Procedure to Solve Mutually Dependent Fault Trees (FT with Loops)

Takeshi MATSUOKA

Collaboration Centre for Research and Development, Utsunomiya University, Japan.
E-mail: mats@cc.utsunomiya-u.ac.jp

This paper presents procedure to solve mutually dependent Fault Trees in success events expression. First, explain the basic procedure by using simple dependent fault trees. Next, the procedure is applied to more complex fault trees with 3 non-linear interrelated loops. FT structures are converted into a physical system model, which satisfy Boolean equations in success events expression. It is shown that the representative possible states of a physical system corresponding to mutually dependent Fault Trees can be obtained by formulary method. And FT with loops can be solvable without approximation.

Keywords: Fault tree analysis, Mutually dependent FT, Loop structure, Boolean equation, Exact method, formulary procedure.

1. Introduction

A variety of methods have been developed to solve mutually dependent Fault Trees. Combination of mutually dependent fault trees produce a fault tree in which top event recursively appears in lower places, that is, a fault tree with logical loops.

Conventional methods; Yang(1997), Vaurio (2007), Lim et al. (2007) propose to solve the logical loop problem by breaking the logical loops at the points where the dependencies among the support systems are relatively weak and developing new fault trees without the logical loops. But this method doesn't give us exact solution of Fault Trees with loops. Yang (1997) built contra-example for this approach, which shows its mistake. FT with loops can be expressed by Boolean equations and they are mathematically solvable with arbitrary sets; Matsuoka (2009).

This paper presents procedure to solve mutually dependent Fault Trees in success event expressions, where operating states are considered for physical model corresponding to FT model. First, explain the basic procedure by using simple dependent fault trees. Next, the procedure is applied to a more complex fault trees with 3 non-linear interrelated loops and the applicability of this procedure is shown. Finally the discussions and conclusions are given.

2. Analysis Conditions

As fault tree structure, gates are expressed by two main logic gates – AND-gate and OR-gate.

In success event expression, change of system states with time is considered and it is assumed that all the components are placed in standby state at initial time, and they are started at designated time. If a component fails, it cannot be repaired, that is, non-repairable model is taken up.

Components are classified into three types, SS-type, G-type and T-type, which adequately explain the operational state of a loop structured system; Matsuoka (2012).

3. Simple Dependent Fault Trees

First, consider a simple fault tree structure and brief procedure is explained for solving mutually dependent fault trees.

3.1 *Analysis in fault tree expression*

Following simple Fault Trees are taken up:

$$A = A_a + A_b \cdot B \quad (1)$$
$$B = B_b + B_a \cdot A, \quad (2)$$

where A and B are top events, A_a, A_b, B_a and B_b are non-repairable basic events. Above two fault trees are combined into one fault tree with only one recursion term A, but this combined fault tree has endless recursion as shown in Fig.1.

3.2 *Analysis in Success Event Expression*

Relations expressed in failure events can be converted into relations expressed in success events by using basic Boolean rules. Let denote a as a complement of failure event A. Then the Eq. (1) and Eq. (2) can be converted into following two expressions.

$$a = a_a \cdot b + a_a \cdot a_b \quad (3)$$
$$b = b_b \cdot a + b_b \cdot b_a, \quad (4)$$

Substitute Eq. (4) into Eq. (3),

$$a = a_a \cdot b_b \cdot a + a_a \cdot b_b \cdot b_a + a_a \cdot a_b \quad (5)$$

Proceedings of the 30th European Safety and Reliability Conference and the 15th Probabilistic Safety Assessment and Management Conference
Edited by Piero Baraldi, Francesco Di Maio and Enrico Zio
Published by Research Publishing, Singapore.
ISBN/DOI: TBA

The left side term "a" is defined by using the term "a" itself.

