Meta-diagnosis via Preference Relaxation for State Trackability

Xavier Pucel, Stéphanie Roussel, Louise Travé-Massuyès, Valentin Bouziat

Meta-diagnosis via preference relaxation for state trackability

Xavier Pucel[1,3], Stéphanie Roussel[1], Louise Travé-Massuyès[2,3], and Valentin Bouziat[1]

[1] ONERA / DTIS, University of Toulouse, Toulouse, France
firstname.lastname@onera.fr
[2] LAAS-CNRS, University of Toulouse, CNRS, Toulouse, France
louise@laas.fr
[3] ANITI, University of Toulouse, Toulouse, France

Abstract. In autonomous systems, planning and decision making rely on the estimation of the system state across time, *i.e.* state tracking. In this work, a preference model is used to provide non ambiguous estimates at each time point. However, this strategy can lead to dead-ends. Our goal is to anticipate dead-ends at design time and to blame root cause preferences, so that these preferences can be revised. To do so, we present the preference-based state estimation approach and we apply a consistency-based meta-diagnosis strategy based on preference relaxation. We evaluate our approach on a robotic functional architecture benchmark.

1 Introduction

For autonomous systems, state tracking is a critical task because it strongly influences decision making, which is essential to the life of the system. It provides the means to diagnose faults and to react to the various hazards that can affect the system.

In this paper, we focus on discrete event systems [17] in which states are Boolean variable assignments and transitions are propositional logic formulae. When the system state is partially observable, the number of candidate state estimates may quickly become too large to be usable. In embedded or distributed systems, memory and communication limitations may also become a problem, even with symbolic representation techniques such as in [15]. These limitations lead us to propose *a single-state estimation* strategy that retains only one state out of the set of candidate estimates at each time step, as in [2]. Although this can be seen as an extreme strategy, it is efficient to feed the decision system with a clear input and it is consistent with main stream works that select a limited number of best candidates according to some preference criterion, for example probabilities [16, 9]. However, the more we limit the number of estimates, the more we may be confronted with the problem of dead-ends, *i.e.* observations proving that previous estimates were wrong and evidencing no continuation of the estimated trajectory. Because backtracking [9] is not a viable solution in regard to real time constraints, some approaches ([4, 14]) build single-state estimators that are guaranteed to avoid dead-ends, making the system single-state trackable. However, this is not always

possible. In this case, there are two options: to refer to multi-estimator strategies as proposed in [5] or to detect and diagnose dead-ends at design time and to identify which part(s) of the estimator should be modified to circumvent the dead-end.

In this work, we focus on the second option that can be qualified as a *meta-diagnosis* strategy. Following the approach of [11, 12, 2], a single-state estimator is defined by a totally ordered conditional preference model [1], which allows to describe conditions for estimating truth values of state variables. Diagnosing a dead-end means identifying a set of preferences that, if modified appropriately, circumvent the dead-end. The approach implements a consistency-based diagnosis approach based on preference relaxation. This paper resumes the work of a short two-pages paper presented as a poster in [3]. It provides the details of the meta-diagnosis approach and the results of a set of experiments used in the validation phase.

The paper is structured as follows. In Section 2, we formally define the process of incremental single-state estimation and dead-ends. In Section 3, we define the semantic for relaxing conditional preferences and describe a meta-diagnosis strategy to circumvent dead-ends based on relaxing preferences. In Section 4, we present a robotic functional architecture used to test our approach and provide the results in Section 5. We then conclude and discuss future work in Section 6.

2 Incremental Single State Estimation

Preliminary notations. For a variable set $\mathtt{V}$, an assignment over $\mathtt{V}$ is a function that assigns a truth value ($true$, $false$) to each variable of $\mathtt{V}$. An assignment over $\mathtt{V}$ is classically extended to assign a value to formulae whose scope is in $\mathtt{V}$.

For a variable set $\mathtt{V}$ and a variable $\mathtt{v} \in \mathtt{V}$, v (resp. $\overline{v}$) denotes the assignment in which $\mathtt{v}$ is assigned *true* (resp. *false*). For two assignments x and y on the variable sets $\mathtt{X}$ and $\mathtt{Y}$, $x.y$ is the assignment on $\mathtt{X} \cup \mathtt{Y}$ such that $\forall \mathtt{x} \in \mathtt{X}$, $xy(\mathtt{x}) = x(\mathtt{x})$ and $\forall \mathtt{y} \in \mathtt{Y}$, $xy(\mathtt{y}) = y(\mathtt{y})$.

For two sets of variables $\mathtt{X}$ and $\mathtt{Y}$ such that $\mathtt{Y} \subseteq \mathtt{X}$, and an assignment x on $\mathtt{X}$, the projection of x on $\mathtt{Y}$ is denoted $x^{\downarrow \mathtt{Y}}$ such that $\forall \mathtt{y} \in \mathtt{Y}, x^{\downarrow \mathtt{Y}}(\mathtt{y}) = y(\mathtt{y})$.

2.1 System Definition

In the following, we adapt definitions of previous works [12, 2]. We suppose that the system is governed by discrete dynamics where each time step lasts the same duration.

