DIAGNOSIS OF TECHNOLOGICAL SYSTEMS BASED ON THE STRUCTURAL DECOMPOSITION OF THEIR COLOURED PETRI NET MODEL

ANNA I. PÓZNA,[1*] MIKLÓS GERZSON,[1] ADRIEN LEITOLD,[2] AND KATALIN M. HANGOS[1,3]

[1] Department of Electrical Engineering and Information Systems, University of Pannonia, PO Box 158, Veszprém, 8201, HUNGARY
[2] Department of Mathematics, University of Pannonia, PO Box 158, Veszprém, 8201, HUNGARY
[3] Institute for Computer Science and Control, Hungarian Academy of Sciences, PO Box 63, Budapest, 1518, HUNGARY

Diagnosing faults during the operation of a system is an essential task when investigating technological systems. In this paper, a new online fault identification method is proposed which is based on the occurrence graph of the coloured Petri net model of the system. The model is able to simulate the normal and faulty operations of the system given in the form of event lists, so called traces. The diagnosis is based on the search for deviations between the traces of the normal and the actual operations. In the case of complex technological systems, the occurrence graph can contain hundreds of nodes; therefore, the computational effort and searching-time increase significantly. Our proposed structural decomposition method can manage these demands so it has a crucial impact on the practical application of diagnostic processes. The main idea of our method is that the complex systems can be decomposed into technological units. Therefore, the diagnosis can be done by components separately and the diagnostic result of a unit can be used for the diagnosis of the other units connected to it. Because of the structural decomposition, the diagnosis has to be performed on much smaller occurrence graphs but the effect of faults in previous units is taken into account. The proposed method is illustrated by a simple case study.

Keywords: technological system, diagnosis, coloured Petri net model, structural decomposition, qualitative model

1. Introduction

Identifying faults and analysing their consequences are important tasks during the investigation of technological systems. A number of diagnostic methods are known in the literature and the model-based methods are very popular among them [1].

Petri nets and their different extensions (such as coloured, timed or hierarchical nets) are powerful tools for modelling discrete event systems [2]. For example, coloured Petri nets are often used for modelling production lines [3]. It is important that the resultant models describe not only the normal (faultless) operation of the system, but they also take into account different, randomly occurring errors in the system.

In many cases, the normal or faulty operations of technological processes can be characterised by a series of events possessing discrete or qualitative valued variables. In this case, the occurring deviations can be generated by the comparison of the normal and actual events. The occurring faults can be detected and identified based on the observed deviations.

Discrete event systems are usually modelled by automata. In this case, the diagnosis is usually based on the idea of unobservable events [4]. Faults can be modelled as unobservable events, which means only the effects of faults can be noticed. The problem with fault detection is specifying whether any fault has occurred or not in the system. Fault isolation is the problem of identifying which fault has occurred exactly. Since faulty events are unobservable by assumption, the detection and isolation problem must be solved based on the available information of the observed non-faulty events. The diagnosability of discrete event systems was first investigated [5] using the methods of automata theory.

Besides automata, Petri nets are also frequently used for modelling discrete event systems (DES). The structural and mathematical representations of Petri nets both can be used for diagnostic purposes. Methods include various techniques such as analysis of the occurrence graph, marking estimation, linear algebra, integer linear programming, diagnoser nets, or reverse nets.

A simple fault detection method based on the measurement of token quantity is given [6]. It is assumed that the given Petri net is conservative, and any change in the token quantity is caused by faults. If the difference between measured and initial token number

*Correspondence: pozna.anna@virt.uni-pannon.hu

exceeds a predefined threshold then a fault has occurred. Sensor signals are used for token determination instead of modelling the faulty behaviour of the system. The proposed method is very simple and can be used for early fault detection; however, it is not able to isolate faults.

Faults can also be modelled as unobservable transitions in Petri nets. The set of places can be also observable; therefore, the marking of the Petri net has to be estimated. The notion of basis marking [7] (set of markings consistent with the observation) and j-vectors (minimal sequences of unobservable transitions to reach basis markings) are introduced. An online algorithm is developed to detect the occurrence of faults, which uses the basis occurrence graph. The main advantage of the proposed algorithm is that in the case of bounded Petri nets the basis occurrence graph can be computed offline. It reduces the computational effort of the online diagnosis. The basis occurrence graph can be used as an online diagnoser.