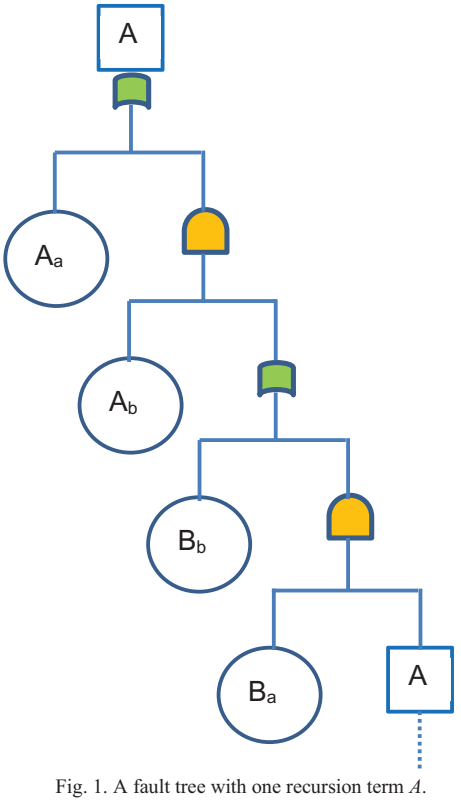

Fig. 1. A fault tree with one recursion term *A*.

3.2.1 *Solution by Considering a Structural Relation of Success Events*

The Boolean relations given by Eq. (3) and Eq. (4) are expressed by Fig. 2. Where "a_b" is a success event associated with some physical element. An arrow means a success event makes product with endpoint event. Product "$a_a \cdot a_b$" is produced. But in Fig.2, success event "a_a" is also related to "b", and "b" and "a_b" are parallel input to "a_a". Then, the resultant event, or output of some physical element, which is denoted as "a", has contributions from the "$a_a \cdot a_b$" and "$a_a \cdot b$".

We consider a physical system model which corresponds to Fig.2, as shown in Fig.3. In this figure, components are explicitly expressed, and the successful operation of component $\mathrm{A_b}$. is denoted as success event "a_b".

Components $\mathrm{A_b}$ and $\mathrm{B_a}$ support components $\mathrm{A_a}$ and $\mathrm{B_b}$, respectively, where components $\mathrm{A_b}$ and $\mathrm{B_a}$ are SS-type components denoted by yellow colour and components $\mathrm{A_a}$ and $\mathrm{B_b}$ are G-type or T-type components; Matsuoka (2012). The operation of SS-type component needs not require any support. The operation of G-type or T-type component requires support by other component.

The component $\mathrm{A_a}$ has two supports "a_b" and "b", then the output of component $\mathrm{A_a}$ (event "a") becomes Eq.(3).

$$a = a_a \cdot b + a_a \cdot a_b \tag{3}$$

Where, event "a_a" expresses elementally characteristics of component $\mathrm{A_a}$ itself, and does not express the actual operating state of component $\mathrm{A_a}$, but expresses some soundness of component $\mathrm{A_a}$; Matsuoka (2018).

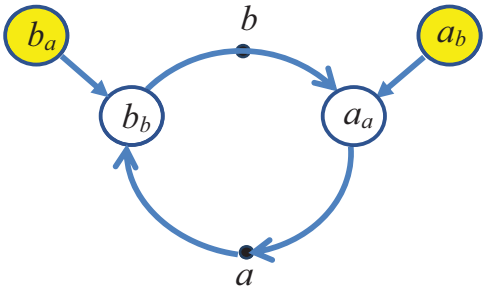

Fig. 2. Relations between success events in Eqs (3) and (4).

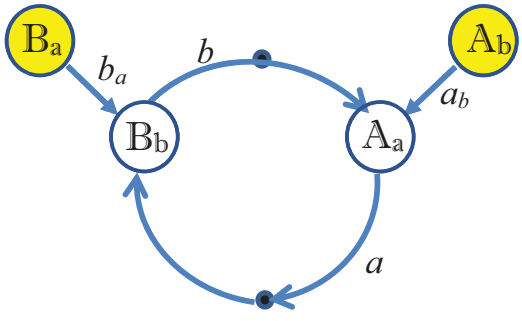

Fig. 3. Physical system model corresponds to Fig. 2.

It is possible to make other three different physical models which correspond to Boolean relations (3) and (4).

The endless recursive situation, which appears in the fault tree expression, is not appeared in Figs. 2 and 3. The term "$a_a \cdot b_b \cdot a$" in Eq. (5) corresponds to a loop from "a" to "a" via "b_b" and "a_a" as seen in Fig. 2. It also corresponds to a loop from "a" to "a" via "$\mathrm{B_b}$" and "$\mathrm{A_a}$" as seen in Fig. 3.

