DOI: 10.15439/2014F363
ACSIS, Vol. 2

Construction of Healthcare System Structure for Reliability Analysis

Miroslav Kvassay, Elena Zaitseva
University of Zilina,
Faculty of Management Science and Informatics
Zilina, Slovakia
Email: {miroslav.kvassay, elena.zaitseva}@fri.uniza.sk

Abstract—**The principal goal of application of information technologies in medicine is improvement of medical care. Modern healthcare systems have to guarantee perfect care of patient. Therefore, the healthcare has to be characterized by high reliability first of all and reliability analysis of such system is an important problem. Most of the reliability analysis methods suppose the investigation of the structure of a system. But the healthcare system consists of heterogeneous components, such as software, hardware, human factor and etc. The structure definition for such system is a complex problem. We propose original approach for the construction of a healthcare system structure based on the monitoring of the system behavior. The monitoring examples are interpreted as Direct Partial Logic Derivatives of the structure function of the healthcare system. The structure function defines the correlation between system components states and system performance level unambiguously.**

I. Introduction

THE applications of information technologies, artificial intelligent methods and systems allow to enhance the quality of the healthcare system. But it is not to obliterate medical errors. It is well-known that about 4–7% people die due to medical errors [1], [2]. The development of a healthcare system with high reliability is one of the ways for solving this problem. In general, the basic reliability concept is defined as the probability that a system will perform its intended function during a period of running time without any failure [1]. A fault is an erroneous state of the system. Although the definitions of a fault are different for different systems and in different situations, a fault is always an existing part of the system and it can be removed by correcting the erroneous part of the system. Therefore, the determination of the fault probability is an important problem that can be decided based on system structure analysis [1]–[3].

A healthcare system consists of some principal components from the point of view of reliability analysis. Two of them have been defined in paper [1]: equipment/device and human factor. Detail structure of human factor and human errors for healthcare system is presented in works [2] and [4]. The healthcare system structure includes three components (technical, human and administrative) in [5]. The technical component consists of two parts as hardware and software. The hardware includes two types of medical devices/equipment that are based on special and standards-based technologies according to [1]. For example, the first type includes the medical decision support system, system for integration electronic medical records or picture archiving communication systems. The second type consists of special medical devices and equipment that can be used for special operation only (as magnetic resonance imaging scanners, for example). The human component of the healthcare system models medical errors. The organization component of the system unites management aspects and maintenance of the healthcare system.

Therefore, healthcare systems consist of heterogeneous components that complicate the definition of the mathematical model of their structure.

Two types of mathematical models are used in reliability analysis: a Binary-State System (BSS) and a Multi-State System (MSS). A BSS allows defining only two states in system/component performance – perfect functioning (presented as state 1) and failed (represented by number 0). The mathematical model based on a BSS permits to describe and investigate two types of system behavior that are system failure and system repair [6]. However, real systems have not only two performance levels and, therefore, one of the main problems of a BSS is to define the boundary between situations in which the system can be regarded as working and situations when it is considered failed [7], [8]. A MSS allows avoiding this problem, because it makes possible defines more than only two states in system/component behavior. Therefore, MSSs permit to investigate the system degradation and improvement. This mathematical model can be used for the analysis and determination of conditions that cause the system fault.

The healthcare system is a complex system and its overall performance can have different levels. Therefore, MSSs are appropriate mathematical model for the representation and estimation of the healthcare system. Most methods for MSS

This work was supported by the grant of 7th RTD Framework Program No 610425 (RASimAs) and grant of Slovak Research and Development Agency SK-PL-0023-12.

978-83-60810-58-3/$25.00 © 2014, IEEE

reliability analysis suppose the structure function as one of initial data [8], [9]. This function defines the system performance level depending on its components states. But the definition of the system structure function with heterogeneous component is a complex problem [5], [8]. One of possible ways for the definition of such system structure function is monitoring the system behavior as the set of the system performance level changes depending on system component state change (Fig. 1).

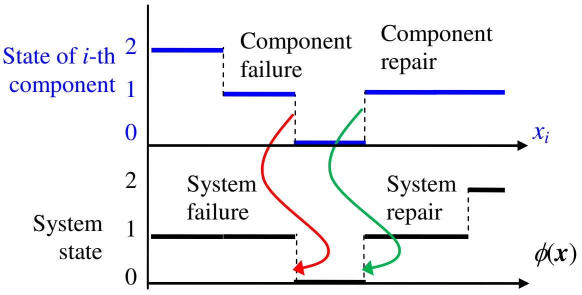

Fig. 1 System reliability interpretations for a MSS

In paper [10] such MSS performance level change depending on the component state change has been defined as Direct Partial Logic Derivative (DPLD) from mathematical point of view. The definition of the DPLD has been proposed in [11] for logic functions. According to this definition, a DPLD with respect to variable x_i allows investigation of the influence of the i-th variable value change on the change of the function value. The application of DPLDs in reliability analysis has been considered in papers [10], [12]. In this paper, we propose a new method for the construction of a MSS structure in form of the structure function by DPLDs. Application and approbation of the method is considered by the example of the typical healthcare system that has been defined in [2].

II. Mathematical Background

A. Structure Function

The correlation between system performance level and states of system components is defined by the structure function. Consider a system that consists of n components and the i-th system component can be in one of m_i states from 0 to m_i-1. Assume m performance levels for the system. Then the structure function of the system is defined as [7], [8]:

$$\phi(\boldsymbol{x}): \{0,\ldots,m_1-1\}\times\ldots\times\{0,\ldots,m_n-1\}\to\{0,\ldots,m-1\}, \quad (1)$$

where $\boldsymbol{x} = (x_1,\ldots, x_n)$ is a vector of system components states (state vector) and state 0 represent the total failure of the system/component while state m-1 (m_i-1) corresponds to perfect functioning of the system (the i-th component).

For a BSS, $m_1 = \ldots = m_n = m = 2$ in (1):

$$\phi(x_1,\ldots,x_n)=\phi(\boldsymbol{x}):\{0,1\}^n\to\{0,1\}. \quad (2)$$

According to definition (2), the BSS structure function can be interpreted as a Boolean function [6].

Every component of a MSS/BSS is characterized by probabilities of its states:

$$p_{i,s}=\Pr\{x_i=s\},\quad s\in\{0,\ldots,m_i-1\}. \quad (3)$$

Equations (1) and (3), and (2) and (3) constitute the overall definition of a steady-state MSS and BSS.

The typical property of many real systems is their coherency. A system is coherent if it meets the following assumptions [6]–[8]:

a) the system structure function is monotone, i.e. the failure (degradation) of any system component cannot causes the repair (improvement) of system performance level;

b) system components are s-independent, i.e. there is no correlation between states of different components.

