STSS/ISOFIC/ISSNP 2021
Special Session: Nuclear Safety Enhancement by Advanced ICT(II)

Takeshi MATSUOKA

Collaboration Center for Research and Development, Utsunomiya University

(mats@cc.utsunomiya-u.ac.jp)

STSS/ISOFIC/ISSNP 2021

Special Session: Nuclear Safety Enhancement by Advanced ICT(II)

Reliability evaluation of FTA with feedback loop

November 2021

Takeshi MATSUOKA

Collaboration Center for Research and Development, Utsunomiya University

(mats@cc.utsunomiya-u.ac.jp)

Introduction

- Many kinds of techniques are used for high reliable and safety systems, for example redundant components, back-up function by stand-by component, principle of diversity, and so on.
- In nuclear power plants, many subsystems are sometimes mutually supported and/or recursively supported by main system.
- This system configuration leads to a problem of solving Fault Trees with feedback loop.
- This paper presents procedure to solve mutually dependent Fault Trees in success event expressions.

Solving Fault Tree (FT)

- Obtain all the possible minimal cut sets.
- It is easy for FTs without feedback loop.
- First, simply make products (cut sets) based on the FT structure, and eliminate sub cut sets.
- Then all the minimal cut sets can be obtained.

Difficulty for FT with feedback loop

- For FT with feedback loop, the top event recursively appear and cut sets are endlessly produced.
- If reappeared top event is ignored at certain point (most of the proposed methods to solve FT with loop), we can obtain some approximate solution.
- But the result is not assured if there are some missing minimal cut sets or not.

Analysis Conditions in the presentation

- 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.
- 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.

Simple Dependent Fault Trees

$$A = Aa + Ab \cdot B$$

$$B = Bb + Ba \cdot A$$

- where *A* and *B* are top events, *Aa, Ab, Ba* and *Bb* are non-repairable basic events.
- Above two fault trees are combined into one fault tree with only one recursion term *A*.
- This combined fault tree has endless recursion as shown in the next figure.