3.2.2 *Minimal Path Sets of Physical System*

All the components are assumed to be standby state at initial time. Then components are started one by one. There are four components, and then there are 24 possible starting orders (4x3x2x1).

Possible operational states can be exhaustively obtained. There are 12 starting orders for reaching the existence of output "*a*", and there are two combinations of component states for the existence of output "*a*".

Next, consider stopping the operation of components one by one and obtain possible operational states in which output "*a*" exists. The results are shown in Table 1. In this table, symbol " ○ " means the operation of component or existence of output "*a*". For other three physical system models, the same procedure has been done and the same final results are obtained.

The case 5 is the pure loop structured operation. Only two components are operating. It represents the situation that two components mutually support their operation.

Table 1. All the possible operating states in which output "a" exists.

Case No.	components Ab	Aa	Ba	Bb	Output "*a*"
1	○	○	○	○	○
2	×	○	○	○	○
3	○	○	×	○	○
4	○	○	○	×	○
5	×	○	×	○	○
6	○	○	×	×	○

Now identify minimal path sets. Minimal path set is a set of components whose simultaneous occurrence ensures that output "*a*" becomes "on", and if any component is removed from the set, the remaining components collectively no longer makes output "*a*" as "on". From table 1, it is seen that the smallest number of success components which makes path set is two. Their combinations are (Ab, Aa) and (Aa, Bb). Only two minimal path sets case 5 and case 6 exist in this physical model system.

3.2.3 *Solution by Boolean Equation*

Output "*a*" is expressed by Eq. (5) as shown in the section 3.2. and it can be solved as follows; Matsuoka (2009),

$$a = m \cdot a_a \cdot b_b + a_a \cdot b_b \cdot b_a + a_a \cdot a_b \quad (6)$$

where m is an arbitrary set in mathematical meaning, and it is determined depending on the starting sequence of operation in actual engineering system.

In the previous section, we examined possible operational states in rather qualitative consideration, and identified two minimal path sets. They correspond to terms $m \cdot a_a \cdot b_b$ and $a_a \cdot a_b$. In Eq. (6), there still exists second term $a_a \cdot b_b \cdot b_a$, because "*m*" is not qualitatively determined, and $a_a \cdot b_b \cdot b_a$ is not determined as a sub set of $m \cdot a_a \cdot b_b$.

3.2.4 *Determination of Arbitrary Set "m"*

Now, consider the process of making loop operation, and obtain exact value of *m* in Eq. (6).

At time t_1, start the components Ab and Ba. A set $a_b(t_1)$ is defined as component Ab is in operating state at time t_1. Next at time t_2, Bb is started, and inputs to Aa become $b_a(t_2)b_b(t_2)$ and $a_b(t_2)$. But Aa is not started and there is no output from Aa. At time t_3, start the component Aa. The outputs of Aa become $b_a(t_3)b_b(t_3)a_a(t_3)$ + $a_b(t_3)a_a(t_3)$, it is equal to "*a*" (output of Aa) and becomes to additional input to Bb as shown in Fig. 4.

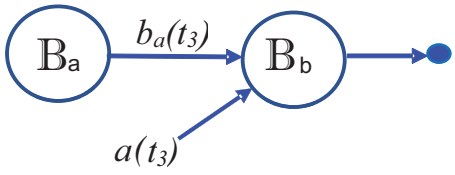

Fig.4. Additional input "*a*" to B_b..

There is takeover phenomenon between $b_a(t_3)$ and $a(t_3)$, and output of Bb ("$b(t_3)$") becomes as follows; Matsuoka (2012).

$$b(t_3) = b_a(t_3) \cdot b_b(t_3) + b_a(\tau_3) \cdot b_b(t_3) \cdot a_a(t_3) \quad (7)$$

Where τ_3 is a fixed value of time t_3 and $b_a(\tau_3)$ is also a fixed value, not a variable. Then, "*a*" (output of Aa) becomes,

$$\begin{aligned} a(t_3) &= b(t_3) \cdot a_a(t_3) + a_b(t_3) \cdot a_a(t_3) \\ &= b_a(\tau_3) \cdot b_b(t_3) \cdot a_a(t_3) \\ &+ b_a(t_3) \cdot b_b(t_3) \cdot a_a(t_3) + a_b(t_3) \cdot a_a(t_3) \end{aligned} \quad (8)$$