Below, we consider only the coherent system.

B. Basic Reliability Characteristics of a System

The basic reliability characteristics of a system are availability and unavailability. The availability is the probability of the system functioning in the fixed time [6]. The unavailability is defined as the probability that the system is failed [6]. Using (2) and (3), the availability of a steady-state BSS can be computed as follows [6]:

$$A=\Pr\{\phi(\boldsymbol{x})=1\}, \quad (4)$$

and its unavailability in the following way [6]:

$$U=\Pr\{\phi(\boldsymbol{x})=0\}=1-A. \quad (5)$$

For a MSS, the availability is defined as probability that the system performance does not fall below given level and, therefore, it should be defined with respect to system performance level [8]:

$$A(j)=\Pr\{\phi(\boldsymbol{x})\geq j\},\quad j\in\{1,\ldots,m\}. \quad (6)$$

The system unavailability can be defined in the same manner as for a BSS [8]:

$$U=\Pr\{\phi(\boldsymbol{x})=0\}=1-A(1). \quad (7)$$

If we know the mean time between two consecutive breakdowns of a system, i.e. the Mean Time Between Failures (MTBF), then there can be derived other reliability characteristics using system availability and unavailability – the Mean Time To Failure (MTTF) and the Mean Time To Repair (MTTR).

For a BSS, these characteristics can be calculated as the following products [6], [13]:

$$MTTF=A\,MTBF, \quad (8)$$

$$MTTR=U\,MTBF. \quad (9)$$

Other application of the structure function in reliability estimation is importance analysis that can be used to identify the influence of system components on the system performance/availability. There has been proposed a lot of measures that can be used for this task. These measures are known as importance measures [8], [10], [14] and some of the most commonly used are the Structural Importance (SI), Birnbaum's Importance (BI), Criticality Importance (CI) and Fussell-Vesely Importance (FVI) (Table I).

TABLE I.
IMPORTANCE MEASURES

Importance measures	Meaning
SI	SI concentrates only on the topological structure of the system. It is defined as the relative number of situations in which a given component is critical for the system activity.
BI	BI of a given component is defined as the probability that the component is critical for the system work.
CI	CI of a given component is calculated as the probability that the system failure (degradation) has been caused by the component failure (deterioration) given that the system is failed (not perfect functional).
FVI	FVI of a given component is defined as the probability that the component contributes to the system failure (degradation) probability.

C. Logical Differential Calculus

Logical differential calculus is a special tool that has been developed to analyze dynamic properties of logic functions [11], [15]. In papers [10], [12], there was considered its using in reliability analysis of BSSs and MSSs, respectively. Direct Partial Logic Derivatives (DPLDs) are one of several instruments of logical differential calculus. They allow identifying situations in which the change of a logic variable value coincides with the change of analyzed logic function. In terms of reliability analysis, a DPLD allows finding correlation between component failure/repair and system failure/repair.

For a BSS, we define a DPLD as follows [12]:

$$\partial\phi(j \to \bar{j})/\partial x_i(s \to \bar{s}) = \begin{cases} 1, \text{ if } \phi(s_i, \boldsymbol{x}) = j \text{ and } \phi(\bar{s}_i, \boldsymbol{x}) = \bar{j} \\ 0, \text{ other} \end{cases}, \qquad (10)$$

where $\phi(a_i, \boldsymbol{x}) = \phi(x_1,\ldots, x_{i-1}, a, x_{i+1},\ldots, x_n)$ for $a \in \{s, \bar{s}\}$ and $s, j \in \{0, 1\}$.

There exist four DPLDs (10) with respect to the i-th variable and they have the following properties [10], [15]:

$$\begin{aligned} \partial\phi(1 \to 0)/\partial x_i(1 \to 0) &= \partial\phi(0 \to 1)/\partial x_i(0 \to 1), \\ \partial\phi(1 \to 0)/\partial x_i(0 \to 1) &= \partial\phi(0 \to 1)/\partial x_i(1 \to 0). \end{aligned} \qquad (11)$$

In the reliability analysis, DPLDs $\partial\phi(1 \to 0)/\partial x_i(1 \to 0)$ and $\partial\phi(0 \to 1)/\partial x_i(0 \to 1)$ can be used to discover situations in which the failure of a given component coincides with the system failure and situations when the repair of the i-th component leads into the repair of the system, respectively. Other two DPLDs make it possible to find situations in which the system failure correlates with the component repair ($\partial\phi(1 \to 0)/\partial x_i(0 \to 1)$) or when the system repair is caused by the component failure ($\partial\phi(0 \to 1)/\partial x_i(1 \to 0)$). However, in this paper only coherent systems are taken into account and therefore there exist no situation in which the component failure can cause system repair or vice versa, and, therefore, these two DPLDs have only zero values for a coherent BSS, which means that they are irrelevant from the reliability point of view [12], [13].

For a MSS, a DPLD with respect to variable i is defined in [10] as follows:

$$\partial\phi(j \to h)/\partial x_i(s \to r) = \begin{cases} 1, \text{ if } \phi(s_i, \boldsymbol{x}) = j \text{ and } \phi(r_i, \boldsymbol{x}) = h \\ 0, \text{ other} \end{cases}, \qquad (12)$$

where $\phi(a_i, \boldsymbol{x}) = \phi(x_1,\ldots, x_{i-1}, a, x_{i+1},\ldots, x_n)$ for $a \in \{s, r\}$; $s, r \in \{0,\ldots, m_i -1\}$, $s \neq r$ and $j, h \in \{0,\ldots, m -1\}$, $j \neq h$.

From the reliability point of view, the nonzero elements of DPLD (12) identifies situations in which the change of the state of the i-th component from value s to value r causes the change of the system performance level from j to h. Clearly, when $j > h$ and $s > r$ then DPLDs (12) can be used to find correlation between system degradation and component degradation, while for $j < h$ and $s < r$, DPLDs (12) discover situations in which component improvement results system improvement. Specially, when $j > h$ and $s < r$ or $j < h$ and $s > r$, DPLDs (12) discover coincidence between system degradation and component improvement or vice versa.

According to property a) of a coherent system, the degradation of any system component cannot cause system improvement and, therefore, there exist no DPLD (12) that would have nonzero values for $j < h$ and $s > r$ [16]. Property a) of a coherent system can also be interpreted as a statement that the improvement of any system component cannot cause system degradation and this implies that all DPLDs (12) for which $j > h$ and $s < r$ have only zero values [16]. So, the result is that only DPLDs (12) for which $j > h$ and $s > r$ or $j < h$ and $s < r$ can contain nonzero values and therefore only these two types of DPLDs (12) are important from the reliability point of view.