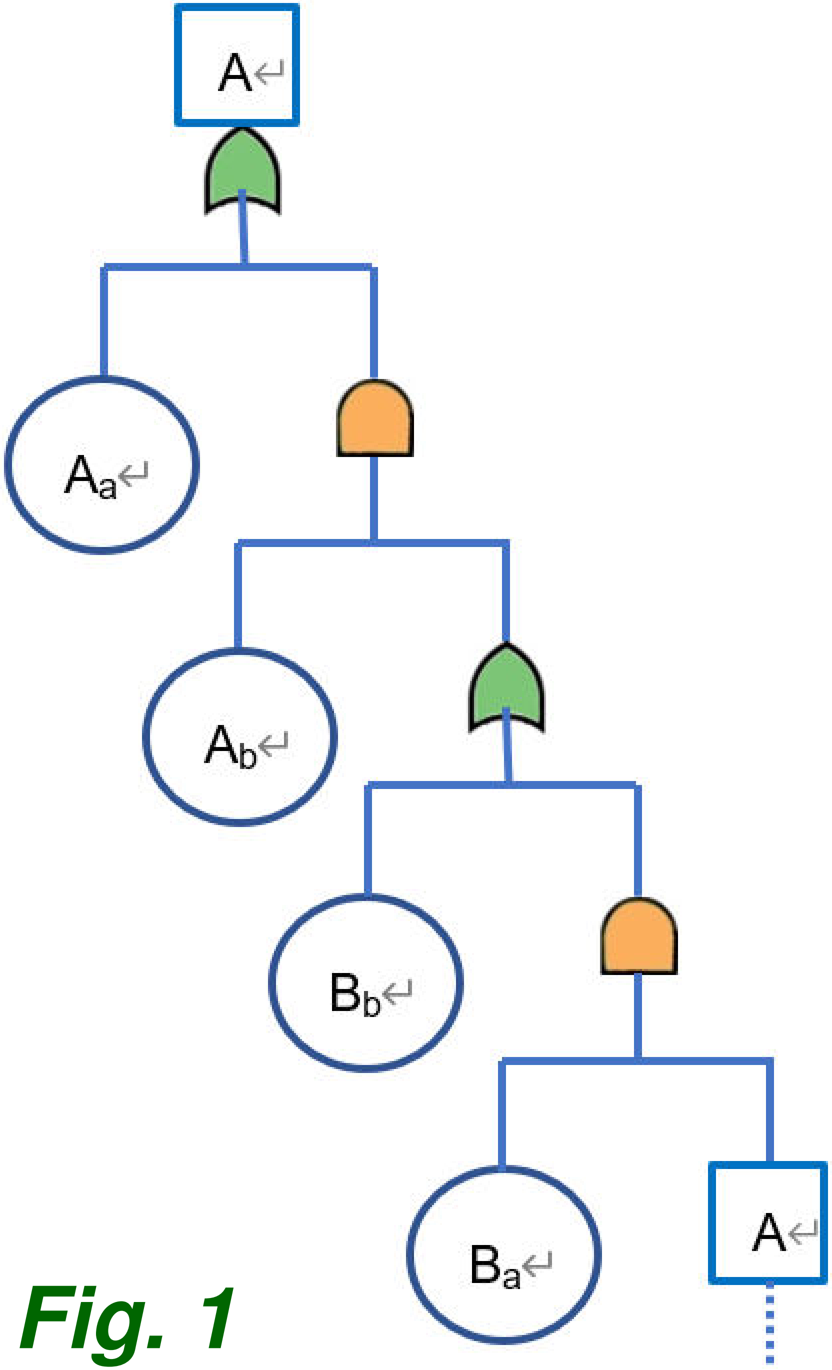

Fig. 1

Analysis by conventional methods

- Simple cut off method,

  $A = Aa + Ab \cdot (Bb + Ba \cdot A) = Aa + Ab \cdot Bb + Ab \cdot Ba \cdot A$

  $\rightarrow A = Aa + Ab \cdot Bb$

- Algorithm by Yang J. E. et al[1]

  When we get some cycle (type A-B-A, etc.), stop expansion on this direction and to delete this Min Cut Set .

  $A = Aa + Ab \cdot B = Aa + Ab \cdot Bb + Ab \cdot Ba \cdot A$

  $\rightarrow A = Aa + Ab \cdot Bb$

- Algorithm by Vaurio[2]

  A recursive method for breaking complex logic loops.

  Start with $A=\phi$, $B=\phi \rightarrow A = Aa$, $B = Bb$

  $\rightarrow A = Aa + Ab \cdot Bb$, $B = Bb + Ba \cdot Aa \rightarrow$ *same*

Analysis by conventional methods

- Factor graph method [3]

  In this method, endless connection is terminated at a certain point.

- The BDD(Binary Decision Diagram) method[4]

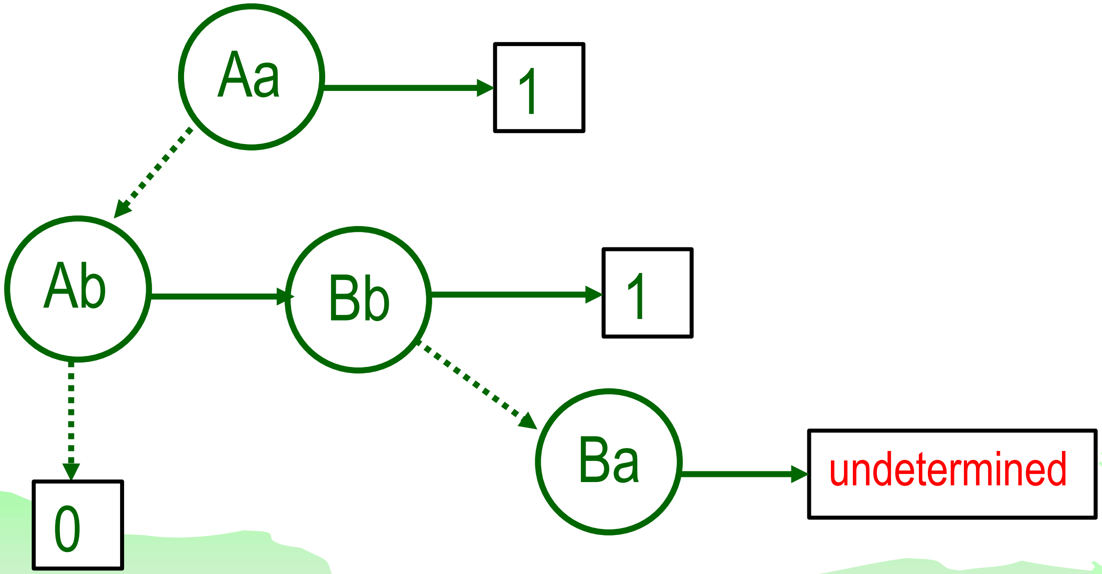

Fig. 2

Analysis in Success Event Expression

- Relations expressed in failure events can be converted into relations expressed in success events.

