Proceedings ACS (2019) 437–452

Problem Recognition, Explanation and Goal Formulation

Sravya Kondrakunta KONDRAKUNTA.2@WRIGHT.EDU
Venkatsampath Raja Gogineni GOGINENI.14@WRIGHT.EDU
Danielle Brown HENDERSON.66@WRIGHT.EDU
Department of Computing Science and Engineering, Wright State University, Dayton, OH 45435 USA

Matthew Molineaux MATTHEW.MOLINEAUX@WRIGHT.EDU
Michael T. Cox MICHAEL.COX@WRIGHT.EDU
Wright State Research Institute, Wright State University, Dayton, OH 45435 USA

Abstract

Goal reasoning agents can solve novel problems by detecting an anomaly between expectations and observations; generating explanations about plausible causes for the anomaly; and formulating goals to remove the cause. Yet not all anomalies represent problems. We claim that the task of discerning the difference between benign anomalies and those that represent an actual problem by an agent will increase its performance. Furthermore, we present a new definition of the term "problem" in a goal reasoning context. This paper discusses the role of explanations and goal formulation in response to developing problems and implements the response. The paper illustrates goal formulation in a mine clearance domain and a labor relations domain. We also show the empirical difference between a standard planning agent, an agent that detects anomalies and an agent that recognizes problems.

Keywords: Goal reasoning, cognitive architecture, case-based reasoning, explanation, goal formulation

1. Introduction

An intelligent, autonomous agent in a partially observable world should formulate its own goals, make plans to achieve those goals and successfully execute those plans. An agent can formulate its own goals based on an anomaly, i.e., the difference between an expected state and an observed state, by generating a hypothesis that explains this anomaly and generating new goals that respond to the hypothesis. However, many anomalies that arise in the real world do not represent a problem to an agent's mission or goal. For example, the playing of unexpected loud music may or may not be a problem for roommates. If a roommate is preparing for an upcoming exam, the music is a problem. If, on the other hand, she is doing her laundry, it is not a problem. Generally speaking, an agent does not need to respond to every observed anomaly; an intelligent agent should be capable of distinguishing between those that signal a problem and those that do not.

The *Goal Driven Autonomy (GDA)* approach (Cox, 2007; 2013; Molineaux et al., 2010; Munoz-Avila et al., 2010) to agency represents an appropriate response for an autonomous agent's anomalies. However, the existing research does not formally address the issue of which anomalies should be considered problems; for example, ARTUE (Klenk, Molineaux, & Aha, 2011) motivations implicitly consider certain situations to require new goals, but the system provides no formal basis for determining whether a certain situation is problematic. In this paper, we consider the task of recognizing whether an anomaly should constitute a problem for an agent. Performing this efficiently will improve both the effectiveness and robustness of the agent. We use the term *problem* in this paper to refer to anomalies that require a response in order to meet the agent's goals. Then, we present such an approach and further discuss the role of explanations and goal formulation in recognizing and responding to problems. We show empirical results illustrating the effectiveness of this approach as part of a GDA agent in two uncertain, dynamic environments.

The paper continues as follows. Section 2 defines the problem recognition task and presents a formalism to represent the problem. Section 3 describes *explanation patterns (XP)* and their role in understanding problems as well as the goals formulated when an agent detects a problem. Section 4 discusses the implementation of the problem recognition task in two domains: mine clearance domain and labor relations domain. The evaluation of GDA agents and the presentation of the results follows in Section 5. We consider related research in Section 6 and conclude with Section 7 with a discussion about some ideas for future research.

2. Formalities and Notation

An anomaly occurs when the expected state of the agent does not match its current observed state, but a problem only occurs when the agent needs to address the above anomaly. As such, problem recognition refers to reasoning about the anomaly and deciding whether it is something that the agent needs to handle. There might be different types of problems as well as different ways to recognize and address them. One such problem is the planning problem.

2.1 Classical Planning Problem Representation

A classical planning domain is defined as a finite state-transition system in which each state $s \in S$ is represented by a finite set of ground atoms (Ghallab, Nau, & Traverso, 2004). A planning action model is a triple $\alpha = (head(\alpha), pre(\alpha), eff(\alpha))$, where $pre(\alpha)$ and $eff(\alpha)$ are preconditions and effects. Each action $a \in A$ is a ground instance of some action model. An action a is executable in a state s if $s \vDash pre(a)$. The state-transition system is a tuple $\Sigma = (S, A, \gamma)$, where S is the set of all states and A is the set of all actions as above. In addition, γ is a state transition function $\gamma : S \times A \rightarrow S$ that returns the resulting state following action execution.

A classical planning problem is a triple $P = (\Sigma, s_0, g)$, where Σ is a planning domain, s_0 is the initial state, and the goal g is a conjunction of first-order literals. A state s_g satisfies a goal if $s_g \vDash g$; in this situation we refer to s_g as a goal state. A plan $\pi_g \epsilon\, \Pi$ represents a solution to P if it consists of a sequence of plan steps $(a_1, a_2, \ldots a_n)$ that incrementally changes the world, starting from the initial state s_0 and ending in a goal state. That is, it is a solution if $\gamma(\pi_g, s_0) = \gamma(\ldots\gamma(\gamma(s_0, a_1), a_2)\ldots, a_n) \vDash g$.

2.2 Extended Planning Problem Representation

Our extended definition of planning problems is intended for usage during execution. New problems arise during problem solving and execution, necessitating updated solutions. These iterated online planning problems therefore incorporate an agent's prior expectations and knowledge about the execution context. The extended problem also considers whether an agent should formulate new goals in response to the changing world. Solutions to the extended problem comprise, in addition to a plan, an explanation of an encountered anomaly and an updated goal agenda. Critically, the plan need not solve every goal in the goal agenda. Formally, we define an extended planning problem $\mathcal{P}_x = (s_c, s_e, Bk, H_c)$ where:

- $s_c \in S$ is the current state of the environment,
- $s_e \in S$ is the state the agent previously expects to hold at this point in time,
- Bk is the agent's background knowledge,
- H_c is an episodic history.