Sufficient conditions of diagnosability of faulty transitions are given in the form of a system of inequalities [8]. In this method, the marking of places is observable. Authors introduce the notion of g-markings (markings with negative elements) and unobservable explanations (sequences of unobservable transitions, whose firings can explain the negative elements of a g-marking). After an observed event, the g-marking is updated according to the Petri net equation. When an observed transition fires it removes tokens from its input places and adds tokens to its output places. If this transition is not enabled under the previous g-marking then the removal of tokens causes negative marking. An online fault detection algorithm has also been developed [9], which is based on solving integer linear programming problems and checking the diagnosability conditions.

The integer linear programming approach has also been used [10] to determine if the system behaviour is normal, faulty, or ambiguous. The algorithm has further been improved [11], for a more general situation where different observable transitions can share the same label. Firing times of transitions are also considered, which add more constraints to the ILP problem making the fault detection algorithm more accurate.

Timing characteristics have also been used [12], but with a different meaning: the faults affect the firing speed of the transitions. The fault detection is based on the generation of residuals, which are computed by comparing the markings of observable places with the reference model.

A bottom-up modelling methodology has been proposed using interpreted Petri nets [13]. In the generated model, the faulty and normal states are represented by places. The authors introduce the definition of input-output diagnosability and also give conditions to test this property. The diagnoser model contains the normal behaviour of the system. An online algorithm based on the difference between the system output and the diagnoser model output is developed for detecting faulty markings.

In the case of large systems, the models and associated diagnosers can be extremely large. Furthermore, the diagnostic methods are computationally expensive. Therefore, it is essential to investigate the possibility of distributed diagnosis. The idea of distributed diagnosis is to divide the system into modules or components then make a local diagnoser for each component. The challenging problem is to ascertain the diagnosis state of the whole system from the results of local diagnosers. It usually requires a distributed algorithm and a communication protocol [14].

Coloured Petri nets (CPN) have the advantage of making compact information representations. A CPN diagnoser equivalent to the classical diagnoser has been built [15]. In this approach, places represent different hypotheses and colours represent diagnosis results. The advantage of the CPN diagnoser is the simplified graphical representation. On the other hand, the coloured diagnoser is not necessarily smaller than the classical diagnoser. Decomposition and methods of modular diagnosis of DES are also studied by the authors.

Backward reachability can also be used for diagnosis purposes. If a marking M is reachable from M_0, then M_0 is backward reachable from M in a Petri net. Possible sources of failures for this method can be determined. Backward reachability is extended to coloured Petri nets [16]. Transformation techniques for the inversion of CPN are also presented here.

2. Basic Concepts

A brief description of the basic concepts and notions of our method are given in this section. At first qualitative ranges are introduced to characterise the measured values. After that events, traces, and deviations in a technological system are defined, then the most important parts of the coloured Petri net model and its analysis are introduced. Finally, the structural decomposition-based diagnosis is described in detail.

2.1. Qualitative Range Spaces

In many applications, it is not always necessary to know the exact values of the measured signals. Qualitative models can be used in this case and it is enough to know whether the value of a signal belongs to a specified range space or not. For example, the measurement range of a sensor can be divided into the following range spaces:

$$Q_s = \{\, e_0, 0, L, N, H, e_1 \,\} \qquad (1)$$

where $0, L, N, H$ denote the zero, low, normal and high measured value, respectively, while e_0 and e_1 may refer to the extremely low and high values caused by sensor errors. The states of actuators, e.g. valves, switches, etc. can be described similarly. For example, a two-state valve can be represented by
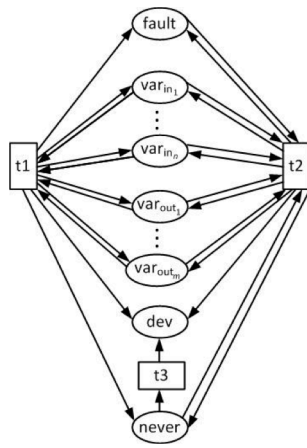
Figure 1. Structure of the Coloured Petri Net Model.

$$Q_v = \{ \, op \, , cl \, \} \tag{2}$$

qualitative range spaces, where *op* and *cl* refer to the open or closed state of the valve.

