EVALUATION OF ANOMALY AND FAILURE SCENARIOS INVOLVING AN EXPLORATION ROVER: A BAYESIAN NETWORK APPROACH.

Daniele Codetta-Raiteri[1], **Luigi Portinale**[1], **Andrea Guiotto**[2], **and Yuri Yushtein**[3]

[1]*DiSIT, Computer Science Institute, University of Piemonte Orientale, 15121 Alessandria, Italy*
[2]*Business Unit Optical Observation & Science Engineering Software, Thales Alenia Space, 10146 Torino, Italy*
[3]*Systems, Software & Technology Department, ESA-ESTEC, 2200 Noordwijk, The Netherlands*

ABSTRACT

Recent studies focused on the achievement of autonomy by spacecrafts, with the aim of avoiding the intervention of the ground control. In this sense, the ARPHA software prototype has been developed for the automatic failure detection, identification and recovery (FDIR), and is based on the on-board analysis of a Dynamic Bayesian Network (DBN) representing the system behaviour conditioned by the conditions of components and environment. In this paper, we describe the main functionalities of ARPHA, and we apply its FDIR capabilities to the power supply subsystem of an exploring rover, taking into account four scenarios leading to anomalies or failures. The DBN model of the system is described. Then, we test the execution of ARPHA, together with a rover simulator providing sensor data and plan data. In particular, we show the results of diagnosis, prognosis and recovery, returned by ARPHA when the scenarios occur.

Key words: Failure Detection, Identification and Recovery; autonomous systems; dynamic Bayesian networks.

1. INTRODUCTION

Recent studies tried to obtain the autonomy by spacecrafts, in order to avoid the involvement of the ground control to solve malfunctioning issues. Several available techniques for system diagnosis and prognosis have been considered and evaluated for the application in the space exploration domain [1, 2, 3]. The traditional approach for on-board FDIR (*Fault Detection, Identification and Recovery*) is based on the run-time observation of the system operational status (health monitoring) in order to detect faults, while the initiation of the corresponding recovery actions uses static pre-compiled look-up tables. In this paper, we present ARPHA (*Anomaly Resolution and Prognostic Health management for Autonomy*), a software prototype for automatic FDIR, based on the on-board analysis (inference) of a *Dynamic Bayesian Networks* (DBN) [4] representing the system behaviour. *Bayesian Networks* (BN) [5] are a typical probabilistic graphical model allowing the computation of diagnostic and predictive measures, possibly conditioned by the partial observation of the system. DBN is a particular form of BN and introduces a discrete temporal dimension (Sec. 2). The system can be initially modelled as a *Dynamic Fault Tree* (DFT) [6] which is a model familiar to reliability engineers. The DFT can be automatically transformed in DBN [7] which can be completed by the user adding the system features that are not captured by the DFT model. The enriched DBN (in *Junction Tree* (JT) [8] form) becomes the on-board model that ARPHA exploits for FDIR conditioned by observations coming from the system sensors and concerning the components or environment conditions (Sec. 3). In this paper, we apply ARPHA to an exploring rover in specific scenarios possibly leading to anomalies or failures. In particular, we take into account the power supply subsystem, with a specific attention to the power generation by solar arrays, the load due to the current action performed by the rover, and the variations to the battery charge due to the difference between the amounts of power generation and load. ARPHA is executed together with a rover simulator generating the observations from the sensors. ARPHA estimates, at each time step, the current system state (Diagnosis) and the future state (Prognosis), and select the most suitable Recovery policy in case of detection of current or future anomalies or failures (Sec. 4).

2. BASIC NOTIONS ABOUT DBN

BN [5] are defined by a *direct acyclic graph* (DAG) where each node is a discrete random variable. Each node has associated a *conditional probability table* (CPT) specifying the probability of each value of the node, conditioned by every instantiation of parent nodes. In this way, it is possible to include local conditional dependencies, by directly specifying the causes that influence a given effect. This allows computing the probability distribution of any variable given the observation of the values of any subset of the other variables. DBN extend BN by providing a discrete temporal dimension. A DBN is essentially the replication of a BN over two time slices, $t - \Delta$ and t, where Δ is the time discretization step. If a variable is

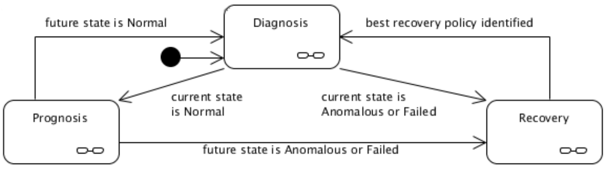


Figure 1. The UML state-chart diagram of ARPHA.

characterized by a temporal evolution, its instance in the time slice t depends on its instance in $t - \Delta$. Moreover, it is possible to establish dependencies involving different variables, in the same time slice, or in different time slices. Given a set of observations up to the current time t, it is possible to compute the probability distribution of a variable at t, in the future, or in the past, conditioned by the observations. A way to efficiently compute conditioned probabilities on a (D)BN, consists of generating and analyzing the JT according to the procedures detailed in [8]. A JT is an undirected unrooted tree where each node (also called a cluster) corresponds to a set of nodes in the original (D)BN. In ARPHA, we implemented the parametric JT-based inference strategy called the Boyen-Koller (BK) algorithm [9].