Definition 1 (System). *A system M is a tuple $(\mathtt{O}, \mathtt{E}, \mathcal{S}, s_0, \Delta)$ where:*

- $\mathtt{O}$ *and* $\mathtt{E}$ *are two finite sets of propositional symbols that represent respectively observable and estimated features of the system;*
- $\mathcal{S}$ *is a subset of assignments on* $\mathtt{O} \cup \mathtt{E}$ *and represents the states of the system;*
- $s_0 \in \mathcal{S}$ *is the initial state of the system;*
- $\Delta \subseteq \mathcal{S} \times \mathcal{S}$ *is the set of transitions of the system.*

Without loss of generality, we consider that all the states of the system are reachable from the initial state s_0. In order to avoid an explicit representation of transitions, Δ

is represented by a set of propositional logic formulae Δ_p that can relate to both the variables of $\mathtt{O} \cup \mathtt{E}$, here denoted $\mathtt{P}$ and the same variables but referring to the previous time step, denoted $\mathtt{P}_{\mathtt{pre}}$. Formally, we define a bijection $\mathsf{pre} : \mathtt{P} \to \mathtt{P}_{\mathtt{pre}}$ such that for all $\mathtt{p} \in \mathtt{P}$, $\mathsf{pre}(\mathtt{p})$ refers to the variable $\mathtt{p}$ of the state at the previous time step. For a state $s \in \mathcal{S}$, s^{pre} is the assignment on $\mathtt{P}_{\mathtt{pre}}$ such that $\forall \mathtt{p} \in \mathtt{P}$, $s^{\mathsf{pre}}(\mathsf{pre}(\mathtt{p})) = s(\mathtt{p})$. The set of transitions Δ is the set $\{(s,t) \in \mathcal{S}^2 | s^{\mathsf{pre}}.t(\Delta_p) = true\}$.

An *observation* is a projection of a state on observable symbols $\mathtt{O}$. The set of observations of a system is denoted $\mathcal{O}$.

We define the function $cands : \mathcal{S} \times \mathcal{O} \to 2^{\mathcal{S}}$ such that for all states s in $\mathcal{S}$, for all o in $\mathcal{O}$, $cands(s,o)$ represents the set of successors of s that have observation o. Formally, $\forall s \in \mathcal{S}, \forall o \in \mathcal{O}$, $cands(s,o) = \{t \in \mathcal{S} | (s,t) \in \Delta \text{ and } t^{\downarrow \mathtt{O}} = o\}$.

We also define $nextObs$ the function $\mathcal{S} \to 2^{O}$ such that $nextObs(s)$ is the set of observations that can be observed just after the system is in state s. Formally, $\forall s \in S, nextObs(s) = \{o \in \mathcal{O} | cands(s,o) \neq \emptyset\}$.

A state sequence seq is a list $(s_0, s_1, \ldots, s_{n-1})$ where each s_i is a state in $\mathcal{S}$; $|seq| = n$ is the length of the sequence and $seq[i] = s_i$ is the ith state in the sequence; $last(seq)$ designates the last state of seq; if s is a state, $seq \cdot s$ is the sequence of length $|seq| + 1$ that begins with seq and ends with s.

Definition 2 (Language, observation language). *The language associated with a system $M = (\mathtt{O}, \mathtt{E}, \mathcal{S}, s_0, \Delta)$ is the set of state sequences accepted by the system and starting with s_0. Formally $\mathcal{L}(M) = \{seq \in \mathcal{S}^+ | seq[0] = s_0 \text{ and } \forall i \in [1, |seq| - 1], (seq[i-1], seq[i]) \in \Delta\}$. The observation language is the language accepted by the system projected on the observations. Formally, $\mathcal{L}_{obs}(M) = \{seq^{\downarrow \mathtt{O}} | seq \in \mathcal{L}(M)\}$.*

We illustrate this modelling approach on a small model that is a simplified version of the experimental benchmark of Section 4.

Example 1 (System). We consider a simple robot functional architecture with three functions: movement, communication and power supply. The health status of each function is represented by the three respective variables $\mathtt{h}_{\mathtt{mv}}$, $\mathtt{h}_{\mathtt{com}}$ and $\mathtt{h}_{\mathtt{pow}}$. Two alarms $\mathtt{al}_{\mathtt{mv}}$ and $\mathtt{al}_{\mathtt{com}}$ can be raised when movement and communication fail respectively. We estimate the value of health variables from the sequence of alarms, *i.e.* $\mathtt{O} = \{\mathtt{al}_{\mathtt{mv}}, \mathtt{al}_{\mathtt{com}}\}$, and $\mathtt{E} = \{\mathtt{h}_{\mathtt{mv}}, \mathtt{h}_{\mathtt{com}}, \mathtt{h}_{\mathtt{pow}}\}$. The initial state is $s_0 = \overline{al_{mv}}.\overline{al_{com}}.h_{mv}.h_{com}.h_{pow}$.

Δ is the conjunction of the following formulae :

$$\neg \mathsf{pre}(\mathtt{h}_{\mathtt{pow}}) \to \neg \mathtt{h}_{\mathtt{pow}} \quad (\delta_1) \qquad \neg \mathtt{h}_{\mathtt{pow}} \to (\mathtt{al}_{\mathtt{mv}} \wedge \mathtt{al}_{\mathtt{com}}) \quad (\delta_2)$$

$$\neg \mathtt{h}_{\mathtt{mv}} \to \mathtt{al}_{\mathtt{mv}} \quad (\delta_3) \qquad \neg \mathtt{h}_{\mathtt{com}} \to \mathtt{al}_{\mathtt{com}} \quad (\delta_4)$$

It expresses that the fault in the power supply is permanent (δ_1), and causes both alarms to be raised (δ_2), movement faults cause movement alarms (δ_3) and communication faults cause communication alarms (δ_4). Note that alarms can also occur without any fault being present (false positives, external perturbations, *etc.*), and that movement and communication faults are intermittent, *i.e.* they are independent of their previous values.