An arbitrary set "*m*" is determined as $b_a(\tau_3)$ compare to Eq. (6). As non-repairable model is assumed, $b_a(\tau_3)$ always include $b_a(t)$; $t>t_3$. Therefore, Eq. (8) finally becomes,

$$a(t_3) = b_a(\tau_3) \cdot b_b(t_3) \cdot a_a(t_3) + a_b(t_3) \cdot a_a(t_3) \quad (9)$$

Eq. (5) can be solved by Boolean arithmetic calculation with a consideration of takeover phenomenon. The result becomes to the same deduced by a method in which step by step check is performed for identifying all the possible operational states and identifying minimal path sets qualitatively, as shown in section 3.2.1.

3.3 *Solution of the Simple Dependent Fault Trees*

Now obtain the solution of fault tree shown in Fig. 1, from the Eq. (9). The Eq. (9) is converted into failure expression for $t>t_3$.

$$A(t) = A_b(t)B_a(\tau_3) + A_b(t)B_b(t) + A_a(t) \quad (10)$$

Graphical style becomes as follows,

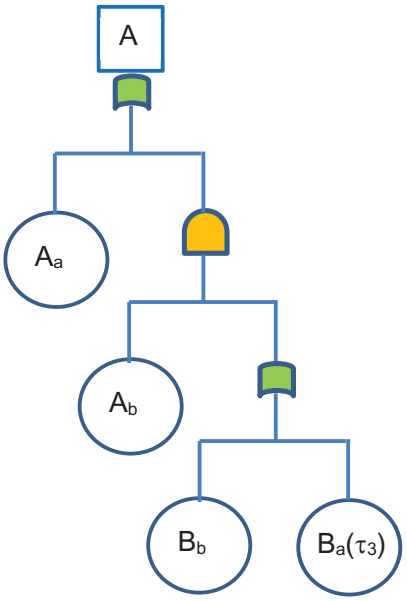

Fig.5. *Solution of simple dependent fault tees for $t>t_3$.*

The exact solution (Fig.5) is different from the one which is simply obtained by cut off the term A or A is replace by 1 in Fig.1. The factor $B_a(\tau_3)$ appears in the present starting process.

4. Complex Fault Trees

In this chapter, the procedure presented in the previous chapter is applied to more general loop structured system, and confirm this procedure is generally applicable to mutually dependent fault trees.

4.1 *Fault Trees with 3 non-linear Interrelated Loops*

Take up fault trees expressed in the set of Eq. (11) to Eq. (13).

$$A = A_a + A_b \cdot B + A_c \cdot C + A_{bc} \cdot B \cdot C \quad (11)$$
$$B = B_b + B_a \cdot A + B_c \cdot C + B_{ac} \cdot A \cdot C \quad (12)$$
$$C = C_c + C_b \cdot B + C_a \cdot A + C_{ab} \cdot A \cdot B \quad (13)$$

4.2 *Analysis in Success Event Expression for Complex Fault Trees*

4.2.1 *Solution by Boolean Equation*

Convert the Eq. (11), Eq. (12) and Eq. (13) into success event expression.

$$a = a_{bc}a_a a_b a_c + a_a a_c b + a_a a_b c + a_a bc \quad (14)$$
$$b = b_{ac}b_a b_b b_c + b_b b_a c + b_b b_c a + b_b ca \quad (15)$$
$$c = c_{ab}c_a c_b c_c + c_c c_b a + c_c c_a b + c_c ab \quad (16)$$

The set of Eq. (14), Eq. (15) and Eq. (16) represent rather complicated but symmetrical loop structured system as shown in Fig. 6.