Another point is that we assume that a component of a MSS degrades gradually, i.e. step by step, which means that only the following DPLDs have to be investigated to find correlation between the system deterioration and component degradation [16]:

$$\partial\phi(j \to h)/\partial x_i(s \to s-1), \; s > 0 \text{ and } j > h. \qquad (13)$$

In paper [16], there has been proposed another type of a DPLD for a MSS that is named as DPLD union and that can be defined in the following way:

$$\begin{aligned} &\partial\phi(\downarrow j \downarrow)/\partial x_i(s \to s-1) = \\ &= \bigcup_{l=j}^{m-1}\left(\bigcup_{h=0}^{j-1} \partial\phi(l \to h)/\partial x_i(s \to s-1)\right) = \\ &= \begin{cases} 1 & \text{if } \phi(s_i, \boldsymbol{x}) \geq j \text{ and } \phi((s-1)_i, \boldsymbol{x}) < j \\ 0 & \text{other} \end{cases}. \end{aligned} \qquad (14)$$

This DPLD allows identifying the total influence of the degradation of the *i*-th system component on the system performance level, because it reveals situations in which the analyzed degradation of the component causes the deterioration of the system below specified level *j*.

However, in terms of maintenance, there exist more strategies on how to perform system improvement. Two basic approaches are minor improvement (by one state) and major improvement (by more than one state) [8]. A special type of major improvement is the fully improvement when the component is replaced by a totally new one. The consequences of the minor improvement can be modelled by the following DPLDs:

$$\partial\phi(j \to h)/\partial x_i(s \to s+1) \quad s < m_i - 1 \text{ and } j < h, \tag{15}$$

and results of the fully improvement as follows:

$$\partial\phi(j \to h)/\partial x_i(s \to m_i - 1), \quad s < m_i - 1 \text{ and } j < h. \tag{16}$$

In paper [16], the concept of DPLD union (14) was originally developed for the modelling of the system improvement caused by minor improvement of the *i*-th system component:

$$\begin{aligned} &\partial\phi(\uparrow j \uparrow)/\partial x_i(s \to s+1) = \\ &= \bigcup_{l=0}^{j-1}\left(\bigcup_{h=j}^{m-1} \partial\phi(l \to h)/\partial x_i(s \to s+1)\right) = \\ &= \begin{cases} 1 & \text{if } \phi(s_i, \boldsymbol{x}) < j \text{ and } \phi((s+1)_i, \boldsymbol{x}) \geq j \\ 0 & \text{other} \end{cases}. \end{aligned} \tag{17}$$

The union (17) of DPLDs has more informative value than DPLD (15), because it analyzes the total influence of the *i*-th component minor improvement on the system performance level *j*.

D. Minimal Path Vectors and Minimal Cut Vectors

Minimal Path Vectors (MPVs) and Minimal Cut Vectors (MCVs) are special types of state vectors. Firstly, consider two arbitrary state vectors $\boldsymbol{x} = (x_1,\dots, x_n)$ and $\boldsymbol{y} = (y_1,\dots, y_n)$. Then using notation $\boldsymbol{x} < \boldsymbol{y}$ means that $x_i \leq y_i$ for any $i \in \{1,\dots, n\}$ and there exists at least one i such that $x_i < y_i$.

For a BSS, a MPV represents such situation in which failure of any working component results system failure and a MCV correlates with a situation when the repair of any failed component leads into the system repair. So, a state vector $\boldsymbol{x}$ is a MPV if $\phi(\boldsymbol{x}) = 1$ and $\phi(\boldsymbol{y}) = 0$ for any $\boldsymbol{y} < \boldsymbol{x}$. Similarly, a state vector $\boldsymbol{x}$ is a MCV if $\phi(\boldsymbol{x}) = 0$ and $\phi(\boldsymbol{y}) = 1$ for any $\boldsymbol{y} > \boldsymbol{x}$.

In the case of MSSs, MPVs and MCVs have to be defined with regard to system performance level [8]. A MPV for a given performance level j (for $j = 1,\dots, m$ -1) of a MSS defines such situation in which degradation of any not-failed component by one state causes degradation of system performance to value less than the specified performance level j, or more formally, a state vector $\boldsymbol{x}$ is a MPV for system performance level j if $\phi(\boldsymbol{x}) \geq j$ and $\phi(\boldsymbol{y}) < j$ for any $\boldsymbol{y} < \boldsymbol{x}$ [8], [16]. In the contrast to a MPV, a MCV for a given performance level of a MSS represents such case when improvement of any not perfect working component by one state results system improvement at least to performance level j (for $j = 1,\dots, m$ -1), i.e. a state vector $\boldsymbol{x}$ is a MCV for system performance level j if $\phi(\boldsymbol{x}) < j$ and $\phi(\boldsymbol{y}) \geq j$ for any $\boldsymbol{y} > \boldsymbol{x}$ [8], [16].

In papers [16], [17], the calculation of MCVs based on DPLDs has been proposed. For a BSS, there was shown that all MCVs can be computed as the special intersection of all modified (extended) DPLDs $\partial_e\phi(0 \to 1)/\partial_e x_i(0 \to 1)$, i.e. for all $i \in \{1,\dots, n\}$. The extended DPLD can be derived from DPLD (10) as follows:

$$\begin{aligned} &\partial_e\phi(j \to \bar{j})/\partial_e x_i(s \to \bar{s}) = \\ &= \begin{cases} 1 & \text{if } x_i = s \text{ and } \phi(s_i, \boldsymbol{x}) = j \text{ and } \phi(\bar{s}_i, \boldsymbol{x}) = \bar{j} \\ 0 & \text{if } x_i = s \text{ and } \phi(s_i, \boldsymbol{x}) = \phi(\bar{s}_i, \boldsymbol{x}) \\ * & \text{if } x_i \neq s \end{cases}, \end{aligned} \tag{18}$$

where symbol "*" denotes situations for which DPLD (10) is not defined, i.e. situation when the studied component is in state $\bar{s}$.

The intersection of two modified DPLDs defines situation in which the state change of at least one component causes the required change of the system state. This intersection is defined in Table II.

TABLE II.
THE INTERSECTION OF TWO MODIFIED DPLDS

		$\partial_e\phi(j \to \bar{j})/\partial_e x_i(s \to \bar{s})$		
		*	0	1
$\partial_e\phi(j \to \bar{j})/\partial_e x_k(s \to \bar{s})$	*	*	0	1
	0	0	0	0
	1	1	0	1

MPVs of a BSS can be computed similarly, but we use the extended DPLDs $\partial_e\phi(1 \to 0)/\partial_e x_i(1 \to 0)$ instead of the previous one. Then value 1 in the intersection of all considered extended DPLDs identifies state vectors that are MPVs.