A solution Ψ_x to a problem $\mathcal{P}_x$ is of the form $(\hat{G}_c, g_n, \chi_c, \pi_c)$ where:

- $\hat{G}_c$ is an updated goal agenda (set of pending goals for the agent to accomplish),
- g_n is a goal from the agenda (i.e., $g_n \in \hat{G}_c$) chosen to address next,
- χ_c is an causal explanation that accounts for any discrepancy between s_c and s_e, and
- π_c is a plan that will accomplish g_i, at least one goal, $g \in \hat{G}_c$ in at least one possible world.

Note that the subscript "*c*" refers to the current iteration throughout this definition and later in this section; as the extended planning problem is iterative, this explicitly links present outputs to future inputs.

The agent's background knowledge Bk is a tuple $(\Sigma,\ \Delta)$ consisting of the planning domain Σ as defined in the classical planning problem along with a set Δ of goal operation models $\delta = (head(\delta),\ parameter(\delta),\ pre(\delta),\ res(\delta))$. In a goal operation $pre(\delta)$ and $res(\delta)$ are a set of preconditions and a result. The transformation's identifier is $head(\delta)$, and its input goal argument is $parameter(\delta)$. Any goal operation model δ with no input goal (written $parameter(\delta) = \emptyset$) models a *goal formulation* operation. Otherwise, δ represents *goal change* operations. Collectively, the agent's goal operation models Δ define an interpretation function $\beta\colon S \times G \rightarrow G$ that transforms an earlier goal into a desired goal (Cox, Dannenhauer, & Kondrakunta, 2017). See prior work for additional detail on the function β.

The episodic history H_c is a tuple $(\hat{G}_h, \pi_h, \varepsilon_h, \chi_h)$ that includes the agent's memory of past goal agendas $\hat{G}_h = (\hat{G}_1, \hat{G}_2, \ldots \hat{G}_{c-1})$, plans $\pi_h = (\pi_1, \pi_2, \ldots \pi_{c-1})$, and causal explanations $\chi_h = (\chi_1, \chi_2, \ldots\ \chi_{c-1})$, as well as the execution history containing states and actions $\varepsilon_h = (s_0, a_1, s_1, a_2, \ldots s_{c-1}, a_c)$.

2.3 Problem Recognition Subproblem

We describe three subproblems we address: problem recognition, goal formulation and change, and replanning. Problem recognition requires the agent to determine what problem, if any, occurred. Specifically, given the extended planning problem tuple (s_c, s_e, Bk, H_c), problem recognition outputs a problem, root cause, and explanation pair (d, ω_c, χ_c) such that:

$$\exists d, g_i\colon s_c \vDash d \wedge s_e \nvDash d \wedge restricts\ (g_i, d, \Sigma) \wedge g_i \in \hat{G}_{c-1} \wedge \omega_c \xrightarrow[\chi_c]{} d \wedge\ \omega \notin \hat{G}_{c-1}$$

This is read: There is some discrepancy, a literal conjunct d, that was observed (i.e., entailed by s_c) but not expected (i.e., not entailed by s_e). The discrepancy d restricts how the agent can address a goal g_i in its legacy goal agenda $\hat{G}_c$. The discrepancy d was caused by a root cause ω_c (a literal conjunct) according to explanation χ_c. Finally, the root cause ω was not intended (i.e., not in the legacy agenda $\hat{G}_c$). In the above, a goal g_i is *restricted by* a discrepancy d if there is *no plan* π that can accomplish the goal g_i without eliminating discrepancy d. The explanation χ_c is a logical derivation or proof tree. When this condition is met, the discrepancy d is considered a problem.

2.4 Goal Formulation and Change Subproblem

Once a problem is recognized, the agent must update its goal agenda to respond. Given a current state s_c, problem cause ω_c, explanation χ_c, and legacy goal agenda $\hat{G}_{c-1}$; goal formulation and change must find a new goal agenda $(\hat{G}_c, g_n)$ that responds "appropriately". Currently, no recognized definition of what constitutes a correct or strong goal agenda outside of a particular target problem exists.

2.5 Replanning Subproblem

The replanning subproblem finds a revised or new plan to accomplish the newly formulated goal g_n. This problem is much the same as the classical planning problem, adding only a legacy plan. It is defined by the tuple $(\Sigma, s_c, g_n, \pi_{c-1})$, and the solution is the new plan π_c. We do not further consider replanning within the scope of this paper, choosing to reuse results of other research.

3. Explanation and Goal Formulation

In this work, an *explanation* is a causal structure that represents a hypothesis about the cause of an anomaly. Problem explanations hypothesize the cause of an anomaly that limits an agent's goals. For example, in the context of doing laundry, an explanation such as "thoughtless neighbors cause loud music" is not a problem explanation. However, in the context of studying for an exam, it *is* a problem explanation. Moreover, reasoning about a hypothesis allows an agent to formulate its own goals to deal with a problem. In a nutshell, explanations help an agent to decide whether an anomaly is a problem or not, while goal formulation helps an agent to resolve a problem.

To implement our ideas, we modified Meta-AQUA (Cox & Ram, 1999), an open-source story understanding system, to generate grounded explanations. In prior work, Meta-AQUA was supplied with a case-base of explanation patterns that explain anomalous actions performed by actors in a story. In our work, we extended Meta-AQUA to retrieve an explanation from memory and adapt it for goal formulation whenever an agent encounters a problem.