2.2. Events, Traces, and Deviations

Considering a technological system as a discrete event system, the state of the system can be characterised with the measured values at a given time. The actions in the system, e.g. interactions by operators, modify the values of input and output variables thus the system state changes. An *event* is defined as the arranged (qualitative) input and output values of the system at a given time instance τ:

$$event_\tau = (\tau, in_1, \ldots, in_m, out_1, \ldots, out_n) \; .$$

The course of the system can be described as a sequence of consecutive events, so called *trace*:

$$trace = (event_1, \ldots, event_n).$$

In a technological system the most important types of traces with respect to the diagnosis are the *nominal*, *faulty*, and *characteristic* traces. The nominal trace describes the normal operation of a system. The faulty trace contains the occurring events if a known fault is present while the characteristic trace refers to the actual course of the process. In this paper, it is assumed that only one fault may occur in a process unit of the technological system and this fault evolves before the start of the operation and remains unchanged during the course of the process. If a fault occurs then the trace of the system differs from its nominal trace. As a result, *deviations* between the nominal trace and the current characteristic trace can be defined. In our diagnosis method the following types of deviations are used:

- *never happened(event$_\tau$)* (abbreviated as H(*event$_\tau$*)): This type of deviation refers to the events of the nominal trace which (*event$_\tau$*) never occur in the characteristic trace of the process.

- *chronological deviations*: If an event of the nominal trace (*event$_\tau$*) happens *later* or *earlier* in the characteristic trace than time point τ, the deviations LAT(*event$_\tau$*) and EAR(*event$_\tau$*) denote them.

- *quantitative deviations*: This type of deviations is used to denote that the ith output value is greater (denoted by GRE_i(*event$_\tau$*)) or smaller (SML_i(*event$_\tau$*)) in the characteristic event at time τ than in the nominal event while the input values are identical.

Our diagnosis method is based on the search and comparison of the deviation list on the reachability graph of the CPN model of the technological system.

2.3. Coloured Petri Nets

Coloured Petri nets (CPNs) are extensions of the ordinary or low-level Petri nets. The main differences with respect to ordinary Petri nets are that so-called colours can be assigned to tokens and functions can be assigned to arcs and transitions, too. The detailed formal definition is given in Ref.[17], only the special concepts used in our models are presented here.

- *Places* of the CPN model of the technological system may have three functions. At first input and output variables are represented by places and the colour of the tokens on them denotes the qualitative value of the variable at the current time. On the other hand, places may refer to the occurred fault and the generated deviations. The colours of tokens in these places denote the type of the fault and the occurring deviations, respectively.

- There are three *transitions* in our model, which have different tasks. Transition t_1 is responsible for the generation of faulty or normal operations at the beginning of the process and the initialisation of the variables according to the investigated operational mode. The function of transition t_2 is the timing of the process. It is assumed that the technological process is time-driven and the values of the variables change at the end of the time steps. Therefore, t_2 fires until the end of the process. Transition t_3 is used for the generation of the '*never happened*'-type deviations at the end of the process.

- *Arc functions* are assigned to the arcs between places and transitions defining the change in the colours and computing deviations.

The structure of the coloured Petri net for modelling and diagnosing technological systems can be seen in *Fig.1*. Places are represented with ellipses and transitions are represented with rectangles. In a technological system, the consequences of a processing step can be stochastic. For example, the step may be completed in a normal way, or a fault occurs. The probabilistic nature of a transition *t* associated with a processing step can be modelled in a CPN by a fault function, which is built into its guard function. This

fault function returns the logical value true or false with predefined probability, and the token values of the adjacent consequence places of transition t can be controlled by this logical value. This type of transition firing is called a *stochastically fired transition*.

The *occurrence graph* of the CPN is a graph, which contains all of the system states reachable from a given initial state [17]. Assuming that the CPN model of the examined system is given, the occurrence graph can be used for its behavioural analysis. The nodes of the graph refer to system states and the arcs connecting them refer to state transitions, e.g. events. Different paths on the occurrence graph refer to different operational modes of the system and they can be used for analysing the causes and consequences of a system state.

2.4. Diagnosis with Structural Decomposition

The disadvantage of the occurrence graph-based method is the increasing size of the graph as the size, i.e. number of places, of the CPN model increases. In the case of even a simpler technological system containing three or four units, the occurrence graph of its CPN model can contain hundreds of nodes depending on the number of sensors and actuators. The refinement of the qualitative measuring range of sensors or the application of control valves instead of two-state actuators may also cause the growth of occurrence graphs because their branches will be longer. With the growth in the size of the graph, the computational effort and searching-time also increase. This is the reason why the structural decomposition has a crucial impact on the practical application of the diagnostic process.