3. ARPHA PROCESS

ARPHA runs on-board. The external components that interact with ARPHA are: *System Context* (memory area that contains data received from sensors, and the configuration of the system), *Autonomy Building Block* (ABB) (dedicated to plan execution and plan generation), *Event Handler* (manager of events, receiving from ARPHA the identificators of the suggested recovery policies). As depicted in Fig. 1, ARPHA cyclically performs:

- **Diagnosis** begins with the retrieval of sensor data and plan data from System Context and ABB respectively. Both kinds of data are converted into observations concerning the variables of the on-board model. Then, the model inference (analysis) conditioned by observations, is performed at the current time. The inspection of the probabilities of specific variables can provide the system state at the current mission time: the possible states are *Normal* (no anomalies or failures are detected), *Anomalous* or *Failed*. If the current state is *Normal*, then Prognosis is performed, else Recovery is performed (Fig. 1).
- **Prognosis** consists of the future state detection. The on-board model is analyzed in the future according to a specific time horizon and taking into account observations given by future plan actions. Future state is detected according to the probability distribution obtained for the variables representing the system state. The future state can be *Normal*, *Anomalous*, or *Failed*. In case of *Normal* state, the ARPHA on-board process restarts with the Diagnosis phase, otherwise Recovery is performed (Fig. 1).
- **Recovery** is performed in this way: given the detected anomaly or failure, the recovery policies facing that anomaly or failure are retrieved from System Context. In particular, each recovery policy is composed by a set of recovery actions, possibly to be executed at different times. Each policy is evaluated in this way: **1)** each policy is converted into a set of observations for the on-board model variables representing actions; **2)** such observations are loaded in the on-board model which is analyzed in the future; **3)** according to the probability distribution returned by the analysis, and a specific utility function, the expected utility (EU) of the policy is computed. The utility functions provides an utility value given the values of specific variables and their probabilities in the future. In other words, EU quantifies the future effects of the recovery policy on the system. The policy providing the best EU is selected and notified to the Event Handler for the execution. Then, the ARPHA on-board process restarts with the Diagnosis phase (Fig. 1).

4. A CASE STUDY

An example case study we have used to test ARPHA, concerned the power supply system of a Mars rover, with a particular attention to the following aspects:
Solar arrays. We assume the presence of three solar arrays (SA), namely SA1, SA2, SA3. In particular, SA1 is composed by two redundant strings, while SA2 and SA3 are composed by three strings. Each SA can generate power if both the following conditions hold: 1) at least one string is not failed; 2) the combination of sun aspect angle (SAA), optical depth (OD), and local time (day or night) is suitable. In particular, the OD is given by the presence or absence of shadow or storm. The total amount of generated power is proportional to the number of SAs which are actually working.
Load. The amount of load depends on the current action performed by the rover.
Battery. We assume the battery to be composed by three redundant strings. The charge of the battery may be steady, decreasing or increasing according to the current levels of load and generation by SAs. The charge of the battery may be compromised by the damage of the battery occurring in two situations: all the strings are failed, or the temperature of the battery is too low.

We are interested in four failure or anomaly scenarios. In the evaluation of such scenarios, we assume that each one can be recovered by a set of potential policies: the goal of ARPHA is to identify the scenario, and then to select the best policy (in terms of EU) among those that can potentially be applied in the identified situation.
Scenario S1. In this scenario, we have the presence of a terrain slope that increases the SAA, causing lower power generation by SAs. The scenario S1 occurs when a low power generation and a not optimal SAA are both present in the system. The SAA influences the degree of power generation; for instance, the SAA is optimal if the sun is perpendicular with respect to the SAs of the rover. The occurrence of the scenario may determine the *Anomalous* state or the *Failed* state of the system according to the degree of generated power (low or very low generation, respectively). Two recovery policies may be applied in

case of detection of S1, with the aim of reducing the negative effects: **P1)** transition to stand-by mode including the suspension of the plan, with the aim of reducing the load while the power generation is limited. **P2)** change of inclination of SA2 and SA3, with the goal of improving the SAA and consequently the level of power generation (the tilting system cannot act on SA1).

Scenario S2. In this scenario, we have the presence of dust or shadow that increases the OD and reduces the power generated by SAs. The scenario S2 occurs when both a low power generation and a compromised OD are present in the system. The presence of dust in the air or the rover positioned in a shadowed area, reduce the irradiation of SAs, and the degree of generated power, as a consequence. In the best situation, there is no dust and the OD is null. The OD grows proportionally to the density of dust in the air, or the level of obscurity. The occurrence of S2 determines the *Anomalous* state or the *Failed* state, given the level of power generation. Two recovery policies may be applied to face S2 and improve the situation: **P3)** movement of the rover into another position, in order to try to avoid the dust or the shadow, and improve the power generation as a consequence. **P4)** modification of the inclination of SA2 and SA3 (to compensate for the OD, with a better SAA), retraction of the drill if a drilling operation is under execution (safe condition for the drilling device), and transition to stand-by mode (to reduce the load).

Scenario S3. In this scenario, we have an unexpected high request of energy by the drilling operation. The power generated by SAs may not be enough to cope with the request of energy, so the battery may be used to provide the additional power. The scenario S3 occurs when the level of battery charge is not optimal during the drilling operation. This leads the system to the *Anomalous* or *Failed* state according to degree of charge of the battery. The recovery policies facing S3 are the following: **P4)** as above; the goal is improving the power generation thanks to a better SAA, and avoiding the use of the battery by suspending the drill. **P5)** suspension of the plan, retraction of the drill if drilling is under execution, and transition to stand-by mode (with the aim of reducing the load).