The relation between MCVs and DPLDs, defined in paper [17], was generalized for MSSs in paper [16]. This generalization is based on the using of other types of DPLDs that has been named as the extended union of DPLDs and the merge of extended unions. Using union (17) of DPLDs, the merge of extended unions is defined as follows:

$$\begin{aligned} &\partial_e\phi(\uparrow j \uparrow)/\partial_e x_i = \\ &= \begin{cases} \partial\phi(\uparrow j \uparrow)/\partial x_i(s \to s+1) & \text{if } x_i < m_i - 1 \\ * & \text{if } x_i = m_i - 1 \end{cases}, \end{aligned} \tag{19}$$

and it identifies all situations in which the change of system component state by one value causes the transition of the system performance from level less than j to level greater than or equal to j. Therefore, the intersection of merges (19) for given system performance level *j* and for all components identifies all MCVs for system performance level *j*. The intersection of two merges (19) is defined same as the intersection of two extended DPLDs (Table II).

MPVs for given system performance level *j* can be computed same as its MCVs, but, instead of merge (19), we use the merge of extended unions that is based on DPLDs (13) and (14) and we denote it as follows:

$$\partial_e\phi(\downarrow j\downarrow)/\partial_e x_i = \\ = \begin{cases} \partial\phi(\downarrow j\downarrow)/\partial x_i(s\to s-1) & \text{if } x_i>0 \\ * & \text{if } x_i=0 \end{cases}. \qquad (20)$$

Value 1 in the intersection of all merges (20) for given system performance level *j* identifies all MPVs for the considered system performance level.

III. Revelation of the System Structure from Direct Partial Logic Derivatives

The structure function is very important in reliability analysis. However, there exist situations that the structure function of the system is unknown and only the consequences of the system components failure/degradation or repair/improvement on the system performance level are known. In these cases, the structure function of the system has to be discovered to find some reliability characteristics of the system. When we know the results of individual system components changes on the system work, then it means that individual DPLDs are known for the system. Therefore, in the next part, we assume that DPLDs are recognized for the analyzed system and we want to identify its structure function from them.

A. *Discovering the Structure Function of a Binary-State System*

The structure function (2) of a BSS is formally identical with the definition of a Boolean function. Moreover, the structure function of a coherent BSS can be interpreted as a monotonic Boolean function. Therefore, some concepts of Boolean algebra can be used to find the structure function of a BSS. One of these concepts is the idea that every Boolean function can be expressed unambiguously in the form of minimal disjunctive (conjunctive) normal form. Moreover, for a monotonic Boolean function, there exists only one minimal disjunctive (conjunctive) normal form [18]. When we want to find minimal disjunctive (conjunctive) normal form of a monotonic Boolean function, then all prime implicants (implicates) of this function have to be found.

An implicant of a monotonic Boolean function is a set of logic variables whose simultaneous true values imply that the monotonic Boolean function has true value. A prime implicant is an implicant from which no variable can be removed without losing its status as implicant [18]–[20]. In terms of reliability analysis, an implicant can be interpreted as a set of components whose simultaneous work ensures that the system will work. Also, in terms of reliability analysis, an implicant is known as a path set and prime implicant as a Minimal Path Set (MPS) [21].

An implicate of a monotonic Boolean function is a set of logic variables whose simultaneous false values imply that the considered function has false value [18]. In reliability analysis, the definition of implicate corresponds to cut set and the definition of prime implicate coincides with the definition of Minimal Cut Set (MCS). So, a cut set is a set of components whose simultaneous failure causes the failure of the system and it is minimal if no component can be removed from it without losing its status as a cut set [21].

According to the previous paragraphs, the revelation of MPSs or MCSs of the analyzed BSS implies discovering the system structure function. However, in this paper, we assume that the structure function is defined in terms of vectors and not in terms of set theory. Therefore, MPSs (MCSs) have to be transformed into the form of state vectors. This can be done by using the relation between prime implicants (implicates) and MPSs (MCSs). Every prime implicant (implicate) can be transformed in the vector form that is known as minimal true vector for prime implicant and maximal false vector for prime implicate, respectively [22]. In the terms of reliability engineering, a minimal true vector corresponds to MPV while maximal false vector agrees with a MCV. Therefore, based on paper [22], there exists one-to-one correspondence between MPSs (MCSs) and MPVs (MCVs), i.e. a MPV (MCV) corresponds to a MPS (MCS) in terms of state vectors. This implies that the structure function of any coherent BSS can be defined via MPVs or MCVs unambiguously. So, if DPLDs $\partial\phi(1\to 0)/\partial x_i(1\to 0)$ for all $i\in\{1,\dots,n\}$ are given, then we can formulate the next algorithm for finding the structure function of a BSS:

1. Derive the extended DPLDs $\partial_e\phi(1\to 0)/\partial_e x_i(1\to 0)$ from $\partial\phi(1\to 0)/\partial x_i(1\to 0)$ for every component, i.e. for $i\in\{1,\dots,n\}$ according to (18).
2. Compute the intersection of all extended DPLDs from the previous step based on the rules in Table II.
3. Define MPVs that agree to value 1 in the intersection, calculated in the previous step.
4. According to the definition of a MPV (section II.D), define the structure function of the system as follows:
   a. if an arbitrary state vector $\boldsymbol{y}$ meets the condition $\boldsymbol{y}\geq\boldsymbol{x}$ at least for one MPV $\boldsymbol{x}$, then $\phi(\boldsymbol{y}) = 1$;
   b. if an arbitrary state vector $\boldsymbol{y}$ meets the condition $\boldsymbol{y}<\boldsymbol{x}$ at least for one MPV $\boldsymbol{x}$, then $\phi(\boldsymbol{y}) = 0$.

Another approach is based on MCVs. In this case, we assume DPLDs $\partial\phi(0\to 1)/\partial x_i(0\to 1)$ for all $i\in\{1,\dots,n\}$ are known (by the way, according to (11), they are identical to $\partial\phi(1\to 0)/\partial x_i(1\to 0)$) and then we can reveal the structure function of the coherent BSS in the following way:

1. Derive the extended DPLDs $\partial_e\phi(0\to 1)/\partial_e x_i(0\to 1)$ from $\partial\phi(0\to 1)/\partial x_i(0\to 1)$ for every component, i.e. for $i\in\{1,\dots,n\}$ according to (18).

2. According to Table II, compute the intersection of all extended DPLDs from the previous step.
3. Define MCVs that agree to value 1 in the intersection, calculated in the previous step.
4. According to the definition of a MCV (section II.D), define the structure function of the system as follows:
   a. if an arbitrary state vector $\boldsymbol{y}$ meets the condition $\boldsymbol{y} \leq \boldsymbol{x}$ at least for one MCV $\boldsymbol{x}$, then $\phi(\boldsymbol{y}) = 0$;
   b. if an arbitrary state vector $\boldsymbol{y}$ meets the condition $\boldsymbol{y} > \boldsymbol{x}$ at least for one MCV $\boldsymbol{x}$, then $\phi(\boldsymbol{y}) = 1$.