Generally speaking, complex systems can be decomposed into technological units. By taking advantage of this, the diagnosis can be done by components separately, and the diagnostic result of a unit can be used for the diagnosis of the other units connected to it.

To perform diagnosis with structural decomposition the full trace of the system should be decomposed, too. To do this, first the time instances belonging to the operation of the investigated units have to be selected. Then the variables referring to this unit are picked out from the events belonging to the selected time instances. If the trace is represented in tabular form then specific rows and columns should be selected. Afterwards, time is shifted back in the case of every unit such that the initial time step of the first event should be 1 in every sub-trace. This means that every unit has its own relative time-scale. By applying this decomposing process, the trace describing the operation of the entire system is disintegrated into the event lists referring to the operation of simpler technological units. As a next step the deviation list of the given subsystem is generated by comparing the nominal trace of the subsystem with its characteristic trace. Then the diagnosis is performed using the CPN model and occurrence graph of the subsystem. The task is to compare the generated deviation list with the token distribution of the terminal nodes on the occurrence graph. If the deviation list corresponds with the token distribution of exactly one terminal node then the fault can be determined based on the token of the fault place. If the deviation list matches the token distribution of more than one terminal node then only the set of possibly occurred faults could be determined. If the deviation list cannot be found in the token distribution of any terminal nodes then an unknown failure occurs in the system.

In the case of complex systems, it is necessary to take into account the effect of faults that have occurred in subsystems connected to the diagnosed unit. Therefore, the CPN model of the units has to be modified such that the place of the *fault* has to contain not only the actual operating mode, but the operating modes of previous subsystems, too. To store this information the colour set of this place is extended with an attribute referring to the type of fault and to the place where the fault occurred.

Let us assume that one fault is diagnosed in the first technological unit. This information is added to the *fault* place of the next unit as a previous fault. Then the occurrence graph of this subsystem is generated based on its CPN model where the fault of the previous unit appears as an initial condition. The resultant occurrence graph now contains those states and deviation lists which can occur in this subsystem if the fault of the previous unit is taken into account. The diagnosis is performed on this graph, the possible fault of this unit is determined based on this investigation, and the result is taken into account during the diagnosis of the following unit. In certain cases, the exact type of fault cannot be determined exactly. If the result of the diagnosis of the unit is a set of possible faults then each element of the set is treated separately. This means that the diagnosis of the next unit has to be performed taking into account every one of the possible previous faults. Occurrence graphs of the subsystem are generated according to each fault of the previous subsystem. The result of the diagnosis is the union of the obtained faults.

The main advantage of the described method is the smaller size of the occurrence graphs of subsystems. Therefore, the search requires less time than in the case of the investigation of the entire technological system.

3. Simple Case Studies

A simple case study is presented in this section as a practical illustration of the diagnosis of complex technological systems based on their structural decompositions.

3.1. Description of the Technological Process

Our simple technological example contains three uniform tanks, *TA*, *TB*, and *TC*, which are serially connected as can be seen in *Fig.2*. Each tank has an input valve, an output valve (denoted by *Vx*, where *x* = A, B, C, and D), and a continuous level sensor (*lev_x*, *x*
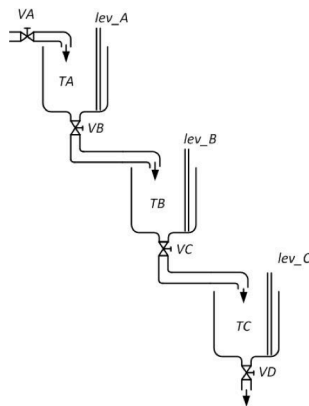
Figure 2. The investigated technological example.

= A, B, and C). The short description of the technological process is as follows: as an initialisation, it is assumed that all the valves are closed. The process starts with the opening of the first valve (*VA*) and then the filling of the first tank (*TA*) starts. The flow of liquid is considered constant so the control of the filling process is based on time. The role of the level sensors is to measure the actual liquid level only. At time step 3, the filling process of tank *TA* completes and its output valve (*VB*) is opened. The second tank *TB* is filled the same way as described for tank *TA*. At time step 5, the *TB* tank is full and its output valve (*VC*) is opened. The filling process of the third tank (*TC*) happens the same way. During the filling of the second and third tanks, the first (*TA*) then the second tank (*TB*) operates as a continuous unit. After the filling of the third tank (*TC*) has completed, the technological systems work in continuous mode.