At time step 1, let us assume we receive observation $o_1 = al_{mv}.\overline{al_{com}}$. We have $cands(s_0, o_1) = \{al_{mv}.\overline{al_{com}}.h_{mv}.h_{com}.h_{pow}, al_{mv}.\overline{al_{com}}.\overline{h_{mv}}.h_{com}.h_{pow}\}$.

In the first candidate state, the al_{mv} alarm is explained as a false alarm or caused by some unknown event, while it is explained by a fault in the move module in the second candidate.

2.2 Preference-based estimation strategy

We now focus on the estimation part for systems defined above. Since we consider non-deterministic, partially observable systems, there may be several state sequences that explain a given observation sequence. We adopt an incremental approach to select a unique explanation, called an *estimation strategy* [14].

Definition 3 (Estimation strategy). *An incremental single-state estimation strategy for a system M is a function $estim : \mathcal{S} \times \mathcal{O} \rightarrow \mathcal{S}$ such that for all s in $\mathcal{S}$, for all o in $nextObs(s)$, $estim(s, o)$ represents the estimated state of the system at time step k if it was estimated in state s at time step $k-1$ and if o is observed at time step k. We impose the estimation strategy to be consistent both across time (i.e. $estim$ is a function) and with the system behaviour (i.e. $estim(s, o)$ belongs to $cands(s, o)$).*

Following [12, 2], the estimation strategy is based on a sequence of conditional preferences.

Definition 4 (Conditional preference). *Let M be a system. A conditional preference γ on a variable* $\mathtt{e}$ *of* $\mathtt{E}$ *is defined by $\langle \mathtt{cond} : e \prec \overline{e} \rangle$, where* $\mathtt{cond}$ *is a propositional formula on* $\mathtt{P} \cup \mathtt{P_{pre}}$. $\mathtt{cond}$ *is called γ's condition and* $\mathtt{e}$ *is γ's target.*

Informally, $\langle \mathtt{cond} : e \prec \overline{e} \rangle$ expresses the fact that we prefer to estimate $\mathtt{e}$ as true if and only if $\mathtt{cond}$ is true. Note that it is equivalent to $\langle \neg \mathtt{cond} : \overline{e} \prec e \rangle$.

From now on, we consider that the set of estimated variables $\mathtt{E}$ contains n variables and that these variables are ordered, from $\mathtt{e_1}$ to $\mathtt{e_n}$. This order can be seen as the order used to estimate the state at every time step.

Definition 5 (Conditional preference model). *A conditional preference model Γ for a system M is an ordered sequence of preferences $(\gamma_1, \gamma_2, \ldots, \gamma_n)$ such that $\gamma_i = \langle \mathtt{cond_i} : e_i \prec \overline{e_i} \rangle$ is a conditional preference on $\mathtt{e_i}$ and that its condition $\mathtt{cond_i}$ only uses variables from $\mathtt{P_{pre}} \cup \mathtt{O} \cup \{\mathtt{e_j} | 1 \leq j < i\}$.*

Informally, if γ_i appears before γ_j in Γ, then the condition of γ_j can depend on the outcome of γ_i, but the reverse is forbidden.

A conditional preference model Γ allows to define an estimation strategy $estim$ as presented in Algorithm 1. Let $s \in \mathcal{S}$ be the state of the system at time step $k-1$ and $o \in nextObs(s)$ the observation received at time step k. We initialise the sets of preferred candidates for $estim(s_{k-1}, o_k)$ with $cands(s_{k-1}, o_k)$ (line 1) and the general idea is to remove non-preferred candidates from this set until it is a singleton. To do so, for each preference γ_i, we compute the value val of $\mathtt{cond_i}$ with the assignment $s_{k-1}{}^{\mathsf{pre}}.o_k.estTargets$ where $estTargets$ is the assignment containing truth values for variables $\mathtt{e_j}$ with $j < i$ (line 3). If there exists a state t in $preferredCands$ such that $t(\mathtt{e_i}) = val$, we apply the preference and consider that $estim(s_{k-1}, ok) = val$ (line 5)

Algorithm 1: $\texttt{PreferredEstimation}(M, s_{k-1}, o_k)$
Returns a singleton containing the preferred state $estim(s_{k-1}, o_k)$

1 $preferredCands \leftarrow cands(s_{k-1}, o_k)$; $estTargets \leftarrow true$
2 **for** $i \leftarrow 1 : n$ **do**
3 $\quad val \leftarrow s_{k-1}{}^{\mathsf{pre}}.o_k.estTargets(\mathtt{cond_i})$
4 $\quad$ **if** $\exists t \in preferredCands$ *such that* $t(\mathtt{e_i}) = val$ **then**
5 $\quad\quad estTargets \leftarrow estTargets.[\mathtt{e_i} \leftarrow val]$
6 $\quad\quad preferredCands \leftarrow preferredCands - \{t | t(\mathtt{e_i}) \neq val\}$
7 $\quad$ **else** $estTargets \leftarrow estTargets.[\mathtt{e_i} \leftarrow \overline{val}]$
8 **return** $preferredCands$

and we remove from $preferredCands$ all states that do not have value val for $\mathtt{e_i}$ (line 6). If such a t does not exist, it means there is no choice for this preference, *i.e.* only the negation of val is possible for $\mathtt{e_i}$ in preferred states (line 7). Note that such an algorithm corresponds to the best transition with respect to a partial order on transitions formally defined in [2].