Scenario S4. If the battery is damaged, the battery charge level may become low. The scenario S4 occurs when both the battery damage and the low battery charge characterize the system. The damage may be due to the low temperature of the battery or the failure of its strings. In this case, we have a damage due to a low external temperature; such a fault is transient, since an increase of the temperature can bring back the battery to work correctly. The *Anomalous* or *Failed* state depends on the degree of charge. The recovery policies considered for S4 are **P2** and **P4** defined as above. In this case, the aim of the policies is to suspend the plan and then to try to get power from a better inclination of SAs. Notice that P2 and P4 are potentially different (even when a drilling operation is not involved), since P2 first stops the current plan and then moves the SAs, while P4 tries to move SAs before stopping the plan.

4.1. The DBN model of the case study

The DBN model (Fig. 2.a) of the case study has the following features:

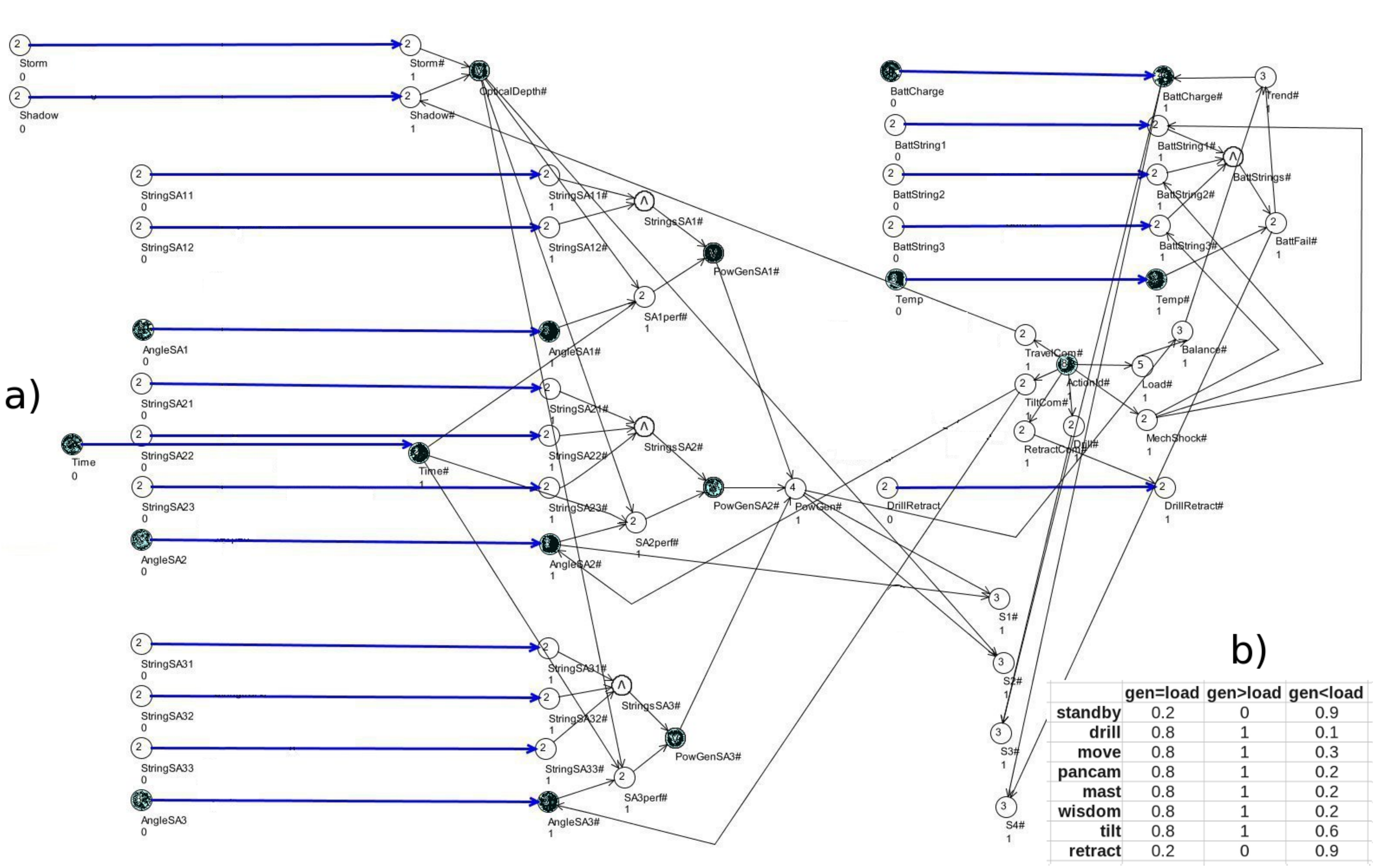
Solar arrays. The variables representing the functioning or failure of basic components or subsystems, are binary; for example, *StringSA11* and *StringSA12* represent the state of a string of SA1, while *StringsSA1* represents the state of the set of strings. The variables modeling environmental conditions are binary, in order to represent the presence or absence of such conditions; this is the case of the variables *Storm* and *Shadow* influencing *OpticalDepth*, and the variable *Time* (day or night). The variable *AngleSA1* representing the SAA of SA1, is ternary (good, discrete, bad). The fact that the combination of *OpticalDepth*, *Time*, *AngleSA1* allows or compromises the power generation by SA1, is represented by the binary variable *SA1perf*. Such variable, together with *StringsSA1*, influences the binary variable *PowGenSA1* modelling the presence or absence of power generation by SA1. SA2 and SA3 are modelled in the same way. The variable *PowGen* is influenced by *PowGenSA1*, *PowGenSA2*, *PowGenSA3*. The size of *PowGen* is 4, in order to represent 4 intermediate levels of power generation depending on the number of SAs generating power.