3.1 Explanation Pattern

In our work, we use a case-base of problem explanations engineered manually to fit the domain. Each explanation in our case-base is an abstract *explanation pattern (XP)* (Schank, 2013) as shown in Figure 1. An XP is a data structure that represents a causal relationship between multiple states and/or actions; variables adapted during or after case retrieval abstractly define each action/state. An action or state is referred to as a node and different types of nodes are described based on their role in an XP as follows:

- Explains node: An observed unpredictable action/state (i.e., the target of the XP).
- Pre-XP node: An observed action/state along with the Explains node.

- XP-asserted node: An action, state, or XP that contributes to the explanation's cause. In the case of a causing XP, the effects of the cause XP can be seen as direct causes and the causes of that XP indirect causes, of the effect XP.

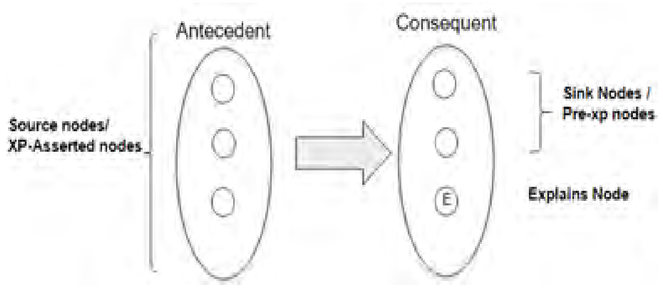
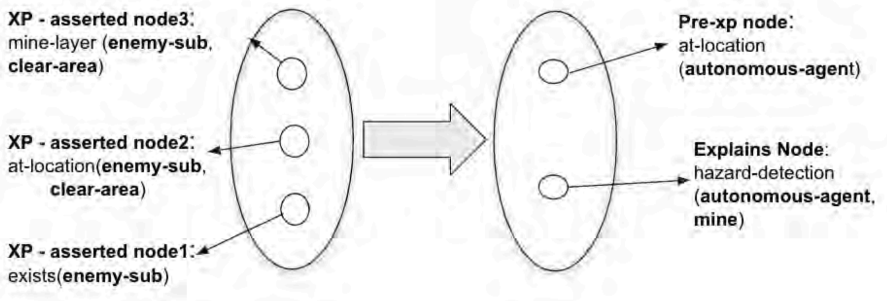


Figure 1. XP Structure.

An explanation pattern represents a causal structure in which XP-asserted nodes form an antecedent and Pre-XP nodes form the consequent. The Pre-XP nodes represent those states that must hold for the XP to be a candidate, including the Explains node itself.



XP - asserted node3: mine-layer (enemy-sub, clear-area)
XP - asserted node2: at-location(enemy-sub, clear-area)
XP - asserted node1: exists(enemy-sub)
Pre-xp node: at-location (autonomous-agent)
Explains Node: hazard-detection (autonomous-agent, mine)

Figure 2. Mine-XP
Bolded symbols represent variables (e.g., enemy-sub, autonomous-agent)

Figure 2 represents an example of an abstract explanation pattern i.e., an explanation pattern whose nodes refer to unbound variables, Mine-XP. It represents the causal pattern of an enemy submarines' path to a specific location and its mine laying action, later these mines might be detected by an autonomous agent if they are on the transit to survey some areas where mines are expected. As mentioned earlier, an XP constitutes of Explains node, Pre-XP nodes and XP-asserted nodes. The whole pattern can be structured as follows:

- XP-asserted node1: An enemy exists (sate)
- XP-asserted node2: enemy submarine at the clear-area (state)
- XP-asserted node3: enemy submarine laying mines (action)
- Explains node: Autonomous agent detects a mine (action).
- Pre-XP node: The autonomous agent is at the specific location (state).

Similarly, all the causal patterns can be structured as explanation patterns. However, approaches to learn such explanations are outside the scope of this paper, but (Ram, 1993) has sketched out a detailed approach on learning explanation patterns.

3.2 Retrieving an Explanation Pattern

Meta-AQUA constantly tries to unify the Explains node of each abstract XP with each new state or action. When such a unification is successful, the Pre-XP nodes of the corresponding case combines with the observations of corresponding states or actions from the story. Also, if this unification is successful, the case is retrieved, and a set of variable bindings are created that join the XP with the story. Substituting these variable bindings into the consequent and the variables to fill the antecedents reuses the retrieved abstract XP. However, after retrieval, if the XP-asserted nodes in the reused XP contain hypothetical information, then the agent can revise the hypothetical information further using the new knowledge obtained from the observations.

In our work, whenever an agent observes an anomalous state or action, the above retrieval process gets a problem explanation from the case-base. In general, the agent might retrieve zero, two, or more explanations. However, for the purpose of this paper we assume the agent retrieves exactly one problem explanation if an anomaly is a problem. However, in the future, we would like to address this assumption by select one problem by evaluating multiple explanations.

In an example as discussed in the previous section, when an autonomous agent detects a mine in a clear area where mines are not expected then Meta-AQUA tries to unify the states and actions with all the XP's in the case-base. Mine-XP in figure 2 becomes a candidate XP and goals are formulated from it as discussed in the next section.

3.3 Goal Formulation

Goal formulation is essential for an intelligent agent to respond to unpredictable events. In our work, we perform formulation by chaining backward on each of the antecedents of a retrieved XP until we reach all the antecedents to which the agent can respond. Antecedent nodes include actions and states; therefore, when the agent wishes to prevent an undesired consequent from recurring, it considers as potential goals the elimination of the actors that performed antecedent actions, or objects that participate in antecedent states. The removal mapping function that takes in the agents or objects and outputs the goals that eliminate them create potential goals.