Example 2 (Preference model). We consider the system of Example 1 and now define the conditional preference model:

$$\langle \neg \mathsf{pre}(\mathtt{al_{mv}}) \wedge \mathtt{al_{mv}} \wedge \neg \mathsf{pre}(\mathtt{al_{com}}) \wedge \mathtt{al_{com}} : \overline{h_{pow}} \prec h_{pow} \rangle \qquad (\gamma_1)$$
$$\langle \mathtt{al_{mv}} \wedge \mathtt{h_{pow}} : \overline{h_{mv}} \prec h_{mv} \rangle \qquad (\gamma_2)$$
$$\langle \mathsf{pre}(\mathtt{al_{com}}) \wedge \mathtt{al_{com}} \wedge \mathtt{h_{pow}} : \overline{h_{com}} \prec h_{com} \rangle \qquad (\gamma_3)$$

If both alarms are raised simultaneously, we blame their common cause, i.e. the power supply (γ_1). Otherwise we blame the respective functions (γ_2 and γ_3). Moreover, for the communication function, we dismiss the first alarm as noise, and only diagnose a communication fault when the alarm persists during several time step (γ_3). Thus, we have $estim(s_0, o_1) = al_{mv}.\overline{al_{com}}.\overline{h_{mv}}.h_{com}.h_{pow}$.

Definition 6 (Estimated sequence). *Let M be a system, seq_{obs} be an observation sequence in $\mathcal{L}_{obs}(M)$ and $estim$ an estimation strategy for M based on a preference model Γ. The estimated sequence for seq_{obs} is the state sequence $\widehat{seq} \in \mathcal{L}(M)$ such that $\widehat{seq}[0] = s_0$ and for all i in $[1, |\widehat{seq}| - 1]$, $\widehat{seq}[i] = estim(\widehat{seq}[i-1], seq_{obs}[i])$.*

2.3 Dead-end

At some point, the estimator may choose a state sequence different from the one actually taken by the system. This may be an issue when the following conditions happen: a) the system is in state s, the estimator estimates that it is in state $\hat{s}$ with $\hat{s} \neq s$; b) the system moves to state t and produces observation $t^{\downarrow \mathtt{o}}$; c) the set of candidates $cands(\hat{s}, t^{\downarrow \mathtt{o}})$ is empty.

Definition 7 (Dead-end). *Let M be a system and $estim$ an estimation strategy based on a preference model Γ. A sequence of observations $seq_{obs} \cdot o$ in $\mathcal{L}_{obs}(M)$ is a* dead-end *if there exists an estimated sequence $\widehat{seq}$ for seq_{obs} and $cands(last(\widehat{seq}), o) = \emptyset$.*

Example 3 (Dead-end). In the system and the estimation strategy presented in Examples 1 and 2, the sequence of observations $(\overline{al_{mv}}.\overline{al_{com}}, al_{mv}.al_{com}, \overline{al_{mv}}.\overline{al_{com}})$ is a dead-end. In fact, at time step 1, two faults occur simultaneously in the movement and communication functions in the system, causing both alarms to activate. The estimator explains it with a fault in the battery, due to preference (γ_1), which constitutes a *divergence* between the real system and the diagnosis. At time step 2, the faults disappear in the real system, causing both alarms to stop and the estimator cannot explain this observation as the fault in the battery is permanent.

3 Meta-diagnosis via consistency-based diagnosis of preferences

While dead-ends can be eliminated by modifying the system, we aim at eliminating them by modifying the estimation strategy defined by the the preference model. Indeed, some dead-ends can be circumvented by modifying the preference conditions in Γ. We address this problem with a consistency-based diagnosis approach [7]. More precisely, given a dead-end and a preference model, we want to know whether the observation sequence associated with the dead-end could be accepted by the estimator if some preferences that we aim to identify were "relaxed". When this is the case, we can indicate to the designer that the dead-end can be avoided by modifying the conditions for the identified subset of preferences. Correcting the preference conditions is left for future work.

3.1 Relaxed preference model

In this subsection, we generalise the notion of preference by introducing *relaxed preferences*. Then, we define the notion of (general) *preference*, allowing us to define a *relaxed preference model*. A *relaxed preference* targeting the variable $\mathtt{e}$ in $\mathtt{E}$ declares that the valuations of $\mathtt{e}$ are incomparable, i.e. there is no preference on the valuations of $\mathtt{e}$ in any context.