Load. The size of the variable *ActionId* is 8, in order to represent 8 actions of interest in the model (in this example, actions may concern either the plan or the recovery). Such variable influences *Load* whose size is 5, in order to represent 5 intermediate levels of power consumption according to the action under execution. The variable *Balance* is ternary and indicates that *PowGen* is equal, higher or lower than *Load*.

Battery. The state (working or failed) of each redundant string composing the battery is represented by *BattString1*, *BattString2*, *BattString3*, while the state of the set of strings is modelled by *BattStrings*. The variable *Temp* is ternary and represents the temperature of the battery (low, medium, high). *Temp* and *BattStrings* influence *BattFail* representing the damage of the battery. The variable *Trend* indicates that the battery charge is steady, increasing or decreasing, according to *Balance* and *BattFail*. Four intermediate levels of battery charge are represented by the variable *BattCharge*.

Scenarios. The variables *S1*, *S2*, *S3*, *S4* are ternary, in order to represent the states *Normal*, *Anomalous* and *Failed* in each scenario (the *Normal* state indicates that the scenario does not currently occur).

In the DBN in Fig. 2.a, the instance of a variable at time slice t is distinguished from the instance at time slice $t - \Delta$, by the symbol # following the name of the variable. For instance, *StringSA11* is present in $t - \Delta$, while *StringSA11#* is present in t. If a variable has a temporal evolution its two instances are connected by a temporal arc (appearing as a thick line in Fig. 2.a). Still in Fig. 2.a, the observable variables appear as black nodes; the values coming from sensors, ABB, and recovery policies, will become observations for such variables. In Fig. 2.b, we provide the utility function (on a $[0, 1]$ scale) necessary to compare the recovery policies in case of failure or



	gen=load	gen>load	gen<load
standby	0.2	0	0.9
drill	0.8	1	0.1
move	0.8	1	0.3
pancam	0.8	1	0.2
mast	0.8	1	0.2
wisdom	0.8	1	0.2
tilt	0.8	1	0.6
retract	0.2	0	0.9

Figure 2. a) DBN model of the case study. b) Utility function for policies.

anomaly detection due to power supply problems. In particular, the utility function provides an utility value depending on the plan or recovery action under execution and the balance between power generation and load. Notice that there are actions ("standby" and "retract" of the drill) that, when used in recovery policies, are meant to avoid the rover halting in an unsafe condition (for example with the drill locked on the ground, with the danger of icing during the night); such actions have a large utility for cases corresponding to low power supply situations (meaning that they are useful for recovering situations where the generated power is less than the load), while they have a small utility in case the power supply is sufficient to cover the required load. Remaining actions (which are also usual standard plan operations of the rover) have a smaller utility in correspondence to cases related to low power supply situations, since in such cases one should avoid power consuming actions.

4.2. Executing ARPHA

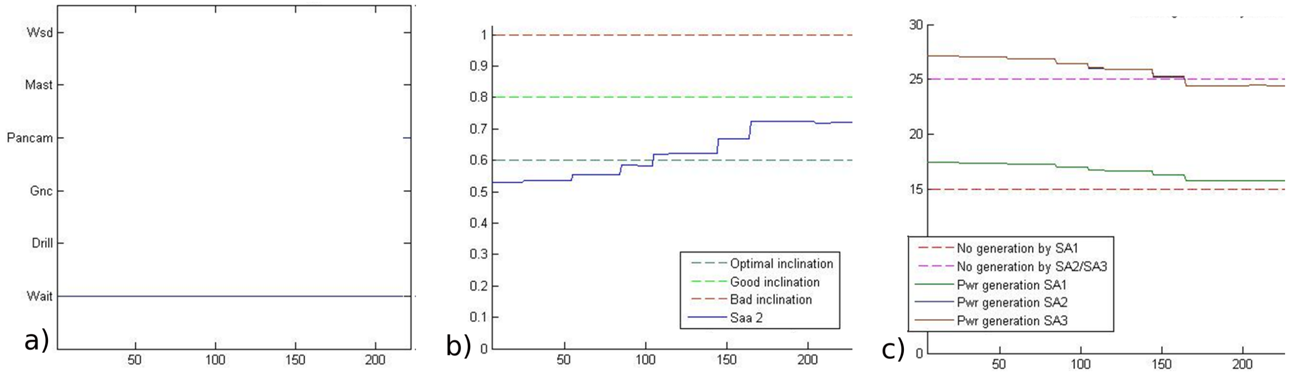
In order to perform an empirical evaluation of the approach, ARPHA has been deployed in an evaluation platform composed by a workstation linked to a PC via Ethernet cable. A rover simulator (called ROSEX) has been installed on the workstation. On the PC we installed the TSIM environment, emulating the on-board computing hardware environment (LEON3), and ARPHA running as a task of RTEMS. Sensors data and plan data that the simulator provides, are the following: OD, power generated by each SA, SAA of SA1, SA2, SA3, charge and temperature of the battery, mission elapsed time, action under execution, plan under execution.