Substitute Eq. (16) into Eq. (14) and Eq. (15), we can eliminate "c".

$$a = a_{bc}a_a a_b a_c + a_a a_b c_{ab}c_a c_b c_c + a_a a_b c_c c_b a + a_a a_c b + a_a c_c c_a b + a_a c_c ab \quad (17)$$

$$b = b_{ac}b_a b_b b_c + b_b b_a c_{ab}c_a c_b c_c + b_b b_c a + b_b c_c c_b a + b_a b_b c_c c_a b + b_b c_c ab \quad (18)$$

Unknown element "b" can be eliminated from Eq. (18) as follows; Matsuoka (2009).

$$b = b_{ac}b_a b_b b_c + b_b b_a c_{ab}c_a c_b c_c + b_b b_c a + b_b c_c c_b a + m_1 b_a b_b c_c c_a + m_2 b_b c_c a \quad (19)$$

Substitute equation (19) into equation (17), then we can obtain an expression of "a" without "b" and "c".

$$\begin{aligned} a = {} & a_{bc}a_a a_b a_c + a_a a_b c_{ab}c_a c_b c_c + \\ & a_a c_c c_a b_{ac}b_a b_b b_c + m_1 a_a c_c c_a b_b b_a + \\ & a_a c_c c_a b_b b_a c_{ab}c_b + a_a a_c c_{ac}b_a b_b b_c + \\ & a_a a_b c_c c_b a + a_a c_c c_a c_b b_b a + a_a a_c b_b b_c a + \\ & a_a a_c b_b c_b c_c a + a_a c_c b_b b_c a + m_2 a_a b_b c_c a \end{aligned} \quad (20)$$

Also "a" in right hand side can be replaced by arbitrary sets $n_1, n_2, \ldots$; Matsuoka (2009).

$$\begin{aligned} a = {} & a_{bc}a_a a_b a_c + n_1 a_a a_b c_b c_c + \\ & a_a a_b c_{ab}c_a c_b c_c + a_a a_c b_{ac}b_a b_b b_c + \\ & n_3 a_a a_c b_b b_c + n_4 a_a a_c b_b c_b c_c + \\ & a_a b_{ac}b_a b_b b_c a_c c_c + m_1 a_a b_a b_b c_a c_c + \\ & a_a b_a b_b c_{ab}c_a c_b c_c + n_2 a_a b_b c_a c_b c_c + \\ & n_5 a_a b_b b_c c_c + n_6 m_2 a_a b_b c_c \end{aligned} \quad (21)$$

4.2.2 *Determination of Arbitrary Sets "m, n"*

We consider a physical system model which corresponds to Fig.6. Physical system model can be made by replacing success events in Fig. 6 by corresponding components, as the same way shown in section 3.2.1.

Now, consider the process of making operational state of this physical system model.