$$a = a_a \cdot b + a_a \cdot a_b (4)$$

$$b = b_b \cdot a + b_b \cdot b_a, (5)$$

- Substitute Equation (5) into Equation (4),

$$a = a_a \cdot b_b \cdot a + a_a \cdot b_b \cdot b_a + a_a \cdot a$$

Relations between success events

- The Boolean relations given by Eq. (4) and Eq. (5) are expressed by the following figure (Fig.3).
- 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

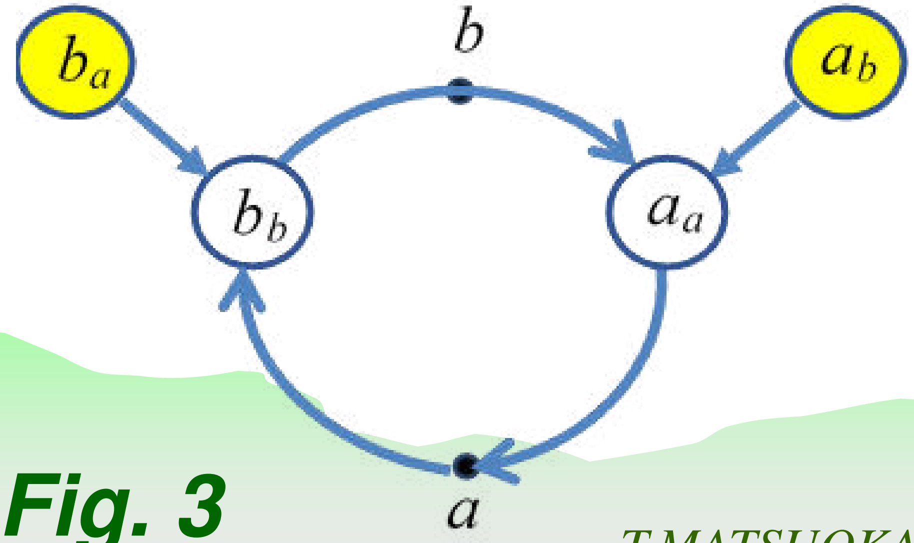

Fig. 3

T.MATSUOKA, Utsunomiya University

Physical system model corresponds to Fig. 3.