It is assumed that five possible faults can occur in each tank:

- *2 faults of the level sensor*: negative or positive bias error. In this case, the sensor signal is lower or higher with qualitative unit than the actual value of the level.
- *leaking of the tank*: it is assumed that all of the incoming liquid runs out through the hole, i.e. it is a "big" hole.
- *2 different faults of the output valves VB, VC, and VD*: besides their normal operations, they can stay closed or open only halfway.

For the sake of simplicity, only one of these faults can occur with each tank and all of the faults evolve before the filling process of the first tank *TA* starts.

3.2. CPN Model of the System

For the diagnostic investigation, a coloured Petri net-based (CPN) model of the technological system is developed as follows: the system can be decomposed into three subsystems. Each subsystem represents a tank together with its input and output valves. As can be seen from the technological description of the system, all the subsystems (tanks with their input and output valves and sensors) work in a very similar way, so the structure of their CPN models is identical. The CPN models were
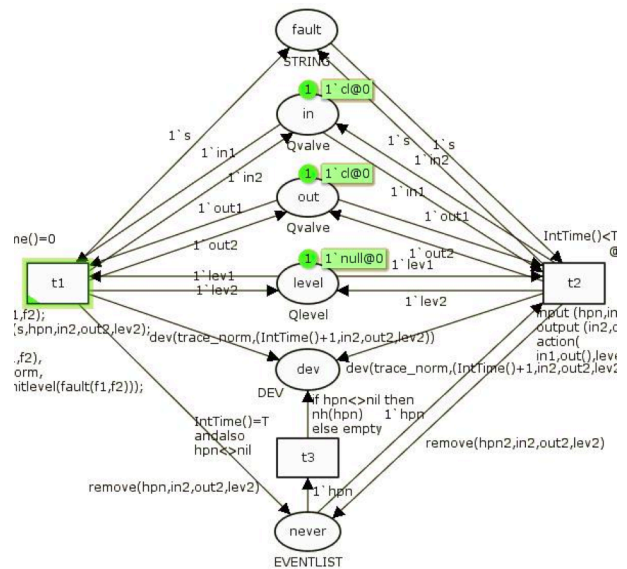

Figure 3. The coloured Petri net model of a tank (for the sake of clarity parts of some inscriptions were omitted).

built using CPN Tools 3.4.0. The CPN model of a subsystem can be seen in *Fig.3*.

The description of the CPN model is as follows: the locations of the Petri nets represent input and output variables, i.e. the state of the input and output valves (denoted by *in* and *out*), and measured level value (*level*), respectively. The qualitative values of these variables are represented by different colour sets.

- colour set of valves: $Q_{valve} = \{cl, op\}$, which represents the closed or open-state of the two-state valve.
- colour set of level sensors: $Q_{level} = \{ e_0, 0, L, N, H, e_1 \}$, where $0, L, N, H$ denotes that the level is zero, low, normal or high, respectively, and e_0, and e_1 indicate that the level is below or above the measurement range, respectively.

Three additional places are needed: one to store the operating mode (place *fault*), one to collect the deviations (place *dev*) and one for the list of events that have not occurred until a given simulation time step (place *never*). The CPN model contains three transitions (t_1, t_2, t_3). t_1 is the initialisation transition, it fires only once at the start of the simulation. It generates an operating mode (normal or faulty) and updates the variables according to the generated operating mode. Afterwards, transition t_2 fires until the end of the process. It updates the values of variables in every simulation step and generates quantitative and chronological deviations except for the '*never happened*'-type. The '*never happened*'-type deviations can be generated after the process has ended. This is done by the firing of transition t_3, which removes the events that remain at place *never* at the end of the simulation and attaches the *NH* guideword to them. The values of variables at a given time step can be read from the trace files. Each trace file contains the list of events that describe the process according to the operating mode. All traces were generated by a MATLAB script.
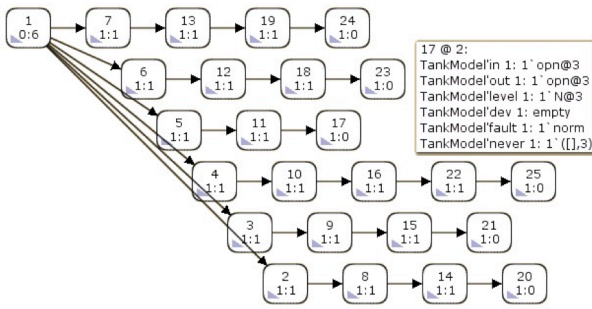
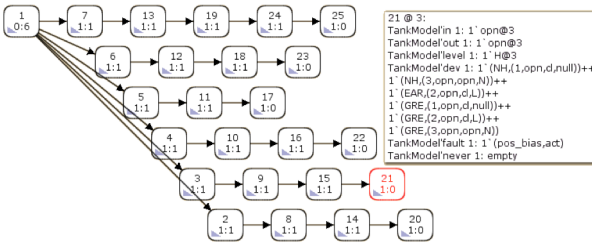*Figure 4*. The occurrence graph of a CPN tank model.