At each mission step (in DBN, time is discretized), the current sensor data and plan action are retrieved and converted into current or future observations for the variables in the on-board model. Such observations are expressed as the probability distribution of the possible variable values. For example, in Fig. 3.c, the "wait" (another name for "standby") action in the plan at the mission step 3, is converted in the probability distribution $1, 0, 0, 0, 0, 0, 0, 0$ concerning the variable *ActionId* in the same step. The first value of the distribution indicates that the first possible value of the variable (0) has been observed with probability 1. This is due to the fact that *ActionId* represents in the model the plan action (or the recovery action). In particular, the value 0 corresponds to the "wait" action. An example about sensor data is the sensor $pwrsa1$ providing the value 15.71248 at the mission step 3 (see Fig. 3.c). This value becomes the probability distribution $1, 0$ for the values of the variable *PowGenSA1*. In other words, *PowGenSA1* is observed equal to 0 with probability 1 at the same time step, in order to represent that SA1 is generating enough power ($pwrsa1 > 15$) in that step (the value 1 instead, represents that the generated power is too low). We assume that the power is high if greater than 15 in the case of SA1, and greater than 25 in the case of SA2 and SA3 (SA1 is composed by two strings, while SA2 and SA3 are composed by three strings).

We provide an example of ARPHA execution during a simulated mission, in the scenarios defined at the begin of Sec. 4.

S1: slope of terrain. In Fig. 3.a, we show the plan under execution; in Fig. 3.b, we show the SAA used to generate the power generation profile (Fig. 3.c), exploiting the rover simulator (ROSEX). In time steps from 0 to 2, ARPHA estimates that both the current and the future state are *Normal*. At time step 3 (mission time: 187 sec.), diagnosis detects that the current state is *Anomalous*. The output of ARPHA in this step is shown in Fig. 3.d: lines 01-04 contain the values of the sensors (generated by ROSEX) and the plan action under execution (*SVF_action*); lines 06-17 concern Diagnosis. In particular, at lines 07-08, the plan action (*SVF_action*=1="wait") performed in the current step is converted into the observation $ActionId = 0$; at lines 09-11, the sensor values are mapped into observations of the corresponding variable values: $pwrsa1 =$ 15.71248 becomes $PowGenSA1 = 0$ (power generation by SA1 is high), $pwrsa2 = 24.40224$ becomes $PowGenSA2 = 1$ (power generation by SA2 is low), $pwrsa1 = 24.41719$ becomes $PowGenSA3 = 1$ (power generation by SA3 is low), $saa1 = 0.722340$ becomes $AngleSA1 = 1$ (SAA1 is not optimal), $saa2 =$ 0.722340 becomes $AngleSA2 = 1$ (SAA2 is not optimal), $saa3 = 0.722340$ becomes $AngleSA3 = 1$ (SAA3 is not optimal), etc. Given such observations, ARPHA performs the inference of the model at the current time step (line 12), querying the variables $S1\#$, $S2\#$, $S3\#$, $S4\#$ representing the occurrence of the scenarios (lines 13-16). The probability that $S1\# = 1$ is higher than a predefined threshold (line 15). In this way, ARPHA detects the scenario S1 and the *Anomalous* state of the system (line 17); as a consequence, Recovery is activated (lines 18-32), in order to evaluate the policies P1 and P2, suitable to deal with S1. At lines 19-24, the actions inside P1 become observations in the next time steps, for the variable $ActionId$. In particular, we observe the "wait" action in the future time steps from 4 to 13. Given such observations, the model is inferenced for 10 steps in the future (line 25) and the utility is computed (line 26). The same process is applied to P2 (lines 27-31). The actions inside P2 become evidences for $ActionId$ at time steps 4 and 5 where the "tilt" action (SA inclination) is observed. The number of actions inside a policy is not constant. Therefore the duration of a policy depends on the number of actions and the duration of each action. For instance, P1 and P2 generate observations for 10 and 2 time steps in the future, respectively, because of their internal actions. The policy P1 provides a better utility, so ARPHA proposes its application (line 32). In this case, P1 is actually better, since the position of the rover on the slope is critical and the safest autonomous action is to stop.

S2: presence of dust or shadow. In Fig. 4.a, we show the plan under execution. In Fig. 4.b, the ROSEX profile of OD is plotted. In Fig. 4.c, the ROSEX profile of power generated by SAs is plotted. At time steps 0, 1, 2, ARPHA detects the *Normal* state as the result of both Di-