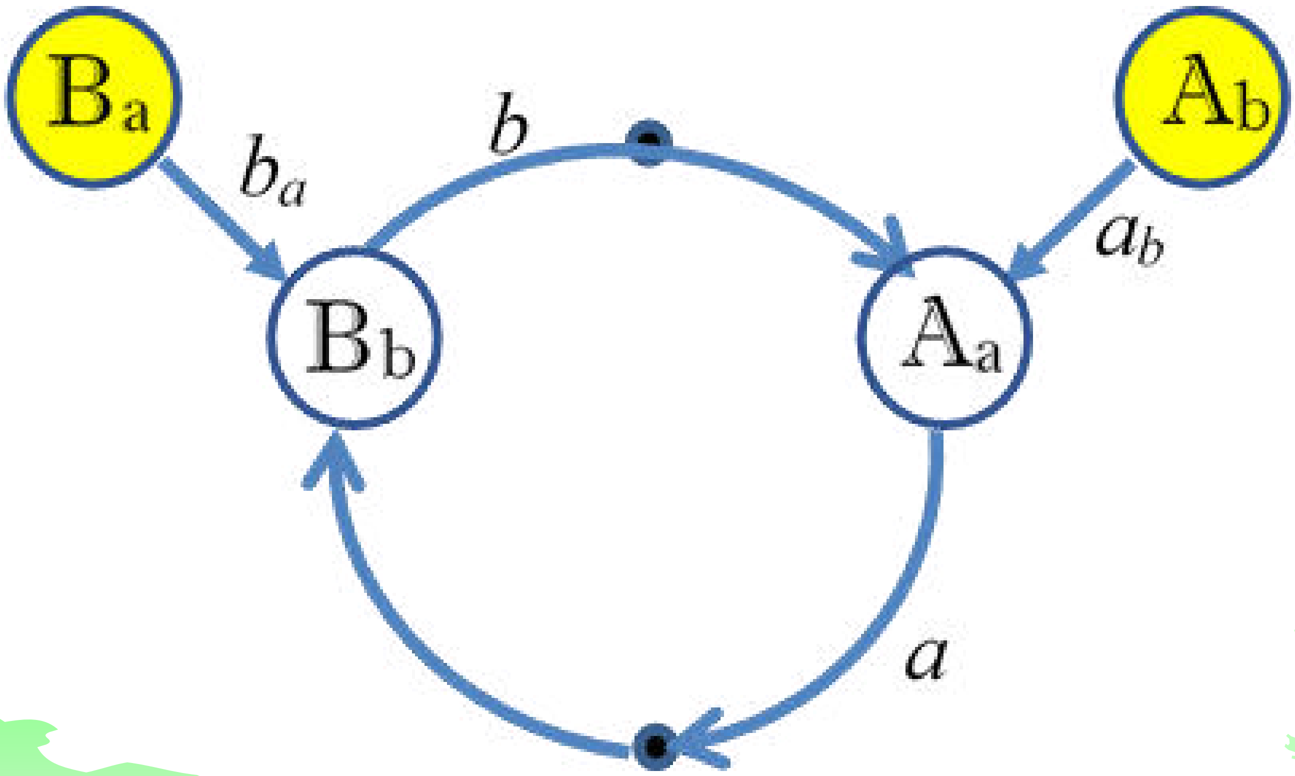

Fig. 4

Structural Relation of Success Events

- The endless recursive situation, which appears in the fault tree expression, is not appeared in Figs. 3 and 4.
- The term "$a_a \cdot b_b \cdot a$" in Eq. (6) corresponds to a loop from "a" to "a" via "b_b" and "a_a" as seen in Fig. 3.
- It also corresponds to a loop from "a" to "a" via "B_b" and "A_a" as seen in Fig. 4.

Solution by Boolean Equation

- Output "a" is expressed by Eq. (5) and it can be solved as follows; *Matsuoka (2009)[5].*

$$a = m \cdot a_a \cdot b_b + a_a \cdot b_b \cdot b_a + a_a \cdot a_b$$

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

Determination of Arbitrary Set "m"

Consider the process of making loop operation and obtain exact value of m in Eq. (7).

- At time t_1, start the components A_b and B_a.
- A set $a_b(t_1)$ is defined as component A_b is in operating state at time t_1.
- Next at time t_2, B_b is started, and inputs to A_a become $b_a(t_2)b_b(t_2)$ and $a_b(t_2)$. But A_a is not started and there is no output from A_a.
- At time t_3, start the component A_a. The outputs of A_a 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 A_a) and becomes to additional input to B_b as shown in Fig. 5.