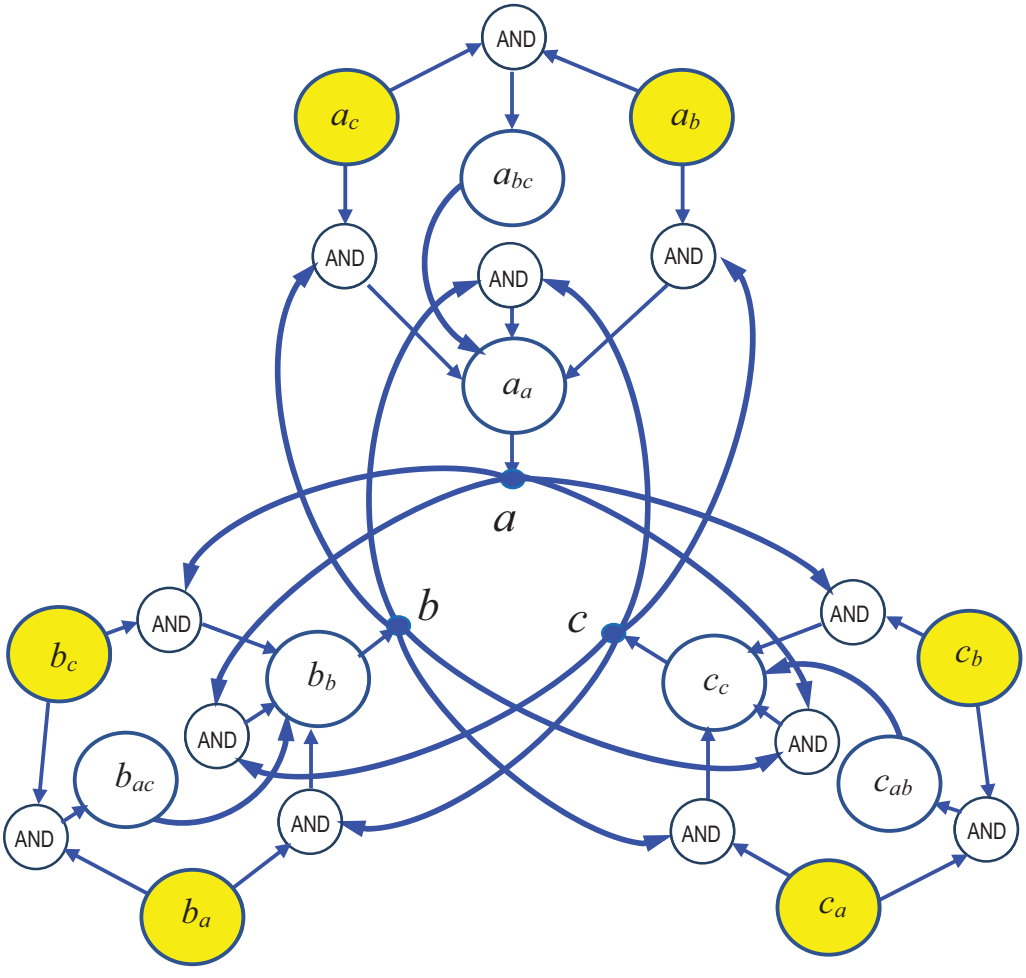

Fig. 6. Relations between success events in Eqs (14), (15) and (16).

At time t_1, start the components Aa, Ab, Ac and Abc. The outputs of Aa become $a_a(t_1)a_b(t_1)a_c(t_1)$ $a_{bc}(t_1)$ at this step. Then continue to start components as follows, Bc at t_2, Bb at t_3, Cc at t_4, Cb at t_5, ….. All the components can be started and placed in operating state. The arbitrary sets m_i, n_j are determined step by step with considering takeover phenomenon.

By eliminating the subsets of minimal path sets, expression of "a" becomes simple with 4 terms.

$$a = a_{bc}a_a a_b a_c + n_1 a_a a_b c_c c_b + n_3 a_a a_c b_b b_c + n_6 m_2 a_a b_b c_c \quad (21)$$

Exact values for m_i, n_j are given.

$$a(t) = a_c(t)a_b(t)a_{bc}(t)a_a(t) + a_c(\tau_5)\, a_{bc}(\tau_5)\, b_c(\tau_4)\, b_b(\tau_5)\, a_a(t)a_b(t)c_b(t)c_c(t) + a_b(\tau_3)a_{bc}(\tau_3)\, a_c(t)a_a(t)b_c(t)b_b(t) + a_b(\tau_3)a_{bc}(\tau_3)\, a_c(\tau_4)\, b_c(\tau_4)\, a_a(t)\, b_b(t)c_c(t). \quad (22)$$

We can see three loop structures for producing output "a" in this physical system model as shown in Figs. 7, 8 and 9.

4.3 *Solution of the Complex Fault Trees*

Now obtain the solution of complex fault trees shown in Eqs. (11), (12) and (13). The final result Eq. (22) is converted into failure expression for $t > t_5$.

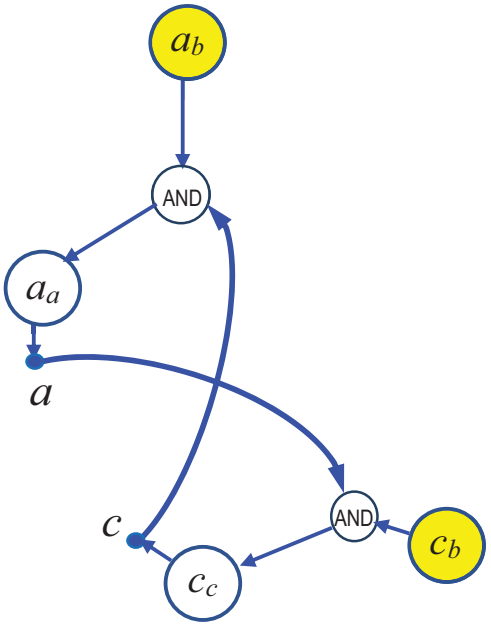

Fig. 7. The second term ($n_1 a_a a_b c_c c_b$) in Eq. (21).