Definition 8 (Relaxed preference model). *A relaxed preference for* $\mathtt{e}$ *in* $\mathtt{E}$ *has the form* $\langle e \nsim \overline{e} \rangle$. *A (general) preference for* $\mathtt{e} \in \mathtt{E}$*, denoted* φ*, is either a conditional preference* $\langle \mathtt{cond} : e \prec \overline{e} \rangle$ *or a relaxed preference* $\langle e \nsim \overline{e} \rangle$. *Given a system* M*, a conditional preference model* $\Gamma = (\gamma_1, \gamma_2, \ldots, \gamma_n)$ *and a subset of preferences* $\Omega \subseteq \Gamma$*, a* relaxed preference model Γ_Ω *is a sequence of preferences* $(\varphi_1, \varphi_2, \ldots, \varphi_n)$ *such that* $\varphi_i = \gamma_i$ *if* $\gamma_i \notin \Omega$*, and* $\varphi_i = \langle e_i \nsim \overline{e_i} \rangle$ *otherwise.*

In comparison with the conditional preference model presented previously, a relaxed preference model may result in more than one preferred state in the set of candidates at each time step. Preferred states only differ with respect to variables that are target of relaxed preferences. In Algorithm 1, two lines should be modified to compute preferred states in that case. First, at line 2, the condition of the loop should be "for all $i \in [1, n]$ such that ϕ_i is a conditional preference". This implies that candidates states can only be removed by conditional preferences. Then, as variables $\mathtt{e_j}$ that are target of relaxed preferences are not assigned a truth value in $estTargets$, line 3 should be replaced by "$val \leftarrow \mathbf{isSAT}(s_{k-1}{}^{\mathsf{pre}}.o_k.estTargets, \mathtt{cond_i})$", where for an assignment x over

variables X, and form a formula whose scope is F, $\mathbf{isSAT}(x, \texttt{form})$ returns $true$ if and only there exists an assignment y over variables $\texttt{X} \cup \texttt{F}$ such that $y^{\downarrow \texttt{X}} = x$ and $y(\texttt{form})$ is true. Therefore, val is true if and only if it is possible to assign targets of relaxed preferences to make condition $\texttt{cond}_\texttt{i}$ true. Finally, the returned preferred candidates may be several as some steps are skipped in the loop.

Note that, following [2], it would be possible to formally define the new order on transitions defined by a relaxed preference model but we do not present it here for conciseness purposes.

3.2 Relaxed estimation process

We now generalise the notions of estimation strategy, estimation sequence and dead-end. Informally, a *relaxed estimation strategy* returns a set of preferred candidates instead of a unique preferred candidate. A *relaxed estimated sequence* is a sequence of states in which any successive states s and t in the sequence are such that t belongs to the set of preferred states of the relaxed estimation strategy for state s. A *relaxed dead-end* is a sequence of observations for which the last observation cannot be explained by any relaxed estimation sequence.

Definition 9 (Relaxed estimation strategy). *A* relaxed estimation strategy *for a system M and a relaxed preference model Γ_Ω is a function $estim_\Gamma^\Omega : \mathcal{S} \times \mathcal{O} \rightarrow 2^\mathcal{S}$ such that for all s in $\mathcal{S}$, for all o in $nextObs(s)$, $estim_\Gamma^\Omega(s, o)$ represents the set of preferred estimated states of the system at time step k if it was estimated in state s at time step $k-1$ and if o is observed at time step k.*

Definition 10 (Relaxed estimation sequence). *Let M be a system, seq_{obs} be an observation sequence in $\mathcal{L}_{obs}(M)$ and $estim_\Gamma^\Omega$ a relaxed estimation strategy for M. A* relaxed estimation sequence *for seq_{obs} is a state sequence $\widehat{seq} \in \mathcal{L}(M)$ such that $\widehat{seq}[0] = s_0$ and for all i in $[1, |\widehat{seq}| - 1]$, $\widehat{seq}[i] \in estim_\Gamma^\Omega(\widehat{seq}[i-1], seq_{obs}[i])$.*

Definition 11 (Relaxed dead-end). *Let M be a system and $estim_\Gamma^\Omega$ a relaxed estimation strategy. A* relaxed dead-end *is a sequence of observations $seq_{obs} \cdot o$ in $\mathcal{L}_{obs}(M)$ such that for all relaxed estimated sequences $\widehat{seq}$ for seq_{obs}, $cands(last(\widehat{seq}), o) = \emptyset$.*

Intuitively, relaxing preferences allows to increase the set of estimated sequences and therefore might reduce the number of relaxed dead-ends. This is expressed through the following proposition.

Proposition 1 (Relaxed dead-end inclusion). *Let M be a system and $estim_\Gamma^{\Omega_1}$ and $estim_\Gamma^{\Omega_2}$ two relaxed estimation strategies such that $\Omega_1 \subseteq \Omega_2$. If seq_{obs} is a relaxed dead-end for $estim_\Gamma^{\Omega_2}$ then it is also a relaxed dead-end for $estim_\Gamma^{\Omega_1}$.*

Proof. We first show that for a state s in $\mathcal{S}$ and an observation o in $nextObs(s)$, $estim_\Gamma^{\Omega_1}(s, o) \subseteq estim_\Gamma^{\Omega_2}(s, o)$, *i.e.* preferred states are still preferred after relaxing preferences. Then, we can show that a relaxed estimation sequence for seq_{obs} in $estim_\Gamma^{\Omega_1}$ is also a relaxed estimation sequence for seq_{obs} in $estim_\Gamma^{\Omega_2}$. It follows that if all relaxed estimation sequences for seq_{obs} in $estim_\Gamma^{\Omega_2}$ cannot be followed by o, so it is for all relaxed estimation sequences for seq_{obs} in $estim_\Gamma^{\Omega_1}$.