Takeover phenomenon between $b_a(t_3)$ and $a(t_3)$

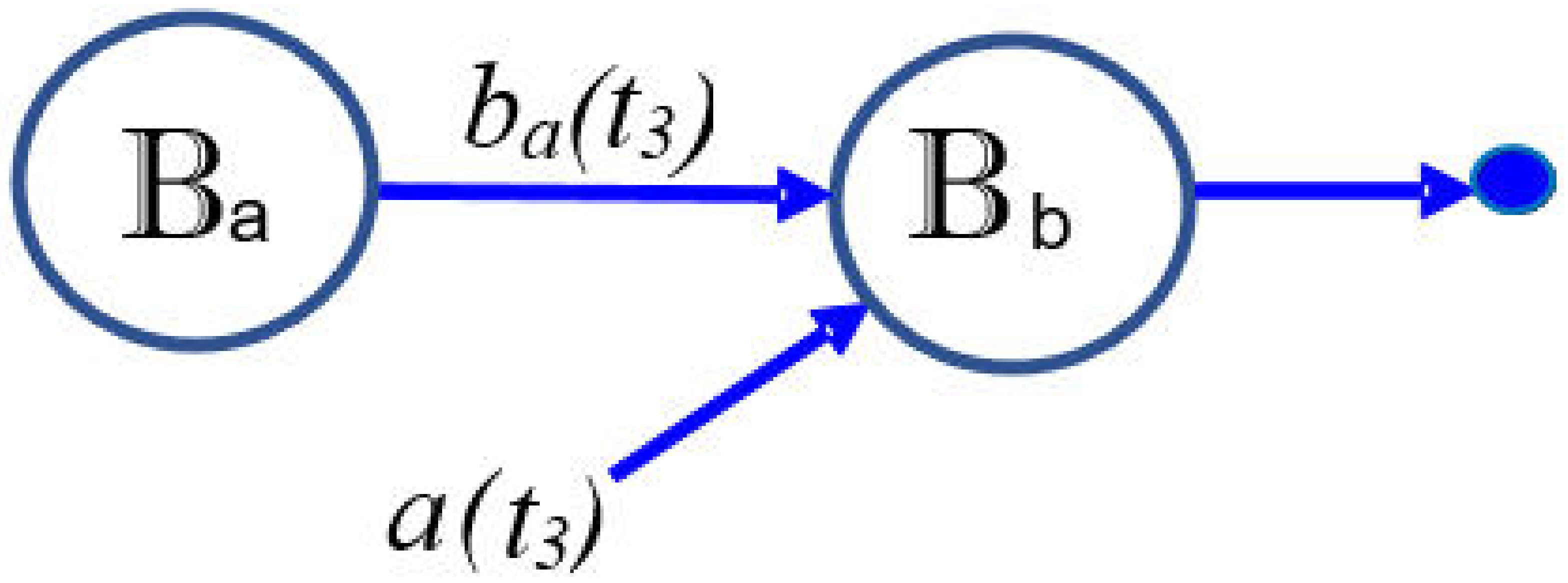

Fig. 5 Additional input "a" to Bb.

$$b(t_3) = b_a(t_3)\cdot b_b(t_3) + b_a(\tau_3)\cdot a(t_3)\cdot b_b(t_3)$$

$$= b_a(t_3)\cdot b_b(t_3) + b_a(\tau_3)\cdot\left(b_a(\tau_3)\cdot b_b(t_3)\cdot a_a(t_3) + a_b(t_3)\cdot a_a(t_3)\right)\cdot b_b(t_3)$$

$$b(t_3) = b_a(t_3)\cdot b_b(t_3) + b_a(t_3)\cdot b_b(t_3)\cdot a_a(t_3) \quad (8)$$

$$a(t_3) = b(t_3)\cdot a_a(t_3) + a_b(t_3)\cdot a_a(t_3)$$

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

$$+ a_b(t_3)\cdot a_a(t_3) \quad (9)$$

$$m = b_a(\tau_3)$$

$$a(t_3) = b_a(\tau_3)\cdot b_b(t_3)\cdot a_a(t_3) + a_b(t_3)\cdot a_a(t_3) \quad (10)$$

- *Eq. (6)* can be solved by Boolean arithmetic calculation with a consideration of takeover phenomenon.
- Now obtain the solution of fault tree shown in Fig. 1, from the *Eq. (10)*. The *Eq. (10)* is converted into failure expression for $t > t_3$.