In the example discussed in section 3.1, when Mine-XP becomes the candidate XP, elimination of agents in the XP- asserted nodes are considered as potential goals. So, the formulated goal is to apprehend the enemy ship.

4. Domains

To illustrate these concepts in this paper and to assess the performance of the resulting GDA agents, it will be useful to consider them in the context of the following concrete examples from the mine clearance domain and the labor relations domain.

4.1 Mine Clearance Domain

To prepare a harbor for use during maritime operations, it is essential to conduct mine clearance activities to ensure that ships can operate safely as they transit between the open sea and the port in the harbor. As searching and clearing mines in the entire harbor is likely to be a time-consuming and expensive undertaking, a network of safe shipping lanes is typically established to reduce the size of the area within the harbor that needs clearing. Such a system is known as a Q-route (Li, 2009).

For experimentation, we created simulated scenarios with a fixed Q-route that consists of a single shipping lane. In simulation, an Autonomous Underwater Vehicle (AUV) controlled by an agent performs both mine detection and clearance. In each scenario, the agent knows of two previously identified areas within the Q-route – green area one (GA1) and green area two (GA2) – where mines are expected. As such, any mines encountered which do not lie within GA1 or GA2 constitute anomalies. However, only anomalous mines within the Q-route are classified as problems, because mines outside the Q-route will not pose a hazard to shipping. It is the role of the agent to determine how to respond to these anomalous mines in each scenario.

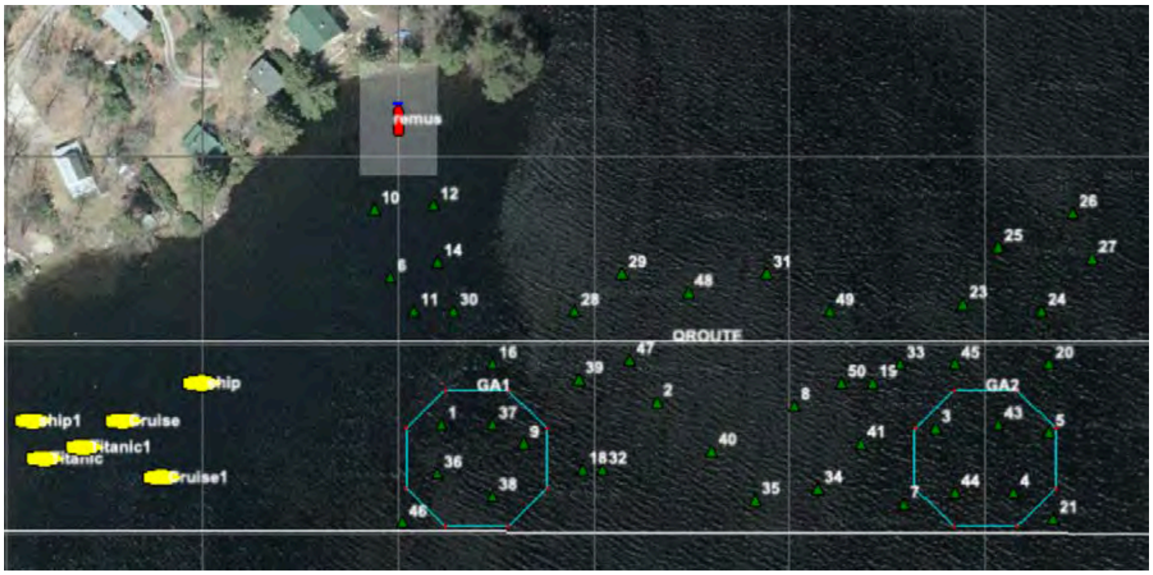


Figure 3. Simulation of the mine clearance domain in Moos IvP. The Q-route extends from the left to the right side of the map and represents the path that ships (6 yellow shapes on the left) will traverse. The remus AUV (red) must attempt to clear mines in green areas GA1 and GA2 to support the goals of ships reaching the shore. Unexpected mines exist within and outside of the route, and the AUV must decide which are problems.

Figure 3 illustrates the mine clearance domain. The red, cylindrical object in the top left corner represents the AUV named Remus and each triangle represents a ground truth mine position, not given to the agent a priori. The area between the two horizontal lines represents the Q-route, and the octagons on the left and right represent GA1 and GA2 respectively. In each scenario, the mines are uniformly distributed throughout the transit area (used by Remus to enter and exit the Q-route) and the Q-route.

4.1.1 Example Mine Clearance Problems

In this section, we give a series of example extended planning problems in the Mine Clearance domain and their solutions. In the Mine Clearance domain, the initial goal agenda ($\hat{G}_c$) includes the following goals: clear the mines in GA1, clear the mines in GA2, and head back to its initial position (home). Let us assume that the agent selects the goals in the provided order so the current goal of the agent would be to clear the mines in GA1. Therefore, initially g_0 = cleared-mines(remus,ga1) is the goal. The expectation of the agent is that the mines are present only in GA1. The initial plan (π_0) is comprised of several steps to achieve the first of its initial goals (g_0); some of which are the following:

π = { a_1 =move_to_the_location (remus, home , location-a),
a_2 =move_to_the_location (remus, location-a , location-b),
a_3 =move_to_the_location (remus, location-b , ga1),
a_4 =survey (ga1),
a_5 =identify-mines (ga1),
a_6 =clear-mines (ga1)}

Here location-a and location-b are locations of waypoints outside the Q-route that describe a path to GA1 from "home", the agent's launching point. After reaching GA1, the plan directs the Remus to survey locations in GA1, locate/identify any mines and clear them. In the initial state (s_0), GA1 is expected to (likely) contain one or more unknown mines. After g_0 is satisfied, however, the agent has cleared all mines in GA1.