B. Discovering the Structure Function of a Multi-State System

For MSSs, the structure function (1) is a multiple-valued function. Specially, when $m_1 = \ldots = m_n = m$ in definition (1), then the structure function can be interpreted as a Multiple-Valued Logic (MVL) function. In a case of MVL functions and also multiple-valued functions, there exist some normal forms that are equivalent to minimal disjunctive (conjunctive) normal form of Boolean functions. These forms are known as minimal Sum-of-Products (SoP) and minimal Products-of-Sum (PoS). By the way, a minimal disjunctive (conjunctive) normal form of a Boolean function is sometimes denoted as minimal SoP (PoS), however, in this paper, we used term minimal disjunctive (conjunctive) normal form when we deal with Boolean functions and minimal SoP (PoS) in a case of multiple-valued functions.

There exist several approaches for definition of SoP (PoS) [23], but definitions based on vector approach are the best ones for our work. According to this approach, there exist two important types of vectors for monotone multiple-valued functions: lower vectors (or lower 9 (boundary) points) and upper vectors (or upper (boundary) points) [24].

Lower points for value j of a monotone multiple-valued function of m values are minimal vectors for which the function has value j, for $j = 1,\ldots, m$ -1. When all lower points for every value of a monotone multiple-valued function are known, then the function has value j for an arbitrary vector $\boldsymbol{x}$ if there exists at least one lower point for level j that is lower than or equal to $\boldsymbol{x}$ and at least one lower point for level $j+1$ (given that $j < m$ -1) that is greater than $\boldsymbol{x}$. (A vector $\boldsymbol{x}$ is lower than a vector $\boldsymbol{y}$ if $\boldsymbol{x} < \boldsymbol{y}$ (see section II.D) and greater than $\boldsymbol{y}$ if $\boldsymbol{x} > \boldsymbol{y}$.)

Upper points for value j of a monotone multiple-valued function of m values are maximal vectors for which the function has value less than or equal to j, for $j = 0,\ldots, m$ -2. Therefore, if we know all upper points for every value of the considered function, then the function has value j for an arbitrary vector $\boldsymbol{x}$ if there exists at least one upper point for level j that is greater than or equal to $\boldsymbol{x}$ and at least one upper point for level j -1 (given that $j > 0$) that is lower than $\boldsymbol{x}$.

According to the previous paragraphs, every monotonic multiple-valued function can be defined by using all its upper points or all its lower points. A lower point for value j (for $j = 1,\ldots, m$ -1) of a monotonic multiple-valued function $\phi(\boldsymbol{x})$ is defined as vector $\boldsymbol{x}$ that meets the following conditions [15], [24], [25]:

a) $\phi(\boldsymbol{x}) \geq j$,

b) $\phi(\boldsymbol{y}) < j$ for any $\boldsymbol{y} < \boldsymbol{x}$.

This definition is same as the definition of a MPV of a coherent MSS (section II.D). This fact implies that the structure function of any coherent MSS can be defined via MPVs unambiguously.

An upper point for value j (for $j = 0,\ldots, m$ -2) of a monotonic multiple-valued function $\phi(\boldsymbol{x})$ is defined as vector $\boldsymbol{x}$ that meets the following conditions [15], [24], [25]:

a) $\phi(\boldsymbol{x}) \leq j$,

b) $\phi(\boldsymbol{y}) > j$ for any $\boldsymbol{y} > \boldsymbol{x}$.

The definition of an upper point is not equal to the MCV definition of a coherent MSS (section II.D). This is caused by the fact that upper points are defined for values $j = 0,\ldots, m$ -2, while MCVs are defined for $j = 1,\ldots, m$ -1. Therefore, assume that we do substitution $j+1 = l$ in the definition of the upper point. Due to the substitution, the upper point $\boldsymbol{x}$ for $l = 1,\ldots, m$ -1 has the following properties:

a) $\phi(\boldsymbol{x}) < l$,

b) $\phi(\boldsymbol{y}) \geq l$ for any $\boldsymbol{y} > \boldsymbol{x}$.

These properties are same as the properties of a MCV for level l of a MSS defined by the considered monotonic multiple-valued function. This implies that an upper point for value j of a monotonic multiple-valued function corresponds to a MCV for value $j+1$ of the MSS defined by the considered function. Therefore, there exists one-to-one relation between upper points for level j and MCVs for level $j+1$ of a MSS. So, the knowledge of all MCVs allows defining the structure function of the considered MSS.

According to the aforementioned text, we only need to find all MPVs (MCVs) of the analyzed MSS to reveal the system structure. So, if the consequences of degradation of any component on the system performance are known, i.e. DPLDs $\partial\phi(\downarrow j \downarrow)/\partial x_i(s \to s-1)$ for all $i \in \{1,\ldots, n\}$, $s \in \{1,\ldots, m_i$ -1$\}$ and $j \in \{1,\ldots, m$ -1$\}$ are defined, then we can formulate the following algorithm for discovering the system structure function:

1. Derive the merge of extended unions $\partial_e\phi(\downarrow j \downarrow)/\partial_e x_i$ from $\partial\phi(\downarrow j \downarrow)/\partial x_i(s \to s-1)$ for fixed performance level j and for every system component, i.e. for $i \in \{1,\ldots, n\}$ according to (20).
2. According to Table II, compute the intersection of all merges (20) of extended unions from the previous step.
3. Define MPVs for given system performance level j that agree to value 1 in the intersection, calculated in the previous step.
4. Repeat steps 1. – 3. for all relevant system performance levels, i.e. for $j \in \{1,\ldots, m$ -1$\}$.
5. According to the definition of a MPV of a MSS (section II.D), define the value of the system structure function for an arbitrary state vector $\boldsymbol{y}$ as follows:

- if the state vector $\boldsymbol{y}$ meets the condition $\boldsymbol{y} \geq \boldsymbol{x}$ at least for one MPV $\boldsymbol{x}$ for system performance level m -1, then $\phi(\boldsymbol{y}) = m$ -1;
- else if the state vector $\boldsymbol{y}$ meets the condition $\boldsymbol{y} \geq \boldsymbol{x}$ at least for one MPV $\boldsymbol{x}$ for system performance level m -2, then $\phi(\boldsymbol{y}) = m$ -2;

  …
- else if the state vector $\boldsymbol{y}$ meets the condition $\boldsymbol{y} \geq \boldsymbol{x}$ at least for one MPV $\boldsymbol{x}$ for system performance level 1, then $\phi(\boldsymbol{y}) = 1$;
- else $\phi(\boldsymbol{y}) = 0$.