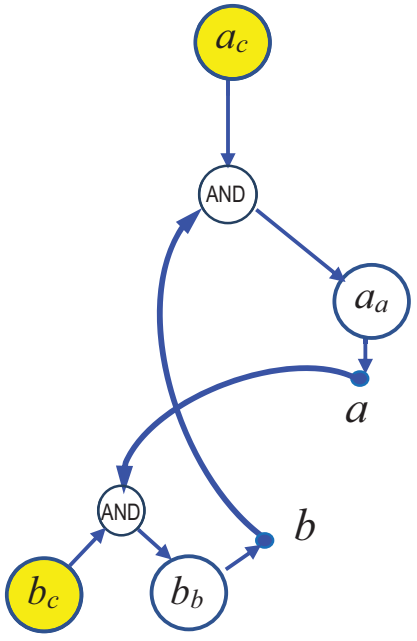

Fig. 8. The third term ($n_3 a_a a_c b_b b_c$) in Eq. (21).

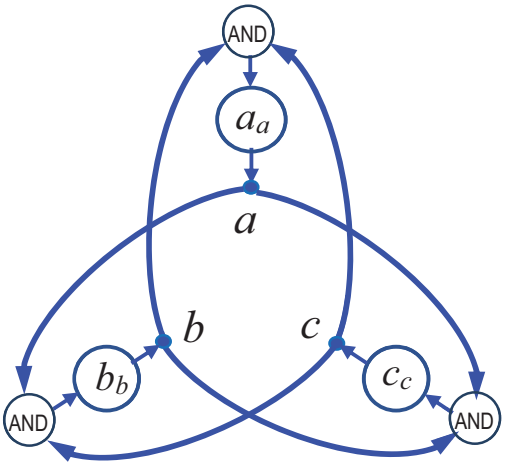

Fig. 9. The fourth term ($n_6 m_2 a_a b_b c_c$) in Eq. (21).

$$
\begin{aligned}
&\mathrm{TOP(A)} \\
&= A_a(t) + A_b(t)B_b(t) \;\; + A_b(t)B_c(t)C_c(t) \\
&+ \mathrm{A}_c\,(t)C_c(t) \;\; + \mathrm{A}_c\,(t)C_b(t)B_b(t) \\
&+ A_{bc}(t)C_c(t)B_b(t) + A_{bc}(t)C_b(t)B_b(t) \\
&+ A_{bc}(t)B_c(t)C_c(t) \;\; + A_b(\tau_3) + A_c(\tau_4) \\
&+ A_{bc}(\tau_3) + A_b(t)B_c(\tau_4) + A_c(t)B_c(\tau_4) \\
&+ A_c(t)B_b(\tau_5) + A_{bc}(t)B_c(\tau_4) + \\
&+ A_{bc}(t)B_b(\tau_5)
\end{aligned}
\tag{23}
$$

Above equation is the solution of complex mutually dependent fault trees deduced by a physical system model with some specific operational sequence. This equation can be expressed in FT style as shown in Fig. 10. There is no unknown element in this FT.

5. Conclusions

This paper presents a method to solve mutually dependent Fault Trees in success event expressions. Components included in a loop structure require support by other component. So, possible combinations of operating states are obtained by considering starting order of component's operations.

Simple FTs (FTs with ordinary loop) and 3 non-linear interrelated FTs are taken up and converted into success event expressions. Corresponding system models, which satisfy Boolean equations in success event expressions, are deduced. And possible operating states are identified by considering these physical system models' structures and starting orders of component's operations.

It is shown that FT with loops can be solvable without approximation.