The agent detects a mine at location-b after it completes the action a_1. This is an anomaly because in the observed current state (s_c), there is a mine at location-b, but not in the expected state (s_e). This triggers problem recognition. However, no explanation is found: a Pre-XP node of the explanation pattern remains false as the detected mine is outside of the Q-route. The anomalous mine is therefore not considered a problem, and goal formulation does not occur. As such, the new solution is $\Psi_x = \left(\hat{G}_0, \emptyset, \emptyset, tail(\pi_0)\right)$, containing the unchanged initial goal agenda, no new goal or explanation, and the remaining actions in π_0. The history $H_c = \left(\hat{G}_h, \pi_h, \varepsilon_h, \chi_h\right)$ is updated with the unchanged goal agenda, updated plan, state, action, and explanation.

After the agent achieves g_0, it travels towards GA2 to achieve the next goal, g_1 = cleared-mines(remus,ga2). Its new plan visits intermediate locations location-d and location-e in the route to clearing mines in GA2. In the expected state (s_e), the Remus is at location-e between GA1 and GA2, where there are no mines. However, in the current state (s_c), a mine is observed. As described previously, this anomaly triggers problem recognition. This time, the anomaly is a problem, as the situation matches the explanation pattern, producing an explanation χ_c with a root cause ω_c = mine-at(location-e) ∧ in-qroute(location-e). Goal formulation then generates a new goal $g_n = \neg$mine-at(location-e), and replanning produces a plan π_c that achieves g_n; the updated goal agenda adds this new goal: $\hat{G}_c = \hat{G}_{c-1} \cup \{g_n\}$. This completes the solution $\left(\hat{G}_c, g_n, \chi_c, \pi_c\right)$, which is used to update the history H_c as before. The agent then switches to the new plan π_c.

4.2 Labor Relations Domain

This domain describes a virtual institution, consisting of an institute head, employees, and customers. The institution starts with an initial reputation and budget represented by numeric values. The head of the institute enacts policies; implementing a policy takes a known fixed amount of budget and increases the institute's reputation value by a fixed amount. Disagreements about enacted policies may occur between the head and the employees with certain intensity. Intensity is a numeric value; high intensity disagreements may lead to a strike. Intensity values vary for each disagreement and are unpredictable. Disagreements can be resolved by negotiating. Negotiating to solve a disagreement also requires a budget, which varies with respect to the intensity of the disagreement. Moreover, negotiations decrease the reputation value by a function of intensity.

The agent acts as the head of the institute with a goal to increase the reputation of the institute, and, in order to achieve that goal, the agent must implement some policies. The expectation is that the employees agree to the policy, but, if the employees disagree then this is considered an anomaly, and only those disagreements that can lead to a strike are considered as problems. Therefore, the intelligence of the agent lies in identifying the anomalies that might lead to strikes and working on them. This domain is not related to trading and marketing agents in artificial intelligence, and is not simulated using third-party software.

4.2.1 Example Labor Relations Domain Problems

In the Labor Relations domain, the current goal agenda ($\hat{G}_c$) include: increasing the reputation of the institution and negotiating with employees. The initial goal g_i = increased reputation(institute, five). The agent expects (s_e) that the policy should be accepted by all the employees. If the employees do not accept the policy, then an intensity value of the rejection is provided. If that intensity value is low then it is not considered a problem, but if the value is high then it is a problem. Therefore, the agent explains an anomaly to be a problem based on the intensity value.

Similar to the Mine Clearance domain, if the intensity value is less than 35, then the observed current state (s_c) is not the same as the expected state (s_e) and a rejection of policy among employees occurs. This triggers problem recognition. However, no explanation is found: a Pre-XP node of the explanation pattern remains false as the intensity is less than 35. The anomalous situation is therefore not considered a problem, and goal formulation does not occur. As such, the new solution is $\Psi_x = \left(\hat{G}_0, \emptyset, \emptyset, tail(\pi_0)\right)$, which contains the unchanged initial goal agenda, no new goal or explanation, and the remaining actions in π_0. The history $H_c = \left(\hat{G}_h, \pi_h, \varepsilon_h, \chi_h\right)$ is updated with the unchanged goal agenda, updated plan, state, action, and explanation.

In this domain, the first goal is repeated iteratively until the agent is out of resources. Let us say an anomaly occurred again with an intensity value higher than 35. Once again the explanation is triggered. This time, the anomaly is a problem, as the situation matches the explanation pattern, producing an explanation χ_c with a root cause ω_c = disagreement-with(policy) ∧ disagreement-intensity(high). Goal formulation then generates a new goal $g_n =$ negotiated(employees), and replanning produces a plan π_c that achieves g_n; the updated goal agenda adds this new goal: $\hat{G}_c = \hat{G}_{c-1} \cup \{g_n\}$. This completes the solution $\left(\hat{G}_c, g_n, \chi_c, \pi_c\right)$, which is used to update the history H_c as before. The agent then switches to the new plan π_c.

5. Evaluation of the Implementation in the Domains

In both domains, in order to perform the evaluation we have introduced two other agents along with our GDA agent, namely an eager agent and a baseline agent. They each respond differently to anomalies. The GDA agent detects all anomalies, but only works on those perceived as problems. In contrast, the eager agent addresses all anomalies that it encounters, i.e., it tries to fix every anomaly it encounters. The baseline agent plans only for its original goals and ignores all anomalies. We assessed the performance of the three agents by varying the environment and averaging the results for 10 different runs for each scenario and presented the results.

5.1 Empirical Results in Mine Clearance Domain

We calculated the performance of the agents based on the number of the ships that reach the other side of the harbor safely in various mine density scenarios. Each scenario includes a total of six ships and three mine densities: low, medium, and high. We also introduced deadlines ranging from 0 to 2 seconds in the domain with increments of 0.5 seconds. Please note that the seconds indicate the simulation time, not a real world time. These deadlines specify the time gap between the agent starting from home to clear the mines and the ships starting their journey from one end of the shore to other.