Figure 5. The occurrence graph of the first tank.

3.3. Diagnosis of the System with Structural Decomposition

According to the general description, the technological system consists of three uniform tanks. Because of the same structure and operational mode, all three tanks have the same CPN model. This model can be seen in *Fig.3*. The occurrence graph of the CPN model of a tank can be seen in *Fig.4*. At the time of the generation of this occurrence graph, any fault occurring in a previous technological unit was not taken into account. As can be seen in *Fig.4*, node No. 17 refers to normal operation of the system, while the other terminal nodes (No. 20, 21, 23–25) belong to the five different faulty modes.

The nominal trace of the complex technological system can be seen in *Table 1*. The input variables of the first tank (tank *TA* in *Fig.2*) are the states of valves *VA* and *VB*, while the output variable is the value of level sensor *lev_A*. The variables of the second tank are *VB*, *VC*, and *lev_B*, while *VC*, *VD*, and *lev_C* belong to the third tank, respectively.

The rows belong to time steps 1, 2, and 3, while columns *VA*, *VB*, and *lev_A* compose the trace of the first tank. These cells are framed with a dotted line in *Table 1*. Similarly, rows 3, 4, and 5 as well as columns *VB*, *VC*, and *lev_B* define the trace of the second tank (framed with a continuous line) while rows 5, 6, and 7, along with columns *VC*, *VD*, and *lev_C* give the trace of the third tank (framed with a dashed line). Consider the characteristic trace of the technological system given in *Table 2*.

As a next step, the trace pieces belonging to each individual tank are removed from the characteristic trace of the entire system. The initial time step is shifted to 1 for every unit. The resultant event lists belonging to the three tanks can be seen in *Table 3*.

The diagnostic process is started with the first tank. By comparing the nominal trace of the first tank with the characteristic trace (first column of Table 3) the deviation list is generated. This deviation list is then

Table 1. Decomposition of the nominal trace. Dotted line: first tank, continuous line: second tank, dashed line: third tank.

time	input variables				output variables		
	VA	*VB*	*VC*	*VD*	*lev_A*	*lev_B*	*lev_C*
1	*op*	*cl*	*cl*	*cl*	0	0	0
2	*op*	*cl*	*cl*	*cl*	L	0	0
3	*op*	*op*	*cl*	*cl*	N	0	0
4	*op*	*op*	*cl*	*cl*	N	L	0
5	*op*	*op*	*op*	*cl*	N	N	0
6	*op*	*op*	*op*	*cl*	N	N	L
7	*op*	*op*	*op*	*op*	N	N	N

Table 2. Decomposition of the characteristic trace. Dotted line: first tank, continuous line: second tank, dashed line: third tank.

time	input variables				output variables		
	VA	*VB*	*VC*	*VD*	*lev_A*	*lev_B*	*lev_C*
1	*op*	*cl*	*cl*	*cl*	L	0	0
2	*op*	*cl*	*cl*	*cl*	N	0	0
3	*op*	*op*	*cl*	*cl*	N	0	0
4	*op*	*op*	*cl*	*cl*	N	0	0
5	*op*	*op*	*op*	*cl*	N	0	0
6	*op*	*op*	*op*	*cl*	N	0	0
7	*op*	*op*	*op*	*op*	N	0	0

Table 3. Characteristic traces of the three tanks after decomposition.