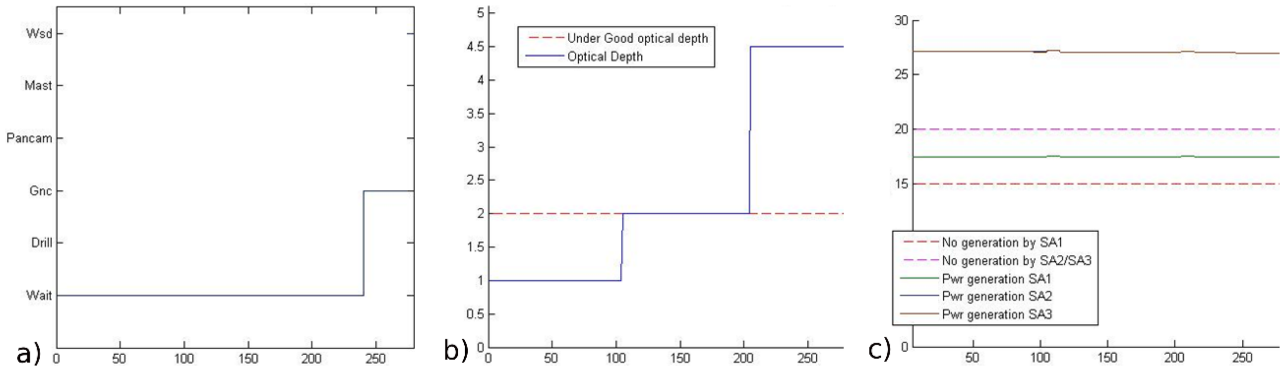
d) ARPHA output:

```
00   *** MISSION STEP: 3 (MISSION TIME: 187 sec.) ***
01   *************** ROSEX VALUES ***************
02   opticaldepht = 1.00000     pwrsa1 = 15.71248       pwrsa2 = 24.40224       pwrsa3 = 24.41719
03   saa1 = 0.72340             saa2 = 0.72340          saa3 = 0.72340          batterycharge = 90.24533
04   batttemp = 273.00000       time = 10.04274         SVF_action = 1          SVF_plan = 1
05   ********************************************
06     *** Diagnosis ***
07   Propagate PLAN STREAM
08   3:ActionId#:1 0 0 0 0 0 0 0 :
09   Propagate SENSORS STREAM
10   3:OpticalDepth#:1 0 :      3:PowGenSA1#:1 0 :      3:PowGenSA2#:0 1 :      3:PowGenSA3#:0 1 :      3:AngleSA1#:0 1 0:
11   3:AngleSA2#:0 1 0 :        3:AngleSA3#:0 1 0 :     3:BattCharge#:0 0 0 1 : 3:Temp#:0 1 0 :         3:Time#:1 0 :
12   Current inference (STEP 3)
13   Pr{S1#=2} = 0.00000000 < 0.99000000     Pr{S2#=2} = 0.00000000 < 0.59000000
14   Pr{S3#=2} = 0.00000000 < 0.99000000     Pr{S4#=2} = 0.00000000 < 0.99000000
15   Pr{S1#=1} = 1.00000000 >= 0.99000000    Pr{S2#=1} excluded because under recovery or minor criticality
16   Pr{S3#=1} = 0.00000000 < 0.99000000     Pr{S4#=1} = 0.00000000 < 0.99000000
17   SYSTEM STATE: "ANOMALOUS" (S1#=1)
18     ## Preventive Recovery ##
19   Policy to convert: P1
20   Propagate POLICY STREAM
21   4:ActionId#:1 0 0 0 0 0 0 0 :      5:ActionId#:1 0 0 0 0 0 0 0 :   6:ActionId#:1 0 0 0 0 0 0 0 :
22   7:ActionId#:1 0 0 0 0 0 0 0 :      8:ActionId#:1 0 0 0 0 0 0 0 :   9:ActionId#:1 0 0 0 0 0 0 0 :
23   10:ActionId#:1 0 0 0 0 0 0 0 :     11:ActionId#:1 0 0 0 0 0 0 0 :  12:ActionId#:1 0 0 0 0 0 0 0 :
24   13:ActionId#:1 0 0 0 0 0 0 0 :
25   Future inference (STEP 13)
26   Utility Function= 0.4736
27   Policy to convert: P2
28   Propagate POLICY STREAM
29   4:ActionId#:0 0 0 0 0 0 1 0 :      5:ActionId#:0 0 0 0 0 0 1 0 :
30   Future inference (STEP 13)
31   Utility Function= 0.0622
32   Best policy for Preventive Recovery is P1
```

Figure 3. Scenario S1: a) plan under execution. b) Sun Aspect Angle (SAA) of solar array SA2. c) Power generation by solar arrays SA1, SA2, SA3. d) ARPHA output at time step 3 (mission time: 187 sec.).

agnosis and Prognosis. According to the output reported in Fig. 4.d, at time step 3 (mission time: 218 sec.), the *Normal* state is still detected by Diagnosis, but not by Prognosis: lines 01-04 show the sensor values and the plan action. Such data are converted into observations at lines 07-11. In particular, the variable *OpticalDepth#* is observed equal to 1, expressing that the OD is not optimal, while the variables *PowGenSA1#*, *PowGenSA2#*, *PowGenSA3#* are observed equal to 0, expressing that the power generation is high. The model undergoes inference (line 12), no scenarios are detected (lines 13-14), and the current state of the system is *Normal* (line 15). Therefore Prognosis is activated: the future actions in the plan become observations for the variable *ActionId* (lines 17-18); then, the model inference is executed (line 19), still querying the variables $S1$, $S2$, $S3$, $S4$ (lines 20-22), but in the four next time steps. ARPHA detects S2 and in particular, the *Failed* state (line 23). Therefore Recovery is activated; the suitable policies are P3 and P4. The actions inside P3 are converted into observations in the next time steps for the variable *ActionId* (lines 25-27), the inference of the model for 10 time steps is executed (line 28), and the utility function is computed (line 29). The same process is applied to P4 (lines 30-36) which provides a better utility, so P4 is suggested by ARPHA. In this case, this is justified, since the movement in another position (P3) does not guarantee to improve power generation, while the tilting action in P4 is more effective.