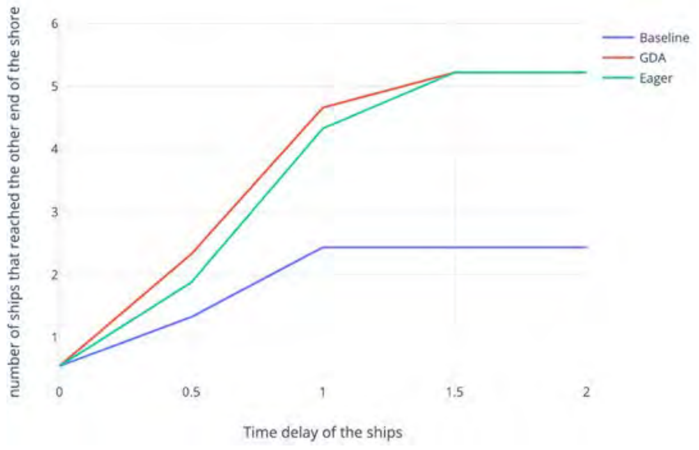


Figure 4. Scores obtained by the agents in mine clearance domain.

Figure 4 shows the scores achieved by the three different agents in all mine density (the average of low, medium, and high density) scenarios for the varied deadlines. The X-axis depicts the delay with which the ships start and the Y-axis indicates the number of ships that safely traverse the Q-route. Here, when we start looking at the values from the left side of the graph, at the delay of 0, very few ships were able to traverse the Q-route successfully, for all three agents. Those that were able to reach the other side were able to cross the Q-route in the low mine density scenarios, while very few or no ships made it across in the medium and high mine density scenarios.

To understand what a delay of 0.5 seconds means, consider what each agent can accomplish within that timeframe in a characteristic scenario. After 0.5 seconds, the baseline agent clears the mines in GA1 and is on its way to clear the mines in GA2. At the same point in time, the GDA agent has cleared the mines in GA1 as well as some mines within the Q-route; the eager agent has cleared mines outside of the Q-route and in GA1.

After 1 second, the baseline agent has cleared the mines in both green areas and is headed towards home, the GDA agent has cleared some mines within the path from GA1 to GA2 and some mines within GA2, whereas the eager agent is working on the mines within the Q-route after clearing the ones in GA1.

In these conditions (delay of 0.5 and 1 seconds), the difference between the performance of the GDA and eager agents does not seem very large. This is because the average of the various mine density scenarios also contains low mine density fields where the two agents perform almost identically, since the eager agent only has a few mines to clear outside of the Q-route. The experiment is also performed on the scenario with only one problem, which is caused a minimal difference between the agents. However, the difference in performance for medium and high mine density scenarios is two ships and it is significant for just one problem situation.

At a delay of 1.5 seconds or greater, all agents have performed all clearance tasks intended; thus, performance does not change for delays greater than 1.5 seconds.

5.2 Empirical Results in Labor Relations Domain

To assess the performance of the three agents in this domain, we compare the reputation values of all agents after they implement a certain number of policies. There are some numerical values in this domain: initial reputation is 500 and total budget is \$4000. Implementing any policy reduces the budget by \$25.

The intensity value of a disagreement is a random number between 1 and 100. When a disagreement arises, the employees can demand a budget amount, which is a random number between 1 and 25. Providing the budget amount of any amount within 40% to 60% of the amount demanded by employees solves the disagreement.

If there is no disagreement when a policy is implemented then the reputation of the institution is increased by five. However, if the agent encounters a disagreement, then it has two options: to solve the disagreement or to ignore the disagreement and strictly adhere to its initial policy. In the first option that addresses the disagreement, the reputation is neither decreased nor increased, i.e., the change in reputation is zero. If the agent does not address the disagreement, then the reputation value is decreases as a function of intensity value. So, if the intensity is <=34 then Rep = -Int/100 and, if the intensity is >=35 then Rep = -[(n+2)*Int]/100 where n = integer((I-35)/5). The integer() acts as a rounding function. There is an interest value added to the budget after implementing every 50 policies with a rate of 2.5%. Reputation values can become negative if the agent does not address disagreements.

Figure 5 shows the reputation achieved by the three different agents over 200 policies. The X-axis depicts the number of policies implemented and the Y-axis indicates the reputation value scaled to 10. All the agents have an initial reputation of five. Each point on the lines contains average of 20 policies and the reputation value is cumulative and can reach a max value of 10. In this experiment if any agent is out of budget then it starts to behave as a baseline agent as the debt should be as minimum as possible when the agent implements all policies.

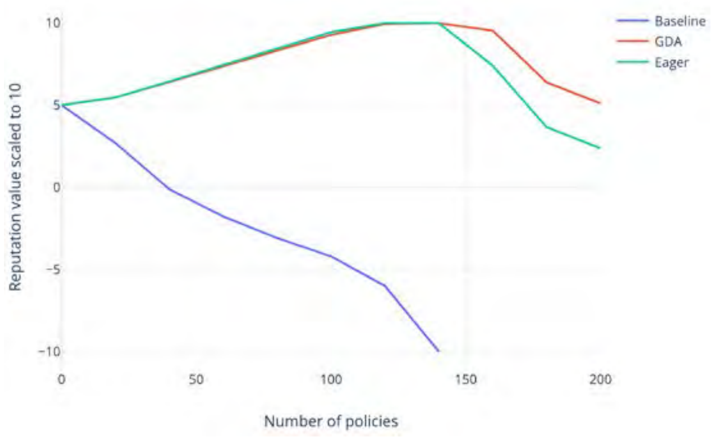


Figure 5. Scores obtained by the agents in labor relations domain.