Example 4. Let us consider the dead-end from Example 3: $seq_{obs} = (\overline{al_{mv}}.\overline{al_{com}}, al_{mv}.al_{com}, \overline{al_{mv}}.\overline{al_{com}})$, and let us relax two preferences $\Omega = \{\gamma_1, \gamma_2\}$. TOTOTO TOTOTOTO TOTOTOTO The sequence of states $(s_0, \widehat{s_1}, \widehat{s_2})$ with $\widehat{s_1} = al_{mv}.al_{com}.h_{pow}.\overline{h_{mv}}.h_{com}$ and $\widehat{s_2} = \overline{al_{mv}}.\overline{al_{com}}.h_{pow}.h_{mv}.h_{com}$ is a relaxed estimation sequence for seq_{obs}, which means that seq_{obs} is not a relaxed dead-end for $estim^{\Omega}_{\Gamma}$.

3.3 Consistency-based preference diagnosis

Checking whether a given observation sequence is a relaxed dead-end can be considered as a form of consistency check (due to Proposition 1). Then, searching for the smallest set(s) of preferences that circumvent a dead-end can be done with a classical consistency-based diagnosis algorithm [13].

Definition 12 (Preference Meta-Diagnosis). *Let M be a system, $estim$ an estimation strategy based on a conditional preference model Γ, and seq_{obs} a dead-end for this estimation strategy. A set of preferences $\Omega \subseteq \Gamma$ is a* preference meta-diagnosis *if seq_{obs} is not a relaxed dead-end for $estim^{\Omega}_{\Gamma}$.*

A meta-diagnosis Ω is a minimal meta-diagnosis *if and only if there is no meta-diagnosis $\Omega' \subseteq \Gamma$ such that $\Omega' \subset \Omega$.*

A meta-diagnosis Ω is interpreted as follows: it is possible to modify the conditions for the preferences in Ω so that the diagnoser does not dead-end on the associated observation sequence. It does not guarantee anything with respect to other potential dead-ends. Proposition 1 ensures that if Ω is a meta diagnosis, then all supersets of Ω are meta-diagnoses as well.

Example 5 (Minimal meta-diagnosis). In example 4, we have seen that seq_{obs} was not a relaxed dead-end for $estim^{\Omega}_{\Gamma}$. This means that $\Omega = \{\gamma_1, \gamma_2\}$ is a meta-diagnosis for this dead-end. The minimal meta-diagnosis for seq_{obs} is in fact $\Omega_2 = \{\gamma_1\}$. Thus, modifying the condition of γ_1, can eliminate dead-end seq_{obs}, for example by replacing γ_1 with $\gamma_1' = \langle \top : h_{pow} \prec \overline{h_{pow}} \rangle$. However it implements a different fault management strategy, that must be validated against the robotic mission requirements.

To check if a sequence of observations seq_{obs} is a relaxed dead-end for a given set of relaxed preferences, it is possible to compute at each time step the set of preferred candidates. To do so, we follow the modification of Algorithm 1 described previously in this section. Then, starting from the initial state, it is possible to compute all relaxed estimation sequences and therefore compute whether seq_{obs} is a relaxed dead-end. Note that this approach is combinatorial in both the number of relaxed preferences and the length of seq_{obs}.

By testing the meta-diagnosis candidates, one can find all the minimal preferences meta-diagnoses. Approaches such as the FASTDIAG algorithm [6] can be used to efficiently browse the meta-diagnosis candidate space.

4 Experiments

We have experimented our approach on a functional robotic architecture along with a complex dynamic and complex preference model. We consider a system with three

functions: movement, communication and power supply. It can raise two alarms $\mathtt{al}_{\mathtt{mv}}$ and $\mathtt{al}_{\mathtt{com}}$ (observations of the system) if they are performing poorly. We model the trust we have in each function with variables $\mathtt{t}_{\mathtt{mv}}$, $\mathtt{t}_{\mathtt{com}}$ and $\mathtt{t}_{\mathtt{pow}}$, in the sense that as we receive alarms, we lose trust in the system's operational capacity. Variables $\mathtt{f}_{\mathtt{mv}}$ and $\mathtt{f}_{\mathtt{com}}$ model external perturbations that impede movement and communication (obstacles, slippery terrain, distance to antenna, etc). $\mathtt{f}_{\mathtt{pow}}$ represents a loss of voltage in the power supply.

The estimator receives alarms as input, and estimates if each function can be trusted for autonomous operation. Figure 1 details the model and illustrates this architecture.