Similarly as in the case of BSSs, the previous algorithm can be reformulated in the terms of MCVs. In this situation, we assume that all DPLDs $\partial\phi(\uparrow j \uparrow)/\partial x_i(s \to s+1)$ for $i \in \{1,\ldots, n\}$, $s \in \{0,\ldots, m_i\text{-}2\}$ and $j \in \{1,\ldots, m\text{-}1\}$ are defined, i.e. the influence of minor improvement of any component on system performance level is known. In this case, the following algorithm can be formulated:

1. Derive the merge of extended unions $\partial_e\phi(\uparrow j \uparrow)/\partial_e x_i$ from $\partial\phi(\uparrow j \uparrow)/\partial x_i(s \to s+1)$ for fixed performance level j and for every component, i.e. for $i \in \{1,\ldots, n\}$ according to (19).
2. According to Table II, compute the intersection of all merges (19) of extended unions that were gained in the previous step.
3. Define MCVs for given system performance level j that agree to value 1 in the intersection, calculated in the previous step.
4. Repeat steps 1. – 3. for all relevant system performance levels, i.e. for $j \in \{1,\ldots, m\text{-}1\}$.
5. According to the definition of a MCV of a MSS (section II.D), define the value of the system structure function for an arbitrary state vector $\boldsymbol{y}$ as follows:
    - if there is a MCV $\boldsymbol{x}$ for system performance level 1 that meets the condition $\boldsymbol{y} \leq \boldsymbol{x}$, then $\phi(\boldsymbol{y}) = 0$;
    - else if there is a MCV $\boldsymbol{x}$ for system performance level 2 that meets the condition $\boldsymbol{y} \leq \boldsymbol{x}$, then $\phi(\boldsymbol{y}) = 1$;

      …
    - else if there is a MCV $\boldsymbol{x}$ for system performance level m -1 that meets the condition $\boldsymbol{y} \leq \boldsymbol{x}$, then $\phi(\boldsymbol{y}) = m$ -2;
    - else $\phi(\boldsymbol{y}) = m$ -1.

IV. Reliability Analysis of a Healthcare System with Unknown Structure

Consider the human module of the health care system from book [2]. This module is formed by two persons – a doctor and a nurse, and it defines the consequences of the wrong doctor and nurse behavior on a patient health. In the terms of reliability analysis, the nurse and doctor can be interpreted as two independents modules of the analyzed system. The nurse can perform three types of errors and doctor can also make three types of bad decisions. The wrong decisions are caused by facts that are defined in Table III. These decisions can be interpreted as independent components of the human module and their occurrence can caused the human module degradation that is defined as the deterioration of the patient health.

Table III.
Components of the Human Module of the Health Care System

System modules	System components	$p_{i,0}$	$p_{i,1}$
Nurse	Correct interpretation of doctor's instructions, x_1	0.01	0.99
	Good work environment, x_2	0.02	0.98
	Not-haste, x_3	0.03	0.97
Doctor	Correct diagnosis, x_4	0.04	0.96
	Good Surroundings, x_5	0.06	0.94
	Not-haste, x_6	0.05	0.95

According to Table III, every component of the human module has two performance levels – failed (an error has occurred) and functioning (a problem has not occurred).

In Table IV, there are defined performance levels of the human module.

Table IV.
Human Module Performance Levels

System performance levels	Interpretation
0	Patient received an inadequate amount of wrong medication
1	Patient received an inadequate amount of correct medication
2	Patient received a safe amount of incorrect medication
3	Patient received an adequate amount of correct medication

The results of failures of individual components of the human module on the patient health are defined by DPLDs $\partial\phi(\downarrow j \downarrow)/\partial x_i(1 \to 0)$, for $i \in \{1,\ldots, 6\}$ and $j \in \{1, 2, 3\}$, in Table V. For example, the nonzero element (1,0,0,0,0,0) of DPLD $\partial\phi(\downarrow 1 \downarrow)/\partial x_1(1 \to 0)$ means that the failure of the first component causes the total failure of the analyzed system in situation when all other components are failed.

Now, we want to find the structure function of the human module. According to the previous section, we need to find all MPVs of the considered module. For this task, the first algorithm from section III.B can be used.

In the first step, the merges $\partial_e\phi(\downarrow j \downarrow)/\partial_e x_i$ of extended unions have to be derived from DPLDs defined in Table V. For example, the merges $\partial_e\phi(\downarrow 1 \downarrow)/\partial_e x_i(1 \to 0)$ are calculated in Table IX (white columns). In the next step, their intersection has to be computed to identify MPVs for system performance level 1 (gray columns in Table IX). This procedure has to be repeated for other relevant performance levels of the system, i.e. for $j = 2, 3$. After that, all MPVs of the system are known (Table VI) and we can reveal the system structure function according to rules defined in the last step of the used algorithm. The discovered structure function of the system is presented in Table VII.