Starting with the baseline agent, at the completion of 50 policies the baseline agent already has a negative reputation because it does not address any of the disagreements, so the reputation value drops and keeps on decreasing monotonically.

The GDA agent gets a little behind the eager agent from 50 to 120 policies because the GDA agent will not address the anomalies with lower intensity values, whereas, the eager agent addresses all the anomalies and spends its budget on every anomaly. However, this means the eager agent is out of its budget much sooner than the GDA agent.

The eager agent is out of budget at 140 policies and starts behaving as a baseline agent, and begins to drop its reputation, while the GDA agent preserves its budget and continues on its increment streak of the reputation value for around 150. The baseline agent's reputation value is still sinking to much lower values but the scale is adjusted such that the negative values are only visible up to -10. These results indicate that the GDA agent should perform better overtime and maintain a higher reputation than the eager agent by a significant amount. The GDA agent underperformed by a negligible amount for a period of time when compared to eager agent because of the higher amount of resources. A smart agent is not needed if the amount of resources present are infinite, but this is not very realistic and as long as resources have a limit, then there will be a need for the GDA agent to use them sustainably.

6. Related Research

Statistical anomaly detection has been the subject of extensive research because of its applications to a variety of detection tasks such as network intrusion (Kumar, 2005), credit card fraud (Aleskerov, Freisleben, & Rao, 1997), and malignant tumors from MRIs (Spence, Parra, & Sajda, 2001) among many others (Chandola, Banerjee, & Kumar, 2009). Those works rely on large volumes of data to build statistical models of expected patterns. In that context, anomalies correspond to outlier patterns deviating from expected patterns. There is also the work on execution monitoring (Fritz, 2005; Alcázar et. al., 2010; Langley et. al., 2017; Sapena & Onaindia, 2002) whenever the current plan is not valid or an anomaly is detected the agents performs replanning. However, our work differs from execution monitoring by addressing only some of the anomalies when replanning is not possible. In our work, our models are planning models and anomalies correspond to deviations of those models. One of the most challenging problems of statistical anomaly detection is the potentially large number of false positives, which trigger unnecessary alarms. In our work, in contrast to the previously mentioned works, explanations for an anomaly are generated to determine the nature of the anomaly and decide if the agent must deal with it.

The concept of anomaly detection has played a central role in GDA research. In this work, we are focusing on environmental failures since the anomalies are the result of the partial observability in the environment. Munoz-Avila et al. (2010), observed that not all anomalies require triggering a new goal. In that study, the GDA agent is operating in an adversarial environment with a reward function (i.e., the score of the game). A reward function is also used in Jaidee et al (2011) which uses reinforcement learning techniques to show GDA knowledge. With both of these, when the current plan is resulting in a positive reward rate, the agent will ignore anomalies. In contrast, in our work, we do not assume a reward function; instead, we generate a causal linkage to determine if a problem underlies the anomaly.

ARTUE (Molineaux, Klenk, & Aha, 2010) is a GDA system that was used to provide control in a Naval strategic simulation of an adversarial and partially observable environment. In this work, explanations were generated using a truth maintenance system that identifies plausible worlds that are consistent with the observations made by the agent and triggers a new goal as a result. ARTUE explains all anomalies, whether problematic or not; goal formulation is responsible for determination of whether the agent should respond. The initial version of ARTUE used rule-based knowledge; extended versions incorporated learning of goal selection knowledge (Powell, Molineaux, & Aha, 2011) and domain-independent motivations (Wilson, Molineaux, & Aha, 2013) responsible for identifying situations that require response. However, these techniques modified the goal formulation process, rather than incorporating a separate problem recognition step prior to explanation generation.

Other kinds of explanations also exist; external explanations describe an agent's anomalous behavior to others (Floyd & Aha, 2016), while internal explanations hypothesize the cause of an anomaly for its own needs (Aamodt, 1993; Molineaux, Kuter, & Klenk, 2012). In our work, we use a variant of the internal explanations called problem explanations.

More recently, the notion of GDA agent's expectations has been extended to consider only the necessary effects of the plan executed so far as opposed to considering the whole state (Dannenhauer & Munoz-Avila, 2015).Our work uses this form of expectations.

Our work is motivated by work on introspective reasoning, where the agent reasons about the decisions that lead to actions taken and how these actions affect the environment. Meta-AQUA (Cox & Ram, 1999) reasons about the processes that lead to a decision which resulted in an anomaly and considers three types of anomalies: novel situations, incorrect background knowledge and mix-indexed knowledge structure; the difference between the last two is that in the latter the agent has the knowledge but it is not retrieved in the appropriate circumstances. Fox and Leake (1995) present a mechanism to fix these retrieval mechanisms using introspective reasoning techniques. In our work, we are focusing on novel situations when there is an expectation failure.

7. Conclusion and Future Research

We have described a formalism for agents that enables them to distinguish between those anomalies that they must deal with from those that they do not. The crucial factor in this is the use of *explanation patterns* so that an agent can formulate its own goals to adapt to unexpected events/situations that require the agent's attention.

Real world scenarios often deal with deadlines and it is practically impossible for an agent to worry about all the anomalies it comes across, reason, react and achieve its primary goals within the given deadline. Although our experiment setting is simulated, adding a deadline to our experiment clearly shows that the performance of the GDA agent is better than the eager and the baseline agents.