Δ represents that we trust the movement and communication functions only if we trust the power supply function as well (δ_1). Movement (resp. communication) perturbations or low voltage cause a movement (resp. communication) alarm (δ_2, δ_3). Once we have lost trust in the power supply, we never trust it again (δ_4). For communication, the alarm is a perfect indicator of our trust in the function (δ_5).

$$(\mathtt{t}_{\mathtt{mv}} \vee \mathtt{t}_{\mathtt{com}}) \rightarrow \mathtt{t}_{\mathtt{pow}} \quad (\delta_1)$$
$$(\mathtt{f}_{\mathtt{mv}} \vee \mathtt{f}_{\mathtt{pow}}) \rightarrow \mathtt{al}_{\mathtt{mv}} \quad (\delta_2)$$
$$(\mathtt{f}_{\mathtt{com}} \vee \mathtt{f}_{\mathtt{pow}}) \rightarrow \mathtt{al}_{\mathtt{com}} \quad (\delta_3)$$
$$\neg \mathsf{pre}(\mathtt{t}_{\mathtt{pow}}) \rightarrow \neg \mathtt{t}_{\mathtt{pow}} \quad (\delta_4)$$
$$\mathtt{t}_{\mathtt{com}} \leftrightarrow \neg \mathtt{al}_{\mathtt{com}} \quad (\delta_5)$$

$$\mathsf{pre}(\mathtt{f}_{\mathtt{pow}}) \vee (\mathbf{up}(\mathtt{al}_{\mathtt{mv}}) \wedge \mathbf{up}(\mathtt{al}_{\mathtt{com}})) : f_{pow} \prec \overline{f_{pow}} \quad (\gamma_1)$$
$$\mathbf{O}(\mathbf{H}_{\kappa}(\mathtt{f}_{\mathtt{pow}})) : \overline{t_{pow}} \prec t_{pow} \quad (\gamma_2)$$
$$\mathbf{H_2}(\mathtt{al}_{\mathtt{mv}}) \wedge \neg \mathtt{f}_{\mathtt{pow}} : f_{mv} \prec \overline{f_{mv}} \quad (\gamma_3)$$
$$\mathbf{H_4}(\neg \mathtt{al}_{\mathtt{mv}}) : t_{mv} \prec \overline{t_{mv}} \quad (\gamma_4)$$
$$\mathtt{al}_{\mathtt{com}} \wedge \neg \mathtt{f}_{\mathtt{pow}} : f_{com} \prec \overline{f_{com}} \quad (\gamma_5)$$
$$\top : t_{com} \prec \overline{t_{com}} \quad (\gamma_6)$$

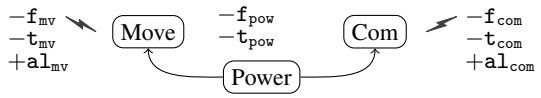

Fig. 1: Δ, Γ and schema of the simple architecture model. Variables labelled with $+$ are observable, $-$ estimated. Arrows represent functional dependency.

In the preferences associated with this model, we use the temporal logic operators from PtLTL to express formula compactly. We rely on [8] to efficiently translate PtLTL formulae into propositional logic formulae. The formula $\mathbf{up}(\mathtt{f})$ is true when the formula $\mathtt{f}$ was false at the previous time step and is now true. The formula $\mathbf{H}_{\kappa}(\mathtt{f})$ (with $\kappa > 0$) is true when the formula $\mathtt{f}$ has been true for the last κ time steps, including now. The formula $\mathbf{O}(\mathtt{f})$ is true when $\mathtt{f}$ has been true at least once in the experiment, including now.

Γ is the ordered sequence $(\gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5, \gamma_6)$ and implements the following strategy. We blame low voltage if and only if both alarms fire simultaneously (γ_1), or if low voltage was already blamed at the previous time step. After a continuous period of size κ with low voltage, we lose trust in the power supply[1] (γ_2). When a movement

[1] Given constraint (δ_4), preference (γ_2) could equivalently be written as $\mathbf{H}_{\kappa}(\mathtt{f}_{\mathtt{pow}})) : \overline{t_{pow}} \prec t_{pow}$. However, we claim that Γ should represent the estimation strategy independently from the system model, for modularity purposes.

alarm not explained by low voltage is on for two time steps, we blame the associated environmental perturbations (γ_3). After 4 time steps without alarm we trust the movement function (γ_4). Communication alarms not explained by low voltage are blamed on environmental perturbations (γ_5). In doubt, we trust the communication function (γ_6).

This model has a dead-end when both movement and communication alarms are raised simultaneously, stay on for κ time steps, then $\texttt{al}_{\texttt{com}}$ turns off. We can control the minimal length of the dead-ends with the κ parameter in (γ_2), as the shortest dead-end has $\kappa+2$ time steps. There are two minimal meta-diagnoses for this dead-end: $\{\gamma_1\}$ and $\{\gamma_2\}$. If we replace γ_1 by $\gamma_1' = \langle \top : \overline{f_{pow}} \prec f_{pow} \rangle$ or γ_2 by $\gamma_2' = \langle \top : t_{pow} \prec \overline{t_{pow}} \rangle$, the dead-end is circumvented, although it changes the estimation strategy.

5 Results

We use Sat4j [10] as a SAT solver to directly compute if there exists a candidate with a particular variable value (see Algorithm 1 line 5), and to implement the $\mathbf{isSAT}$ function described in Section 3.1. Experiments have been conducted on an Intel(R) Xeon(R) CPU E5-2660 v3 @ 2.60GHz processor with 62GiB of RAM, although only around 2Gib were used.

The results depicted in Table 1 show that the computation is very fast for short dead-ends, but grows exponentially with dead-end length. This is due to the fact that at each time step, for each relaxed preference, there may be two outcomes, which means that in the worst case we need to explore an exponential number of paths to check if a path is a relaxed dead-end.