TABLE V.
THE CONSEQUENCES OF COMPONENTS FAILURES ON THE PATIENT HEALTH DEFINED BY DPLDS

Component (i)	**System performance level** (j)	**Nonzero elements of DPLD** $\partial\phi(\downarrow j\downarrow)/\partial x_i(1\rightarrow 0)$
1	1	(1,0,0,0,0,0)
	2	(1,0,0,0,1,1) (1,0,0,1,0,1) (1,0,0,1,1,0) (1,0,0,1,1,1) (1,1,0,1,0,0) (1,1,1,0,0,0) (1,1,1,0,0,1) (1,1,1,0,1,0) (1,1,1,1,0,0)
	3	(1,0,1,1,1,1) (1,1,0,1,1,1) (1,1,1,0,1,1) (1,1,1,1,0,1) (1,1,1,1,1,0)
2	1	(0,1,0,0,0,0)
	2	(0,1,0,0,1,1) (0,1,0,1,1,0) (0,1,0,1,1,0) (0,1,0,1,1,1) (1,1,0,1,0,0) (1,1,1,0,0,0) (1,1,1,0,0,1) (1,1,1,0,1,0) (1,1,1,1,0,0)
	3	(0,1,1,1,1,1) (1,1,0,1,1,1) (1,1,1,0,1,1) (1,1,1,1,0,1) (1,1,1,1,1,0)
3	1	(0,0,1,0,0,0)
	2	(0,0,1,0,1,1) (0,0,1,1,0,1) (0,0,1,1,1,0) (0,0,1,1,1,1) (1,1,1,0,0,0) (1,1,1,0,0,1) (1,1,1,0,1,0)
	3	(0,1,1,1,1,1) (1,0,1,1,1,1) (1,1,1,0,1,1) (1,1,1,1,0,1) (1,1,1,1,1,0)
4	1	(0,0,0,1,0,0)
	2	(0,0,1,1,0,1) (0,0,1,1,1,0) (0,1,0,1,0,1) (0,1,0,1,1,0) (0,1,1,1,0,1) (0,1,1,1,1,0) (1,0,0,1,0,1) (1,0,0,1,1,0) (1,0,1,1,0,1) (1,0,1,1,1,0) (1,1,0,1,0,0) (1,1,0,1,0,1) (1,1,0,1,1,0)
	3	(0,1,1,1,1,1) (1,0,1,1,1,1) (1,1,0,1,1,1) (1,1,1,1,0,1) (1,1,1,1,1,0)
5	1	(0,0,0,0,1,0)
	2	(0,0,1,0,1,1) (0,0,1,1,1,0) (0,1,0,0,1,1) (0,1,0,1,1,0) (0,1,1,0,1,1) (0,1,1,1,1,0) (1,0,0,0,1,1) (1,0,0,1,1,0) (1,0,1,0,1,1) (1,0,1,1,1,0) (1,1,0,0,1,1)
	3	(0,1,1,1,1,1) (1,0,1,1,1,1) (1,1,0,1,1,1) (1,1,1,0,1,1) (1,1,1,1,1,0)
6	1	(0,0,0,0,0,1)
	2	(0,0,1,0,1,1) (0,0,1,1,0,1) (0,1,0,0,1,1) (0,1,0,1,0,1) (0,1,1,0,1,1) (0,1,1,1,0,1) (1,0,0,0,1,0) (1,0,0,1,0,1) (1,0,1,0,1,1) (1,0,1,1,0,0) (1,1,0,0,1,1)
	3	(0,1,1,1,1,1) (1,0,1,1,1,1) (1,1,0,1,1,1) (1,1,1,0,1,1) (1,1,1,1,0,1)

TABLE VI.
THE MPVS OF THE CONSIDERED HUMAN MODULE

System performance level	**MPVs**
1	(0,0,0,0,0,1) (0,0,0,0,1,0) (0,0,0,1,0,0) (0,0,1,0,0,0) (0,1,0,0,0,0) (1,0,0,0,0,0)
2	(0,0,1,0,1,1) (0,0,1,1,0,1) (0,0,1,1,1,0) (0,1,0,0,1,1) (0,1,0,1,0,1) (0,1,0,1,1,0) (1,0,0,0,1,1) (1,0,0,1,0,1) (1,0,0,1,1,0) (1,1,0,1,0,0) (1,1,1,0,0,0)
3	(0,1,1,1,1,1) (1,0,1,1,1,1) (1,1,0,1,1,1) (1,1,1,0,1,1) (1,1,1,1,0,1) (1,1,1,1,1,0)

Now, we can use the revealed structure function with combination of data from Table III for computation of system availability (6) and unavailability (7). The final values are in Table VIII. According to results in Table VIII, there is very little probability that the considered human module of a health care system will failed.

TABLE VII.
THE STRUCTURE FUNCTION OF THE CONSIDERED HUMAN MODULE

	$x_1\,x_2\,x_3$							
$x_4\,x_5\,x_6$	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
0 0 0	0	1	1	1	1	1	1	2
0 0 1	1	1	1	1	1	1	1	2
0 1 0	1	1	1	1	1	1	1	2
0 1 1	1	2	2	2	2	2	2	3
1 0 0	1	1	1	1	1	1	2	2
1 0 1	1	2	2	2	2	2	2	3
1 1 0	1	2	2	2	2	2	2	3
1 1 1	1	2	2	3	2	3	3	3

TABLE VIII.
AVAILABILITY AND UNAVAILABILITY OF THE CONSIDERED HUMAN MODULE

System performance level (j)	$A(j)$	U
0	-	7.2e-10
1	9.9999e-1	-
2	9.9966e-1	-
3	9.8392e-1	-

V. CONCLUSION

In this paper, a new method for the construction of the structure function based on the DPLDs is considered. This method can be used for the design of the system mathematical model in a case when the initial system has complex structure and correlation between components is not clear defined. The monitoring result of the initial system behavior is interpreted as the set of system performance changes depending on the changes of fixed system component states and these sets are collected for all system components. According to the definition of a DPLD, such sets are interpreted as DPLDs of the structure function. According to the proposed method based on DPLDs, the structure function (1) or (2) of MSS or BSS can be constructed. Then, numerous reliability indices and measures presented in section II.B can be calculated to investigate the initial system.

REFERENCES

[1] B. S. Dhillon, *Medical Device Reliability and Associated Areas*. Boca Raton FLA: CRC Press, 2000, 240 p.

[2] B. S. Dhillon, *Human Reliability and Error in Medical System*. Singapore: World Scientific, 2003, 232 p.

[3] E. Zio, “Reliability engineering: Old problems and new challenges,” *Reliability Engineering & System Safety*, vol. 94, no. 2, pp. 125–141, Feb. 2009, http://dx.doi.org/10.1016/j.ress.2008.06.002.

[4] M. Lyons, S. Adams, M. Woloshynowych and Ch. Vincent, “Human reliability analysis in healthcare: A review of techniques,” *International Journal of Risk & Safety in Medicine*, vol. 16, no. 4, pp. 223–237, Jan. 2004.

[5] E. Zaitseva and M. Rusin, "Healthcare system representation and estimation based on viewpoint of reliability analysis," *Journal of Medical Imaging and Health Informatics*, vol. 2, no. 1, pp. 80–86, March 2012, http://dx.doi.org/10.1166/jmihi.2012.1067.

[6] M. Xie, Y.-S. Dai and K.-L. Poh, *Computing System Reliability. Models and Analysis.* New York, NY: Kluwer Academic Publishers, 2004, 293 p.

[7] B. Natvig, *Multistate Systems Reliability Theory with Applications*. New York, NY: Wiley, 2011, 262 p., http://dx.doi.org/10.1002/9780470977088.

[8] A. Lisnianski and G. Levitin, *Multi-state System Reliability. Assessment, Optimization and Applications*. Singapore: World Scientific, 2003, 376 p.

[9] A. Lisnianski, I. Frenkel and Y. Ding, *Multi-state System Reliability Analysis and Optimization for Engineers and Industrial Managers*. London, UK: Springer-Verlag London Ltd., 2010, 393 p., http://dx.doi.org/10.1007/978-1-84996-320-6.

[10] E. Zaitseva and V. Levashenko, "Multiple-Valued Logic mathematical approaches for multi-state system reliability analysis," *Journal of Applied Logic*, vol. 11, no. 3, pp. 350–362, Special Issue, 2013, http://dx.doi.org/10.1016/j.jal.2013.05.005.