TA	TB	TC
$(1, op, cl, L)$	$(1, op, cl, 0)$	$(1, op, cl, 0)$
$(2, op, cl, N)$	$(2, op, cl, 0)$	$(2, op, cl, 0)$
$(3, op, op, N)$	$(3, op, op, 0)$	$(3, op, op, 0)$

searched for among the terminal nodes of the occurrence graph of the first tank. (This occurrence graph can be seen in *Fig.5*). It can be stated that terminal node No. 21 contains the same deviation list and based on the token of the *fault* place the type of fault can be determined: the level sensor exhibits a positive failure bias in the first tank.

The diagnosed fault in the first tank is used during the investigation of the second tank. This fault is added to the place *fault* as a token (*pos_bias*, *prev1*) in the model of the second tank. Then the occurrence graph of the second tank is generated which contains those states that can occur in the second tank if the sensor of the first tank exhibits a positive bias error. The resultant graph can be seen in *Fig.6*.

The deviation list of the second tank is generated by comparing the second column of *Table 3* with the characteristic trace of the second tank. By checking the terminal nodes of the occurrence graph in *Fig.6*, it can be stated that terminal node No. 24 exhibits the same deviation list. This means that the fault of the second tank is leakage and it can be identified unambiguously.

The diagnosed faults of the first and second tanks are added to the model of the third tank in the form of tokens (*pos_bias*, *prev1*) and (*leak*, *prev2*) belonging to the *fault* place. Based on this information the occurrence graph of the third tank is generated in accordance with *Fig.7*. The nodes on this occurrence graph refer to the states if a positive failure bias occurs in the first tank and leak in the second tank.
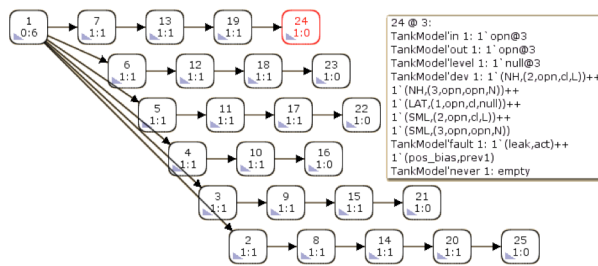
Figure 6. The occurrence graph of the second tank in the case of a positive failure bias in the first tank.
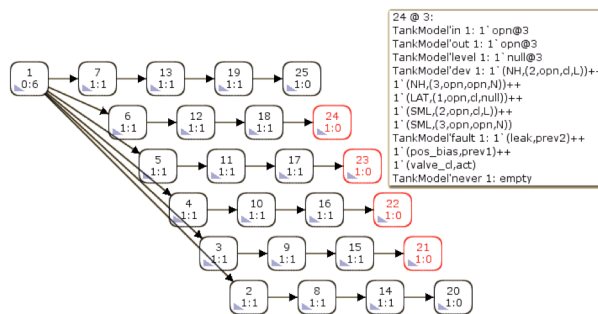

Figure 7. The occurrence graph of the third tank in the case of a positive failure bias in the first tank and leak in the second tank.

The deviation list based on the trace pieces stemming from the filling process of the third tank (see the third column of *Table 3*) is generated and compared to the terminal node of the occurrence graph. It can be stated that terminal nodes No. 21–24 possesses the same deviation list as the deviation list obtained from the characteristic trace of the third tank. This means that the operating mode of the third tank cannot be unambiguously determined, the set of possible operating modes, i.e. normal, leak, or fault, of valves can be defined.

4. Conclusion

A novel method for online fault diagnosis in a technological system is described in this paper. The method is based on the structural decomposition of a complex technological system. The process starts with the modelling of the technological system in the form of coloured Petri nets. For the characterisation of sensor values and actuator states, qualitative value sets are used in the form of coloured tokens. This modelling method allows for the simulation of both normal and known faulty operations of the system. The diagnosis is performed using the occurrence graph of the basic units of the complex system. By generating the deviation list based on the normal and characteristic traces the fault or the set of possible faults can be determined. As a result of the structural decomposition, the diagnosis has to be performed on much smaller occurrence graphs but the effect of faults in previous units are taken into account. Our method reduces the demand of computational efforts and search time. The proposed method was illustrated by a simple case study.