S3: High energy request by drill. In Fig. 5.a, we show the plan under execution, while in Fig. 5.b, we show the ROSEX battery profile that decreases in linear way. No anomalies or failures are detected from time step 0 to time step 4. At time step 5 (mission time: 338 sec.), the variable $ActionId$ is observed to be equal to 1 (*SVF_action*=2="drill"), while $BattCharge\#$ is observed to be equal to 2, expressing a reduced level of the battery charge provided by the sensor $batterycharge =$ 89.04301. The model inference at the current step returns the *Anomalous* state concerning the scenario S3. Recovery is activated: the policies P4 and P5 are evaluated; P5 is suggested by ARPHA because of the better utility computed by model inference in the future, according to the observations about policy actions. As in the case of S1, putting the rover in a safe configuration is the priority; P5 allows to retract the drill (to avoid a future damage) without taking the risk of loosing power in the attempt of



d) ARPHA output:

```
*** MISSION STEP: 3 (MISSION TIME: 218 sec.) ***
*************** ROSEX VALUES ***************
opticaldepht = 4.50000     pwrsa1 = 17.22273       pwrsa2 = 26.67850       pwrsa3 = 26.67641
saa1 = 0.51575             saa2 = 0.51575          saa3 = 0.51575          batterycharge = 90.28925
batttemp = 273.00000       time = 10.05112         SVF_action = 1          SVF_plan = 1
*********************************************
  *** Diagnosis ***
Propagate PLAN STREAM
3:ActionId#:1 0 0 0 0 0 0 0
Propagate SENSORS STREAM
3:OpticalDepth#:0 1 :    3:PowGenSA1#:1 0 :      3:PowGenSA2#:1 0 :        3:PowGenSA3#:1 0 :     3:AngleSA1#:1 0 0 :3:
AngleSA2#:1 0 0 :        3:AngleSA3#:1 0 0 :     3:BattCharge#:0 0 0 1 :   3:Temp#:0 1 0 :        3:Time#:1 0 :
Current inference (STEP 3)
Pr{S1#=2}=0.000<0.990   Pr{S2#=2}=0.000<0.590   Pr{S3#=2}=0.000<0.990   Pr{S4#=2}=0.000<0.990
Pr{S1#=1}=0.000<0.990   Pr{S2#=1}=0.000<0.590   Pr{S3#=1}=0.000<0.990   Pr{S4#=1}=0.000<0.990
SYSTEM STATE: "Normal"
  ## Prognosis ##
Propagate PLAN STREAM
4:ActionId#:1 0 0 0 0 0 0 0 :   5:ActionId#:1 0 0 0 0 0 0 0 :   6:ActionId#:1 0 0 0 0 0 0 0 :   7:ActionId#:0 0 1 0 0 0 0 0 :
Future inference (STEP 7)
Pr{S1#=2}=0.38471501<0.99   Pr{S2#=2}=0.60604805>=0.59   Pr{S3#=2}=0.01966910<0.99   Pr{S4#=2}=0.05214530<0.99
Pr{S1#=1} excluded because under recovery or minor criticality   Pr{S2#=2} excluded because under recovery or minor criticality
Pr{S3#=1}=0.09944675<0.99   Pr{S4#=1}=0.29860398<0.99
FUTURE SYSTEM STATE: "Failed" (S2#=2)
  ## Preventive Recovery ##
Policy to convert: P3
Propagate POLICY STREAM
4:ActionId#:0 0 1 0 0 0 0 0 :      5:ActionId#:0 0 1 0 0 0 0 0 :
Future inference (STEP 13)
Utility Function = 0.0890
Policy to convert: P4
Propagate POLICY STREAM
4:ActionId#:0 0 0 0 0 0 1 0 :   5:ActionId#:0 0 0 0 0 0 1 0 :   6:ActionId#:0 0 0 0 0 0 0 1 :   7:ActionId#:0 0 0 0 0 0 0 1 :
8:ActionId#:1 0 0 0 0 0 0 0 :   9:ActionId#:1 0 0 0 0 0 0 0 :   10:ActionId#:1 0 0 0 0 0 0 0 :  11:ActionId#:1 0 0 0 0 0 0 0 :
12:ActionId#:1 0 0 0 0 0 0 0 :  13:ActionId#:1 0 0 0 0 0 0 0 :
Future inference (STEP 13)
Utility Function= 0.8764
Best policy for Preventive Recovery is P4
```

Figure 4. Scenario S2: a) plan under execution. b) Optical Depth (OD). c) Power generation by solar arrays SA1, SA2, SA3. d) ARPHA output at time step 3 (mission time: 218 sec.).

changing inclination of SAs as in P4.

S4: Damage to battery. In Fig. 6.a and Fig. 6.b, we provide the ROSEX profile of battery temperature and charge, respectively. No anomalies or failures are detected from time step 0 to time step 26. At time step 27 (mission time: 1816 sec.), the variable $BattCharge\#$ is observed to be equal to 2, expressing a reduced level of the battery charge provided by the sensor $batterycharge = 89.34396$, while the variable $Temp\#$ is observed to be equal to 0, expressing the low temperature of the battery as indicated by the sensor $batttemp = 253$. The model inference at the current step returns the *Anomalous* state concerning the scenario S4. Recovery is activated: the policies P2 and P4 are evaluated: P4 is suggested by ARPHA. This is justified by the fact that the power level is enough to try a tilt before stopping the plan.