For future work, we would like to work on several different enhancements that can improve the performance and reasoning capabilities of the GDA agent in our future research. First, adding an importance factor to the problem formalism would help the agent to prioritize anomalies that are classified as a problem with the goals it possesses. Moreover, given that an agent only has finite resources, prioritizing the anomalies could also assist an agent to delegate goals to other agents. Second, adding goal monitors (Dannenhauer & Cox, 2018) after formulating goals could help an agent to adapt as the world changes. For example, during mine clearance, if the establishment of a Q-route changed from one location to another then it is highly likely that it would not need to continue to clear mines along the originally proposed Q-route. In the labor relations domain, the budget could be created through profits or donations, thus changing the world. Finally, if the number of anomalies flagged were excessive given what might be anticipated in a particular context, then this could serve as a cue for the agent to generate a goal with a broader scope than the current goal. For example, our experimental setting has around ten mines within the proposed Q-route. Instead of clearing just the mines on the agent's path from GA1 to GA2, if the number of mines encountered were too great, then the agent could generate, or delegate to another agent, a goal to survey the entire region between GA1 and GA2. Finally, in the labor relations domain, we would like to explore adding other factors such as customer satisfaction, profits, or time and possibly go a step further and create a multiagent scenario.

Acknowledgements

This material is partially based upon work supported by NSF grant #1849131, AFOSR grant #FA2386-17-1-4063, ONR grant #N00014-18-1-2009, and DARPA contract #N6600118C4039.

References

Aamodt, A. (1993). Explanation-driven case-based reasoning. *European Workshop on Case-Based Reasoning* (pp. 274–288). Heidelberg, Berlin: Springer.

Alcázar, V., Guzmán, C., Prior, D., Borrajo, D., Castillo, L., & Onaindia, E. (2010). PELEA: Planning, learning and execution architecture. *PlanSIG'10* (pp. 17-24).

Aleskerov, E., Freisleben, B., & Rao, B. (1997). Cardwatch: A neural network based database mining system for credit card fraud detection. *Proceedings of the IEEE/IAFE 1997 computational intelligence for financial engineering* (pp. 220–226). IEEE.

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. *ACM computing surveys*, *41*(3), 15.

Cox, M. T. (2013). Goal-driven autonomy and question-based problem recognition. In *Second Annual Conference on Advances in Cognitive Systems*, Poster Collection (pp. 29-45). Palo Alto, CA: Cognitive Systems Foundation.

Cox, M. T. (2007). Perpetual self-aware cognitive agents. *AI magazine*, *28*(1), 32.

Cox, M. T., Dannenhauer, D., & Kondrakunta, S. (2017). Goal operations for cognitive systems. *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*.

Cox, M. T., & Ram, A. (1999). Introspective multistrategy learning: On the construction of learning strategies. *Artificial Intelligence*, *112*(1–2), 1–55.

Dannenhauer, D., & Munoz-Avila, H. (2015). Raising expectations in GDA agents acting in dynamic environments. *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*.

Dannenhauer, Z. A., & Cox, M. T. (2018). Rationale-based perceptual monitors. *AI Communications*, (pp. 1–16).

Floyd, M. W., & Aha, D. W. (2016). Incorporating transparency during trust-guided behavior adaptation. *International Conference on Case-Based Reasoning* (pp. 124–138). Springer.

Fox, S., & Leake, D. (1995). Using introspective reasoning to refine indexing. *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence,* (pp. 391–397).

Fritz, C. (2005). Execution monitoring–a survey. *Tech. Rep.*

Ghallab, M., Nau, D., & Traverso, P. (2004). *Automated Planning: theory and practice*. Elsevier.

Jaidee, U., Muñoz-Avila, H., & Aha, D. W. (2011). Integrated learning for goal-driven autonomy. *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*. IJCAI.

Kumar, V. (2005). Parallel and distributed computing for cybersecurity. *IEEE Distributed Systems Online*, 10, 1.

Langley, P., Choi, D., Barley, M., Meadows, B., & Katz, E. P. (2017). Generating, executing, and monitoring plans with goal-based utilities in continuous domains. *Proceedings of the Fifth Annual Conference on Advances in Cognitive Systems*. Cognitive Systems Foundation. Troy, NY.

Li, P. C. (2009). *Planning the optimal transit for a ship through a mapped minefield*. Masters Thesis, Dept of Operations Research, Naval Postgraduate School, Monterey, CA.

Molineaux, M., Klenk, M., & Aha, D. (2010). Goal-driven autonomy in a Navy strategy simulation. *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*.

Molineaux, M., Kuter, U., & Klenk, M. (2012). DiscoverHistory: Understanding the past in planning and execution. *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems Volume 2* (pp. 989–996). International Foundation for Autonomous Agents and Multiagent Systems.

Munoz-Avila, H., Aha, D. W., Jaidee, U., Klenk, M., & Molineaux, M. (2010). Applying goal driven autonomy to a team shooter game. *Proceedings of the Twenty-Third International FLAIRS Conference*.

Powell, J., Molineaux, M., & Aha, D. W. (2011). Active and interactive discovery of goal selection knowledge. *Proceedings of the Twenty-Fourth International FLAIRS Conference*.

Ram, A. (1993). Indexing, elaboration and refinement: Incremental learning of explanatory cases. *Case-Based Learning* (pp. 7–54). Boston, MA: Springer.

Sapena, O., & Onaindia, E. (2002). Execution, monitoring and replanning in dynamic environments. *Working Notes of the AIPS*, *2*.

Schank, R. P. (2013). *Explanation patterns: Understanding mechanically and creatively*. Psychology Press.

Spence, C., Parra, L., & Sajda, P. (2001). Detection, synthesis and compression in mammographic image analysis with a hierarchical image probability model. *Proceedings IEEE Workshop on Mathematical Methods in Biomedical Image Analysis* (pp. 3–10). IEEE.

Wilson, M. A., Molineaux, M., & Aha, D. W. (2013). Domain-independent heuristics for goal formulation. *The Twenty-Sixth International FLAIRS Conference*.