$$A(t) = A_b(t)B_a(\tau_3) + A_b(t)B_b(t) + A_a(t) \quad (11)$$

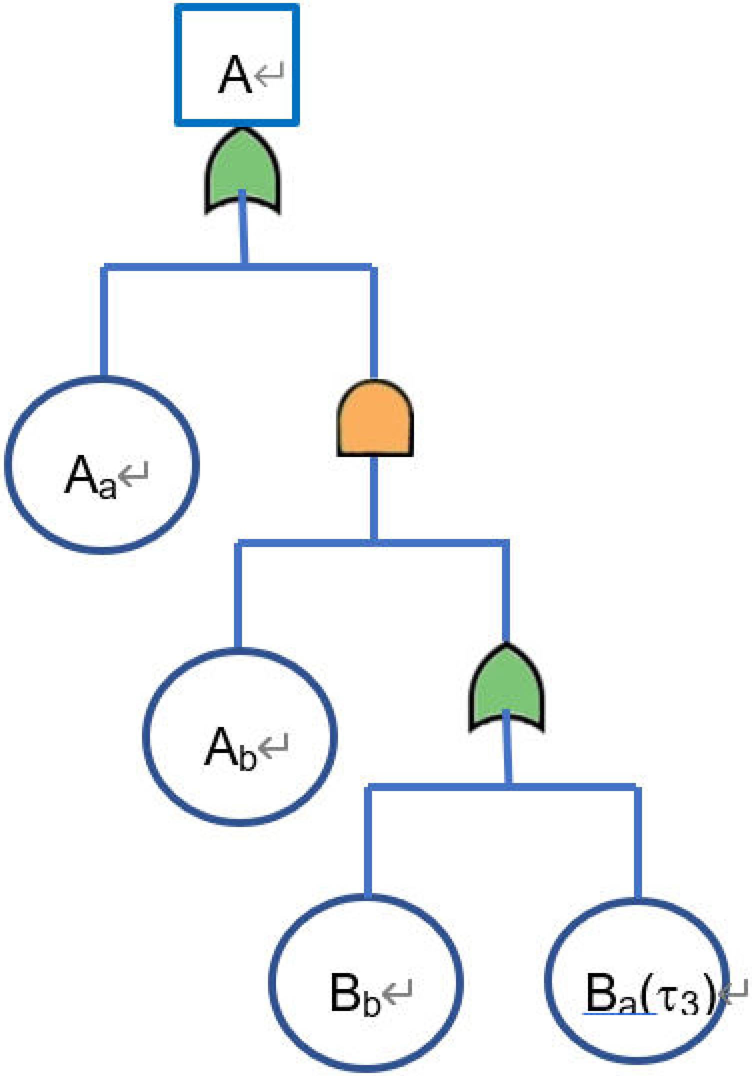

Fig. 6 Solution of simple dependent fault trees for t>t3.

Complex Fault Trees

- Apply to more general loop structured system, and confirm this procedure is generally applicable to mutually dependent fault trees.

$$A = A_a + A_b \cdot B + A_c \cdot C + A_{bc} \cdot B \cdot C \quad (12)$$
$$B = B_b + B_a \cdot A + B_c \cdot C + B_{ac} \cdot A \cdot C \quad (13)$$
$$C = C_c + C_b \cdot B + C_a \cdot A + C_{ab} \cdot A \cdot B \quad (14)$$

$$a = a_{bc} a_a a_b a_c + a_a a_c b + a_a a_b c + a_a bc \quad (15)$$
$$b = b_{ac} b_a b_b b_c + b_b b_a c + b_b b_c a + b_b ca \quad (16)$$
$$c = c_{ab} c_a c_b c_c + c_c c_b a + c_c c_a b + c_c ab \quad (17)$$

Expression by arbitrary sets m_i, n_j

- With some tedious calculations "a" is obtained as follows,

$$a = a_{bc}\cdot a_a\cdot a_b\cdot a_c + n_1\cdot a_a\cdot a_b\cdot c_b\cdot c_c + a_a\cdot a_b\cdot c_{ab}\cdot c_a\cdot c_b\cdot c_c + a_a\cdot a_c\cdot b_{ac}\cdot b_a\cdot b_b\cdot b_c$$
$$+ n_3\cdot a_a\cdot a_c\cdot b_b\cdot b_c + n_4\cdot a_a\cdot a_c\cdot b_b\cdot c_b\cdot c_c + a_a\cdot b_{ac}\cdot b_a\cdot b_b\cdot b_c\cdot c_a\cdot c_c + m_1\cdot a_a\cdot b_a\cdot b_b\cdot c_a\cdot c_c$$
$$+ a_a\cdot b_a\cdot b_b\cdot c_{ab}\cdot c_a\cdot c_b\cdot c_c + n_2\cdot a_a\cdot b_b\cdot c_a\cdot c_b\cdot c_c + n_5\cdot a_a\cdot b_b\cdot b_c\cdot c_c + n_6\cdot m_2\cdot a_a\cdot b_b\cdot c_c$$

- Relations between success events becomes next figure.