5. CONCLUSIONS

In ARPHA, FDIR functions are not simply based on sensor monitoring and look-up tables as in the traditional approach, but we perform a reasoning based FDIR exploiting DBN modelling and analysis (inference). To this aim, DBN can represent: both observable and not observable conditions or events, multi-state components or functions (e.g. power generation, load, battery charge and temperature), effects of actions and external conditions on the system (e.g. the effect of tilting on SAA, the load changing according to the action). In the ARPHA process, Diagnosis (current state detection) takes into account data coming from sensors and plan, while Prognosis (future state detection) is possible by means of the analysis in the future of the on-board model. Recovery may be a consequence of Diagnosis. In this case, Recovery has a reactive role and tries to solve the detected anomaly or failure. If instead, Recovery follows Prognosis, then we

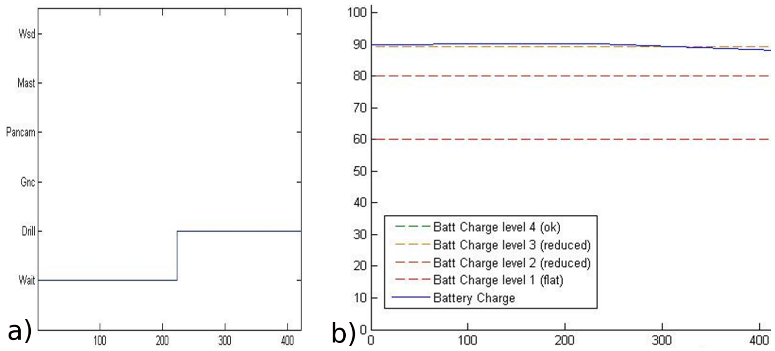


Figure 5. Scenario S3: a) plan under execution. b) charge of the battery.

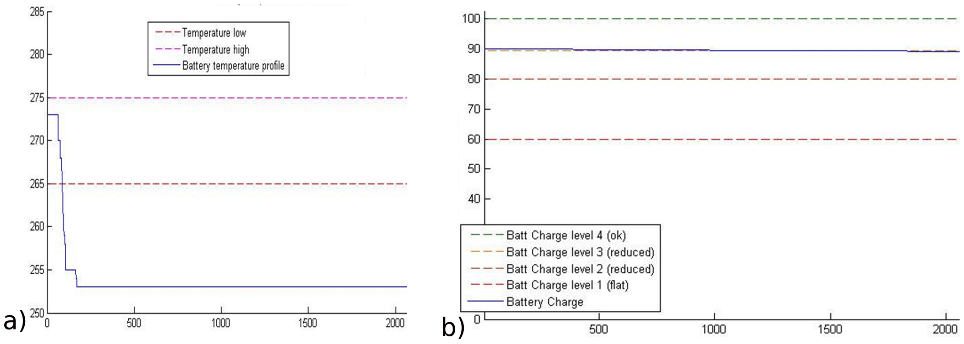


Figure 6. Scenario S4: a) temperature of the battery. b) charge of the battery.

can talk about preventive recovery trying to avoid imminent anomalies or failures. In both cases, Recovery selects the most useful policy. A policy is a set of actions, possibly executed at different times, and is evaluated taking into account the future effect of the policy on the system. This is done by means of the model analysis in the future. All these capabilities are performed in automatic way, without the assistance of the ground control (this increases the achievable level of autonomy), and have been tested through a case study taking into account four particular scenarios.

ACKNOWLEDGMENTS

This work has been funded by ESA/ESTEC under the VERIFIM study (grant n. TEC-SWE/09259/YY).

REFERENCES

[1] P. Robinson, M. Shirley, D. Fletcher, R. Alena, D. Duncavage, and C. Lee. Applying model-based reasoning to the FDIR of the command and data handling subsystem of the ISS. In *Proc. iSAIRAS 2003*, Nara, Japan, 2003.

[2] M. Schwabacher, M. Feather, and L. Markosian. Verification and validation of advanced fault detection, isolation and recovery for a NASA space system. In *Proc. Int. Symp. on Software Reliability Engineering*, Seattle, WA, 2008.

[3] W. Glover, J. Cross, A. Lucas, C. Stecki, and J. Stecki. The use of PHM for autonomous unmanned systems. In *Proc. Conf. PHM Society*, Portland, OR, 2010.

[4] K. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD Thesis, UC Berkley, 2002.

[5] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[6] J.B. Dugan, S.J. Bavuso, and M.A. Boyd. Dynamic fault-tree models for fault-tolerant computer systems. *IEEE Trans. on Reliability*, 41:363–377, 1992.

[7] S. Montani, L. Portinale, A. Bobbio, and D. Codetta-Raiteri. RADYBAN: a tool for reliability analysis of dynamic fault trees through conversion into dynamic Bayesian networks. *Reliability Engineering and System Safety*, 93(7):922–932, 2008.

[8] C. Huang and A. Darwiche. Inference in Belief Networks: A Procedural Guide. *International Journal of Approximate Reasoning*, 15:225–263, 1996.

[9] X. Boyen and D. Koller. Tractable inference for complex stochastic processes. In *Proc. UAI 1998*, pages 33–42, 1998.