the designer to describe the system. Thus, we present the second basic operator, 'and', which is notated by a ' $\sqcap$ '. The expression $C_1 \sqcap C_2$ represents all the combinations that are found in the intersection of those that are defined by C_1 and those defined by C_2. In fact, one might notice that any expression of the form $C_1 \sqcap C_2$, may be reduced to an equivalent coordination-matrix, that represents exactly the same set of combinations. This is the coordination-matrix that is the result of a logical-and in an element-by-element fashion between C_1 and C_2.

We call this extended structure of combined coordination-matrices using operators a *rule*. An example for a complex rule is:

$$\mathcal{R} = C_1 \sqcup ((C_2 \sqcup C_3) \sqcap (C_4 \sqcup C_5)) \sqcup C_6 \sqcup (C_7 \sqcap C_8 \sqcap C_9)$$

Back to the diagnosis problem, to find a diagnosis we should compare by 'and'ing operator the product matrix of the interpretation-matrix and the observation-matrix (Ω) against the coordination-matrix. Testing Ω against a rule $\mathcal{R} = C_1 \sqcup C_2 \sqcup \cdots \sqcup C_p$ is simple. One must perform the all-zero row test presented earlier for each of the p coordination-matrices. That is, for each C_k in $\mathcal{R}$, calculating the result matrix R_k by logically 'and'ing Ω with C_k in an element-by-element fashion, and then check whether R_k has all-zero row. Due to the nature of the operator ' $\sqcup$ ', it is enough to verify that at least one such R_k has no all-zero row, in order to conclude that the agents are coordinated. For 'and' operator, on the other hand, (for instance $C_1 \sqcap C_2$) the absence of the property of all-zero row must hold for *both* C_1 and C_2.

In fact we have shown [Kalech *et al.*, 2007] an algorithm which reduces a rule to a collection of coordination-matrices that are all combined by an or operator. Thus we could detect failure by 'anding' each one of the coordination-matrices with Ω, and check all-zero rows in the result matrices.

For the diagnosis purpose we should provide a set of abnormal agents. Based on the diagnosis definition we have presented here, an indication to a fault is once all the p coordination-matrices produce all-zero rows in the corresponding R matrices. Then each one of the matrices R_k produces a diagnosis. For recovery purpose we prefer to explore minimal diagnoses. A minimal diagnosis is a diagnosis which no proper subset of it is a diagnosis. To this end, during the diagnosis process we prune all the diagnoses that are not minimal. In order to model complex rules in terms of model-based diagnosis we should define the $\sqcup$ and $\sqcap$ operators. Intuitively, since our model is defined in first order logic, we can define these operators using the regular logical operators $\vee$ and $\wedge$. A formal representation is beyond the scope of this paper.

8 Summary and Future Work

In this paper we presented formalization for diagnosing coordination failures in multi agent systems, in terms of model-based diagnosis. In contrast to previous work, the model

presented in this paper is more efficient and reflects the real world, by defining non-binary constraints between the agents and by enabling to gather multiple states in one constraint.

To solve the diagnosis problem we defined a matrix-based notation for the fundamental parts of the diagnosis representation, which serves as a general framework for coordination design and definition in multi agent systems. Using this representation, we showed an efficient fault detection and diagnosis algorithm in a space and time complexity that is linear by the number of agents and states.

In the future we plan to add partial observations capabilities which will find the minimum set of agents that may together provide the full information, or at least the best possible information. Combining this with explicit communication among agents may result a system that is cheap in resources, yet very reliable. In addition, at the moment our algorithm assumes that the coordination among the team members is defined at the beginning and must be consistent along the system lifetime. However, real-world multi-agent systems are dynamic, and the desired coordination may change, so we plan to extend our algorithm to dynamic coordination.

Another interesting field of future research is using probabilistic values for observations, rather than binaries. In this way, rather than defining the policy as 'pessimistic' or 'optimistic', we will be able to define the probability that a fault has occurred at a given moment.