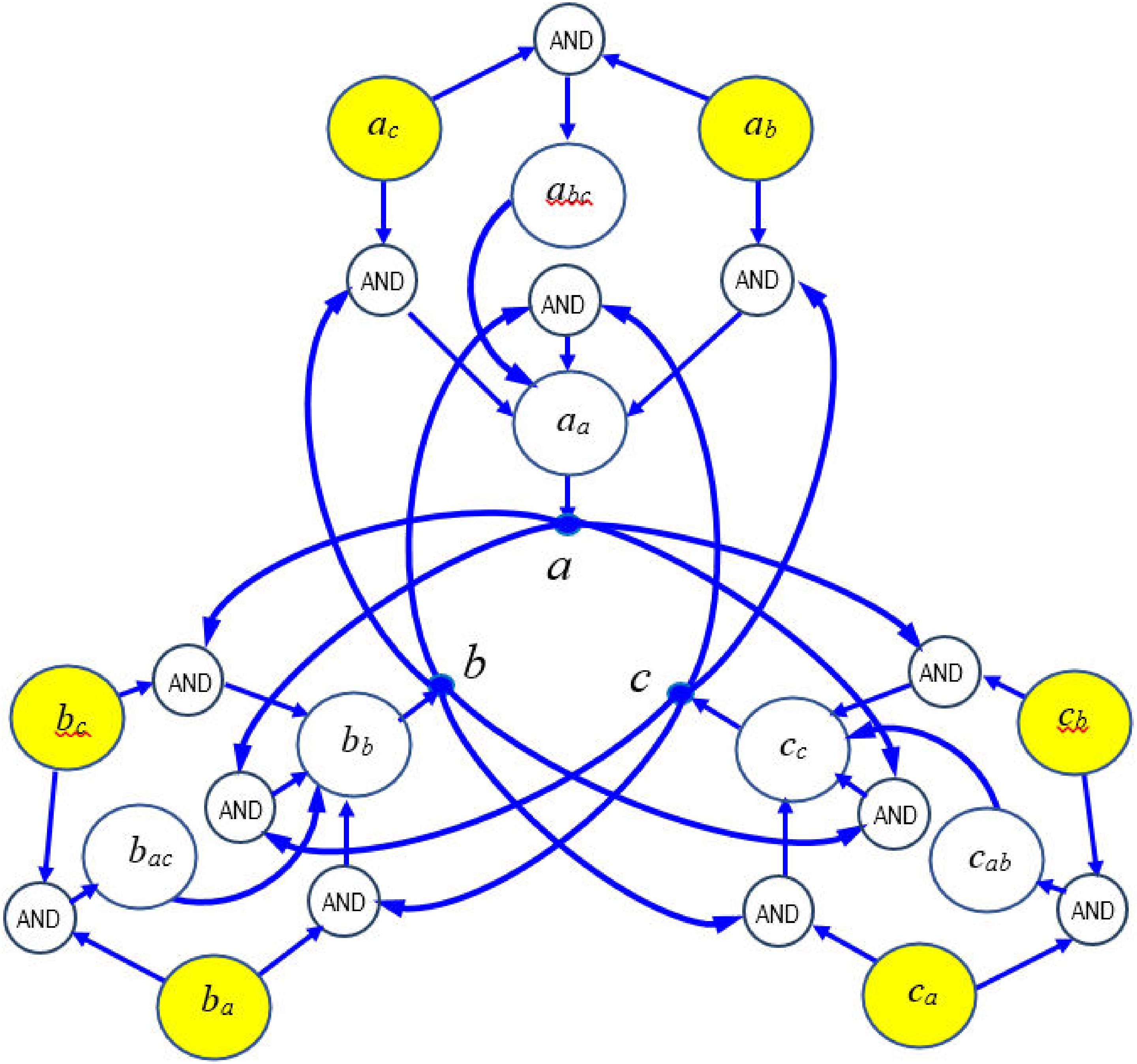

Fig. 7 Relations between events in Eq. (15),(16) and (17).

Final Results

- 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.

Failure expression

$$
\begin{aligned}
TOP(A) = {} & A_a(t) + A_b(t)\cdot B_b(t) + A_b(t)\cdot B_c(t)\cdot C_c(t) + A_c(t)\cdot C_c(t) + A_c(t)\cdot C_b(t)\cdot B_b(t) + A_{bc}(t)\cdot C_c(t)\cdot B_b(t) + A_{bc}(t)\cdot C_b(t)\cdot B_b(t) \\
& + A_{bc}(t)\cdot B_c(t)\cdot C_c(t) + A_b(\tau_3) + A_c(\tau_4) + A_{bc}(\tau_3) + A_b(t)\cdot B_c(\tau_4) + A_c(t)\cdot B_c(\tau_4) + A_c(t)\cdot B_c(\tau_5) + A_{bc}(t)\cdot B_c(\tau_4) + A_{bc}(t)\cdot B_b(\tau_5)
\end{aligned} \tag{18}
$$

Final fault tree is shown in the next figure.