We consider the performance satisfactory for two reasons. First, memory usage is not a limiting factor, and time is not a constraint during design. Second, long dead-ends are difficult to find, but also difficult to interpret and debug by the designer. When an estimation model becomes too large for maintenance, architectural responses may help dividing it in smaller decentralized models.

κ	Dead-end len.	Comp. time (s)
3	5	0.56
4	6	0.84
5	7	1.52
6	8	3.03
7	9	6.07

κ	Dead-end len.	Comp. time (s)
8	10	11.57
9	11	23.48
10	12	47.72
11	13	94.26
12	14	186.93

Table 1: Meta-diagnosis computation time (in seconds) against dead-end length.

6 Conclusion

In this paper, we present an approach for blaming a dead-end on a set of preferences at design time. It follows a consistency-based meta-diagnosis strategy based on relaxing

conditional preferences. We have defined and implemented algorithms with satisfactory performances. For large benchmarks, several approaches can be explored to improve the associated computation time. For instance, we could parallelize the algorithms by dividing the space search among several computation cores. We could also define an intelligent heuristic for finding relevant scenarios faster.

During our experiments we noted that many dead-ends reproduce the same pattern. A perspective is to identify dead-end patterns to represent them more compactly. Another perspective is to find relaxations that circumvent several dead-ends at once. A final perspective is the correction of conditions of preferences belonging to a meta-diagnosis to avoid a dead-end or a set of dead-ends.

Acknowledgements

This project has been supported by ANITI, the "Artificial and Natural Intelligence Toulouse Institute", through the French "Investing for the Future – PIA3" program under the Grant agreement ANR- 19-PI3A-0004.

References

1. Boutilier, C., Brafman, R.I., Domshlak, C., Hoos, H.H., Poole, D.: Preference-based constrained optimization with CP-nets. In: Computational Intelligence, pp. 137–157 (2004)
2. Bouziat, V., Pucel, X., Roussel, S., Travé-Massuyès, L.: Preferential discrete model-based diagnosis for intermittent and permanent faults. In: Proceedings of the 29th International Workshop on Principles of Diagnosis (DX'18) (2018)
3. Bouziat, V., Pucel, X., Roussel, S., Travé-Massuyès, L.: Preference-based fault estimation in autonomous robots: Incompleteness and meta-diagnosis. In: E. Elkind, M. Veloso, N. Agmon, M.E. Taylor (eds.) Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '19, Montreal, QC, Canada, May 13-17, 2019, pp. 1841–1843. International Foundation for Autonomous Agents and Multiagent Systems (2019). URL http://dl.acm.org/citation.cfm?id=3331937
4. Bouziat, V., Pucel, X., Roussel, S., Travé-Massuyès, L.: Single state trackability of discrete event systems. In: Proceedings of the 30th International Workshop on Principles of Diagnosis (DX'19) (2019)
5. Coquand, C., Pucel, X., Roussel, S., Travé-Massuyès, L.: Dead-end free single state multi-estimators for DES -the 2-estimator case. In: 31st International Workshop on Principles of Diagnosis (DX-2020). Nashville, Tennessee, United States (2020). URL https://hal.laas.fr/hal-03089427
6. Felfernig, A., Schubert, M., Zehentner, C.: An efficient diagnosis algorithm for inconsistent constraint sets. Artificial Intelligence for Engineering Design, Analysis and Manufacturing: AI EDAM **26**(1), 53 (2012)
7. Hamscher, W., et al.: Readings in model-based diagnosis (1992)
8. Havelund, K., Rosu, G.: Synthesizing monitors for safety properties. In: Proc. of the 8th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS-02), pp. 342–356 (2002)
9. Kurien, J., Nayak, P.P.: Back to the future for consistency-based trajectory tracking. In: AAAI/IAAI, pp. 370–377 (2000)

10. Le Berre, D., Parrain, A.: The Sat4j library, release 2.2. JSAT **7**(2-3), 59–6 (2010). URL https://satassociation.org/jsat/index.php/jsat/article/view/82
11. Pralet, C., Pucel, X., Roussel, S.: Diagnosis of intermittent faults with conditional preferences. In: Proceedings of the 27th International Workshop on Principles of Diagnosis (DX'16) (2016)
12. Pucel, X., Roussel, S.: Intermittent fault diagnosis as discrete signal estimation: Trackability analysis. In: 28th International Workshop on Principles of Diagnosis (DX'17) (2017)
13. Reiter, R.: A theory of diagnosis from first principles. Artificial intelligence **32**(1), 57–95 (1987)
14. Roussel, S., Pucel, X., Bouziat, V., Travé-Massuyès, L.: Model-based synthesis of incremental and correct estimators for discrete event systems. In: C. Bessiere (ed.) Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020, pp. 1884–1890. ijcai.org (2020). DOI 10.24963/ijcai.2020/261. URL https://doi.org/10.24963/ijcai.2020/261
15. Torta, G., Torasso, P.: An on-line approach to the computation and presentation of preferred diagnoses for dynamic systems. AI Communications **20**(2), 93–116 (2007)
16. Williams, B.C., Nayak, P.P.: A model-based approach to reactive self-configuring systems. In: Proceedings of the 13th AAAI Conference on Artificial Intelligence, pp. 971–978 (1996)
17. Zaytoon, J., Lafortune, S.: Overview of fault diagnosis methods for discrete event systems. Annual Reviews in Control **37**(2), 308–320 (2013)