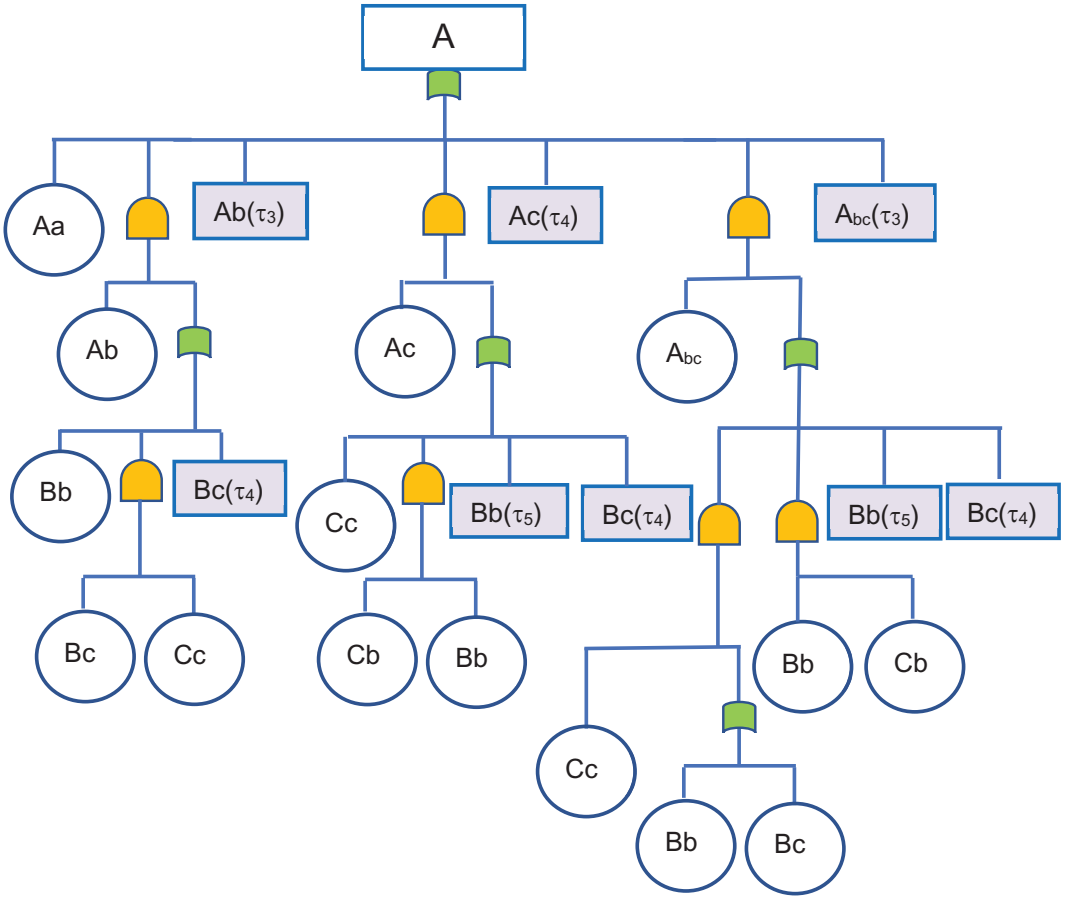

Fig. 10. Solution of complex mutually dependent fault trees *for* $t>t_5$.

References

Lim, H.G. and Jang, C.S. (2007). An Analytic Solution for a Fault Tree with Circular Logics in Which the Systems are Linearly Interrelated. *Reliability Engineering and System Safety* 92, 804-807.

Matsuoka, T. (2009). An Exact Method for Solving Logical Loops in Reliability Analysis. *Reliability Engineering and System Safety* 94, 1282-1288.

Matsuoka, T. (2012). Generalized Method for Solving Logical Loops in Reliability Analysis. *Proceedings of International Conference on Probabilistic Safety Assessment and Management, PSAM11*(25th-29th June 2012, Helsinki, Finland)

Matsuoka, T. (2018). Dynamic Behaviour of Nuclear Power Plant State under Severe Accident Conditions; Analysis by the GO-FLOW Methodology and the Consideration of Loop Structures. In Tunc Aldemir (Ed.), *Advanced Concepts in Nuclear Energy Risk Assessment and Management, Chapter 12* pp.427-476, World Scientific Publishing Co Pte Ltd.

Vaurio, J.K. (2007). A Recursive Method for Breaking Complex Logic Loops in Boolean System Models. *Reliability Engineering and System Safety* 92,1473-1475.

Yang, J.E. et al. (1997). Analytic Method to Break Logical Loops Automatically in PSA. *Reliability Engineering and System Safety* 56,101-105.