REFERENCES

[1] Blanke, M.; Kinnaert, M.; Lunze, J.; Staroswiecki, M.: Diagnosis and fault-tolerant control (Springer-Verlag, Berlin, GERMANY) 2006 DOI: 10.1007/978-1-84628-877-7

[2] Hrúz, B.; Zhou, M.: Modeling and control of discrete-event dynamic systems with Petri nets and other tool (Springer-Verlag, London, U.K.) 2007

[3] Campos, E.J.; Seatzu, C.; Xie, X.: Formal methods in manufacturing (CRC Press Taylor and Francis Group, Boca Raton, USA) 2014 DOI: 10.1201/b16529

[4] Zaytoon, J.; Lafortune, S.: Overview of fault diagnosis methods for discrete event systems, *Ann. Rev. Control* 2013 **37**(2), 308–320 DOI: 10.1016/j.arcontrol.2013.09.009

[5] Sampath, M.; Sengupta, R.; Lafortune, S.; Sinnamohideen, K.; Teneketzis, D.: Diagnosability of discrete-event systems, *IEEE Trans. Automat. Control* 1995 **40**(9), 1555–1575 DOI: 10.1109/9.412626

[6] Prock, J.: A new technique for fault detection using Petri nets, *Automatica* 1991 **27**(2), 239–245 DOI: 10.1016/0005-1098(91)90074-C

[7] Cabasino, M.P.; Giua, A.; Seatzu, C.: Fault detection for discrete event systems using petri nets with unobservable transitions, *Automatica* 2010 **46**(9), 1531–1539 DOI: 10.1016/j.automatica.2010.06.013

[8] Basile, F.; Chiacchio, P.; Tommasi, G.D.: Sufficient conditions for diagnosability of Petri nets, in *Proc. 9th Int. Workshop on Discrete Event Systems, WODES* (Göteborg, SWEDEN) pp. 370–375, 2008 DOI: 10.1109/WODES.2008.4605974

[9] Basile, F.; Chiacchio, P.; Tommasi, G.D.: An efficient approach for online diagnosis of discrete event systems, *IEEE Trans. Automat. Control* 2009 **54**(4), 748–759 DOI: 10.1109/TAC.2009.2014932

[10] Dotoli, M.; Fanti, M.P.; Mangini, A.M.; Ukovich, W.: Online fault detection in discrete event systems by Petri nets and integer linear programming, *Automatica* 2009 **45**(11), 2665–2672 DOI: 10.1016/j.automatica.2009.07.021

[11] Fanti, M.P.; Mangini, A.M.; Ukovich, W.: Fault Detection by Labeled Petri Nets and Time Constraints, *Proc. 3rd Int. Workshop on Dependable Control of Discrete Systems* (DCDS, Saarbrucken, GERMANY) pp. 168–173, 2011 DOI: 10.1109/DCDS.2011.5970336

[12] Lefebvre, D.; Aguayo-Lara, E.: Initial study for observers application to fault detection and isolation with continuous timed petri nets, *IFAC-PapersOnLine* 2015 **48**(7), 97–103 DOI: 10.1016/j.ifacol.2015.06.479

[13] Ramirez-Trevino, A.; Ruiz-Beltran, E.; Rivera-Rangel, I.; Lopez-Mellado, E.: Online fault diagnosis of discrete event systems. A Petri net-based approach, *IEEE Trans. Automation Sci. Engng.* 2007 **4**(1), 31–39 DOI: 10.1109/TASE.2006.872120

[14] Genc, S.; Lafortune, S.: Distributed diagnosis of place-bordered Petri nets, *IEEE Trans. Automation Sci. Engng.* 2007 **4**(2), 206–219 DOI: 10.1109/TASE.2006.879916

[15] Pencolé, Y.; Pichard, R.; Fernbach, P.: Modular fault diagnosis in discrete-event systems with a CPN diagnoser, *IFAC-PapersOnLine* 2015 **48**(21), 470–475 DOI: 10.1016/j.ifacol.2015.09.571

[16] Bouali, M.; Barger, P.; Schon, W.: Backward Reachability of colored Petri nets for systems diagnosis, *Reliability Engng. System Safety* 2012 **99**, 1–14 DOI: 10.1016/j.ress.2011.10.003

[17] Jensen, K.: Coloured Petri nets: Basic concepts, analysis methods and practical use (Springer-Verlag, Berlin, GERMANY) 1997 DOI: 10.1007/978-3-642-60794-3