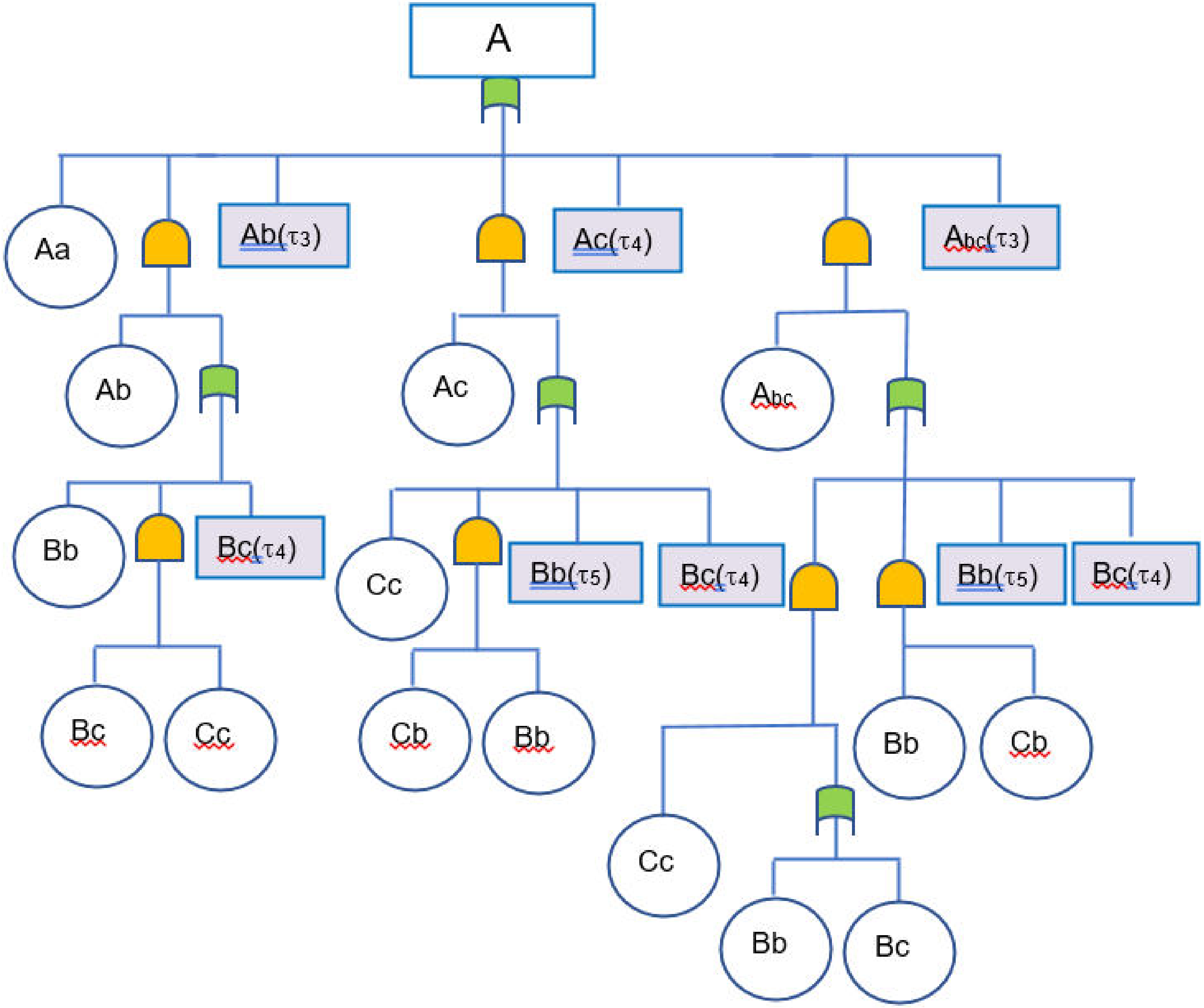

Fig. 8 Solution of complex mutually dependent fault tree for t>t3.

Summary

- 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.
- Simple FTs (FTs with ordinary loop) and 3 non-linear interrelated FTs are taken up.
- FTs are converted into success event expressions.

Summary

- Corresponding system models, which satisfy Boolean equations in success event expressions, are deduced.
- 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.

References

[1]J.E. Yang, S.H. Han, J.H. Park , Y.H. Jin, Analytic method to break logical loops automatically in PSA, Reliability Engineering and System Safety, Vol. 56, pp.101-105, 1997.

[2]J.K.Vaurio, A Recursive method for breaking complex logic loops in Boolean system models, Reliability Engineering and System Safety, Vol.92, pp.1473-1475, 2007.

[3]Y. H. Chae, S. G. Kim, P. H. Seong. Reliability of the system with loops: Factor graph based approach. Reliability Engineering and System Safety, Vol.208, 2021, 107407

[4]W.S. Jung, S.H. Han, J. Ha, A fast BDD algorithm for large coherent fault trees analysis, Reliability Engineering and System Safety Vol.83, pp. 369–374, 2004.

[5]T. Matsuoka, An exact method for solving logical loops in reliability analysis, Reliability Engineering and System Safety, Vol.94, pp. 1282-1288, 2009.

[6]T.Matsuoka, Procedure to solve mutually dependent Fault Trees (FT with loops), Reliability Engineering and System Safety, Vol.214, 2021, 107667.

Thank you for your kind attention !