[11] M. A. Tapia, T. A. Guima and A. Katbab, "Calculus for a multivalued-logic algebraic system," *Applied Mathematics & Computation*, vol. 42, no. 3, pp. 255–285, April 1991, http://dx.doi.org/10.1016/0096-3003(91)90004-7.

[12] E. N. Zaitseva and V. G. Levashenko, "Importance analysis by logical differential calculus," *Automation and Remote Control*, vol. 74, no. 2, pp. 171–182, Feb. 2013, http://dx.doi.org/10.1134/S000511791302001X.

[13] W. G. Schneeweiss, "A short Boolean derivation of mean failure frequency for any (also non-coherent) system," *Reliability Engineering & System Safety*, vol. 94, no. 8, pp. 1363–1367, Aug. 2009, http://dx.doi.org/10.1016/j.ress.2008.12.001.

[14] W. Kuo and X. Zhu, *Importance Measures in Reliability, Risk, and Optimization*. Chichester, UK: John Wiley & Sons, Ltd, 2012, 472 p., http://dx.doi.org/10.1002/9781118314593.

[15] S. N. Yanushkevich, D. M. Miller, V. P. Shmerko and R. S. Stankovic, *Decision Diagram Techniques for Micro- and Nanoelectronic Design. Handbook*. Boca Raton, FL: CRC Press, 2006, 952 p.

[16] M. Kvassay, E. Zaitseva, V. Levashenko and J. Kostolny, "Minimal cut vectors and logical differential calculus," in *Proc. IEEE 44th International Symposium on Multiple-Valued Logic (ISMVL) 2014*, pp. 167–172, http://dx.doi.org/10.1109/ISMVL.2014.37.

[17] E. Zaitseva, J. Kostolny, M. Kvassay, V. Levashenko and K. Pancerz, "Failure analysis and estimation of the healthcare system," in *Proc. Federated Conference on Computer Science and Information Systems (FedCSIS) 2013,* pp. 235–240.

[18] T. Eiter, K. Makino and G. Gottlob, "Computational aspects of monotone dualization: A brief survey," *Discrete Applied Mathematics*, vol. 156, no. 11, pp. 2035–2049, June 2008, http://dx.doi.org/10.1016/j.dam.2007.04.017.

[19] R. B. Cutler and S. Muroga, "Derivation of minimal sums for completely specified functions," *IEEE Transactions on Computers*, vol. C-36, no. 3, pp. 277–292, March 1987, http://dx.doi.org/10.1109/TC.1987.1676900.

[20] P. Jain and G. Gopalakrishnan, "Efficient symbolic simulation-based verification using the parametric form of boolean expressions," *IEEE Transactions On Computer-Aidede Design of Integrated Circuits and Sysytem*, vol. 13, no. 8, pp. 1005–1015, Aug. 1994, http://dx.doi.org/10.1109/43.298036.

[21] M. Rausand and A. Høyland, *System Reliability Theory: Models, Statistical Methods, and Applications*. Haboken, NJ: John Wiley & Sons, Inc., 2004, 664 p.

[22] V. Gurvich and L. Khachiyan, "On generating the irredundant conjunctive and disjunctive normal forms of monotone Boolean functions," *Discrete Applied Mathematics*, vol. 96–97, pp. 363–373, , Oct. 1999, http://dx.doi.org/10.1016/S0166-218X(99)00099-2.

[23] K. Nakashima, Y. Nakamura and N. Takagi, "Logic expressions of monotonic multiple-valued functions," in *Proc. IEEE 26th International Symposium on Multiple-Valued Logic (ISMVL) 1996*, pp. 290–295, http://dx.doi.org/10.1109/ISMVL.1996.508370.

[24] R. A. Boedigheimer and K. C. Kapur, "Customer-driven reliability models for multistate coherent systems," *IEEE Transactions on Reliability,* vol. 43, no. 1, pp. 46–50, March 1994, http://dx.doi.org/10.1109/24.285107.

[25] J. C. Hudson and K. C. Kapur, "Modules in coherent multistate systems," *IEEE Transactions on Reliability,* vol. R-32, no. 2, pp. 183–185, June 1983, http://dx.doi.org/10.1109/TR.1983.5221522.

TABLE IX

THE MERGE (20) OF EXTENDED DPLDS UNIONS FOR LEVEL 1 OF THE CONSIDERED HUMAN MODULE

	$x_1\,x_2\,x_3$															
$x_4\,x_5\,x_6$	0 0 0		0 0 1		0 1 0		0 1 1		1 0 0		1 0 1		1 1 0		1 1 1	
0 0 0	* * * * * *	*	* * 1 * * *	**1**	* 1 * * * *	**1**	* 0 0 * * *	0	1 * * * * *	**1**	0 * 0 * * *	0	0 0 * * * *	0	0 0 0 * * *	0
0 0 1	* * * * * 1	**1**	* * 0 * * 0	0	* 0 * * * 0	0	* 0 0 * * 0	0	0 * * * * 0	0	0 * 0 * * 0	0	0 0 * * * 0	0	0 0 0 * * 0	0
0 1 0	* * * * 1 *	**1**	* * 0 * 0 *	0	* 0 * * 0 *	0	* 0 0 * 0 *	0	0 * * * 0 *	0	0 * 0 * 0 *	0	0 0 * * 0 *	0	0 0 0 * 0 *	0
0 1 1	* * * * 0 0	0	* * 0 * 0 0	0	* 0 * * 0 0	0	* 0 0 * 0 0	0	0 * * * 0 0	0	0 * 0 * 0 0	0	0 0 * * 0 0	0	0 0 0 * 0 0	0
1 0 0	* * * 1 * *	**1**	* * 0 0 * *	0	* 0 * 0 * *	0	* 0 0 0 * *	0	0 * * 0 * *	0	0 * 0 0 * *	0	0 0 * 0 * *	0	0 0 0 0 * *	0
1 0 1	* * * 0 * 0	0	* * 0 0 * 0	0	* 0 * 0 * 0	0	* 0 0 0 * 0	0	0 * * 0 * 0	0	0 * 0 0 * 0	0	0 0 * 0 * 0	0	0 0 0 0 * 0	0
1 1 0	* * * 0 0 *	0	* * 0 0 0 *	0	* 0 * 0 0 *	0	* 0 0 0 0 *	0	0 * * 0 0 *	0	0 * 0 0 0 *	0	0 0 * 0 0 *	0	0 0 0 0 0 *	0
1 1 1	* * * 0 0 0	0	* * 0 0 0 0	0	* 0 * 0 0 0	0	* 0 0 0 0 0	0	0 * * 0 0 0	0	0 * 0 0 0 0	0	0 0 * 0 0 0	0	0 0 0 0 0 0	0