Acknowledgements

Much of the work on this paper was done in the course of my postdoctoral fellowship at the School of Engineering and Applied Sciences at Harvard University. My deep appreciation goes out to Prof. Barbara Grosz for her support during this time, and to Rani Nelken for his help in the logic sections.

References

[Bacchus and van Beek, 1998] F. Bacchus and P. van Beek. On the conversion between non-binary and binary constraint satisfaction problems. In *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI-98)*, pages 311–318, 1998.

[de Kleer and Williams, 1987] J. de Kleer and B. C. Williams. Diagnosing multiple faults. *Artificial Intelligence*, 32(1):97–130, 1987.

[Fröhlich *et al.*, 1997] Peter Fröhlich, Iara de Almeida Mora, Wolfgang Nejdl, and Michael Schröder. Diagnostic agents for distributed systems. In *ModelAge Workshop*, pages 173–186, 1997.

[Horling *et al.*, 1999] Bryan Horling, Victor R. Lesser, Regis Vincent, Ana Bazzan, and Ping Xuan. Diagnosis as an integral part of multi-agent adaptability. Technical Report CMPSCI Technical Report 1999-03, University of Massachusetts/Amherst, January 1999.

[Kalech and Kaminka, 2005] Meir Kalech and Gal A. Kaminka. Towards model-based diagnosis of coordination failures. In *American Association for Artificial Intelligence (AAAI-05)*, 2005.

[Kalech and Kaminka, 2006] Meir Kalech and Gal A. Kaminka. Diagnosis of multi-robot coordination failures using distributed csp algorithms. In *American Association for Artificial Intelligence (AAAI-06)*, 2006.

[Kalech *et al.*, 2007] Meir Kalech, Michael Lindner, and Gal A. Kaminka. Matrix-based representation for coordination fault detection: A formal approach. In *Proceedings of the sixth international joint conference on autonomous agents and multiagent systems (AAMAS)*, 2007.

[Kaminka and Tambe, 2000] Gal A. Kaminka and Milind Tambe. Robust multi-agent teams via socially-attentive monitoring. *Journal of Artificial Intelligence Research*, 12:105–147, 2000.

[Kim *et al.*, 2001] Phil Kim, Brian C. Williams, and Mark Abramson. Executing reactive, model-based programs through graph-based temporal planning. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2001.

[Kraus *et al.*, 1998] Sarit Kraus, Sycara Katia, and Amir Evenchik. Reaching agreements through argumentation: a logical model and implementation. *Artificial Intelligence*, 104(1–2):1–69, 1998.

[Matsubara *et al.*, 1998] Hitoshi Matsubara, Ian Frank, kumiko Tanaka-Ishii, Ituski Noda, Hideyuki Nakashima, and Koiti Hasida. Automatic soccer commentary and robocup. In Minoru Asada, editor, *the Second RoboCup Workshop (RoboCup-98)*, pages 7–22, Paris, France, 1998.

[Reiter, 1987] R. Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32(1):57–96, 1987.

[Roos *et al.*, 2004] Nico Roos, Annette ten Teije, and Cees Witteveen. Reaching diagnostic agreement in multi-agent diagnosis. In *Proceedings of Autonomous Agents and Multi Agent Systems (AAMAS-04)*, pages 1254–1255, 2004.

[Tambe *et al.*, 2005] M. Tambe, E. Bowring, H. Jung, G. Kaminka, R. Maheswaran, J. Marecki, P. J. Modi, R. Nair, S. Okamoto, J. P. Pearce, P. Paruchuri, D. Pynadath, P. Scerri, N. Schurr, and P. Varakantham. Conflicts in teamwork: hybrids to the rescue. In *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems (AAMAS–05)*, pages 3–10, 2005.

[Tambe, 1997] Milind Tambe. Towards flexible teamwork. *Journal of Artificial Intelligence Research*, 7:83–124, 1997.

[Williams *et al.*, 2001] B.C. Williams, P. Kim, M. Hofbaur, J. How, J. Kennell, J. Loy, R. Ragnoand J. Stedl, and A. Walcott. Model-based reactive programming of cooperative vehicles for mars exploration. June 2001.