Transactions of the Korean Nuclear Society Autumn Meeting
Busan, Korea, October 27-28, 2005

Logical Loop Breaking Method in Fault Tree Analysis

Woo Sik Jung, Joon-Eon Yang, Jaejoo Ha
Korea Atomic Energy Research Institute, P.O.Box 105, Yusong, Daejon, Korea, woosjung@kaeri.re.kr

1. Introduction

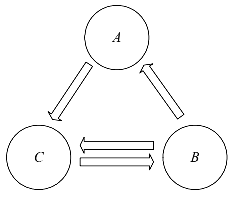
Logical loops or circular logics are interpreted as circular supporting relations among systems or their fault trees [1-4]. The logical loops could be found by examining the supporting relations among the systems before developing their fault trees. The merged fault tree is created by combining the system fault trees. The logical loop in a merged fault tree exists in a shape of a circular connection of gates.

There are two ways to break logical loops such as the analytical [5-9] and manual breaking methods [1-4]. Guidance for the manual breaking of the logical loops is provided in NUREG/CR-2728 [1]. ASME probabilistic risk assessment standard [5] recommends that a significant conservatism or non-conservatism be avoided when breaking the logical loops.

Yang [6] presented an analytical method to break the logical loops. By using the analytical method [6], KIRAP [7] and FTREX [8,9] recursively search and break the logical loops in a fault tree and then solve the broken fault tree.

2. Analytical Method

The analytical method to break logical loops was presented by Yang [6]. The example in Fig. 1 is provided as an illustration of the analytical logical-loop breaking method. In this example, the circular relations are represented by the following Boolean equations.



$$\begin{aligned} A &= A_a + A_b B \\ B &= B_b + B_c C \\ C &= C_c + C_a A + C_b B \ . \end{aligned} \quad (1)$$

Fig.1 Example of logical loop

The fault tree in Fig. 1 has 3 combinations that cause the logical loops such as *A<B<C<A, B<C<B*, and *C<B<C*. The fault tree is expanded in a top-down way. During the expansion, the combinations of events are examined and the combinations that cause the logical loops are deleted as follows:

$$\begin{aligned} A &= A_a + A_b(B_b + B_c C) \\ &= A_a + A_b B_b + A_b B_c (C_c + C_a A + C_b B) \\ &= A_a + A_b B_b + A_b B_c C_c + \underline{A_b B_c C_a A} + \underline{A_b B_c C_b B} \\ &= A_a + A_b B_b + A_b B_c C_c \end{aligned} \quad (2)$$

The underlined 2 combinations in Eq. (2) cause the logical loops. The combinations $A_bB_cC_aA$ and $A_bB_cC_aB$ are instances of the logical loops *A<B<C<A* and *B<C<B* respectively. The underlined 2 combinations are deleted.

The analytical solution at the system level could be obtained without an actual manipulation of the fault trees[10]. The manual operation of the fault trees in the analytical solution is performed, and then the broken fault tree is solved by any fault tree quantifier. Let us consider the example in Fig. 1 as

$$\begin{aligned} A &= f(B \to \overline{A}) = A_a + A_b B \\ B &= f(C \to \overline{B}) = B_b + B_c C \\ C &= f(A,B \to \overline{C}) = C_c + C_a A + C_b B \ . \end{aligned} \quad (3)$$

In this case, the independent fault trees are

$$\overline{A} = A_a + A_b \ , \ \overline{B} = B_b + B_c \ , \ \overline{C} = C_c + C_a + C_b \ . \quad (4)$$

If *A<B<C<A* is found, then the following solution is obtained

$$\begin{aligned} C_1 &= f(A,B \to \overline{C})/(A,B) = C_c \\ B_1 &= f(C_1 \to \overline{B})/A = B_b + B_c C_1 \ . \\ A &= f(B_1 \to \overline{A}) = A_a + A_b B_1 \end{aligned} \quad (5)$$

The logical structure in Eq. (5) is identical to the one in Eq. (2).

3. Background of the analytical solution

When a logical loop among several systems *X<Y<Z<…<X* is arranged for the system *X* , the Boolean equation becomes

$$X = a + bX \quad (6)$$

where *X*, *Y*, and *Z* are systems or Boolean equations such as system functions or fault trees. It has a general solution

$$X = a + bG \quad (7)$$

where *G* is an arbitrary Boolean equation.

The solutions of a system function *X* are Prime Implicants (PIs) or minimal cut sets (MCSs). The type of solutions and the necessary method are determined by the reliability analyst. By the definitions of PIs and MCSs, the system function should be TRUE (Ω), when making all the elements of a solution TRUE.

The solutions are generated from the two terms *a* and *bG*. When making all the elements in one PI or MCS of *a* TRUE, *a* and the system function in Eq. (6) become TRUE as

$$X = \Omega + X = \Omega \ . \quad (8)$$

Although each solution of *bG* can make *b* and *G* TRUE, it can not make Eq. (6) TRUE. That is, the state of *X* is still indefinite as

$$X = a + X. \quad (9)$$

Thus, the solutions of *bG* can not make the system function in Eq. (6) TRUE. In order for all the solutions from Eq. (7) to satisfy the basic definitions of the solutions in the fault tree analysis, the general solution in Eq. (7) should be reduced to

$$X = a. \quad (10)$$

4. Procedure for breaking logical loops

After integrating the fault trees of the supporting systems, the logical loops in the fault tree should be carefully examined. In this study, the logical loops are categorized into three types as follows:

1. Type A logical loops are nonsense logical loops. The logical loops should be manually fixed.
2. Type B logical loops are nonsense logical loops.
$$X = a + bX \rightarrow X = a \quad (11)$$
3. Type C logical loops are proper logical loops.
$$X = a + bX \rightarrow X = a \quad (12)$$

The type B and C logical loops could be broken together since they have the same resultant formula. In this study the following procedures are recommended:

1. Find logical loops and determine their types
2. Manually delete Type A logical loops
3. Manually break Type B and C logical loops. It is recommended to use dedicated computational tools such as KIRAP or FTREX to automatically break the logical loops.
4. Solve the broken fault tree.

A typical example of Type A logical loop is in Ref. 4. In order to illustrate Type A logical loop and the breaking procedure in Step 2, let us consider the failure to start of the diesel generator in the case of a station black out (SBO) condition as

$$\textit{DG failure to start} = c + d \times \textit{125V DC failure} \quad (13)$$
$$\textit{125V DC failure} = e + \textit{Battery failure} \times \textit{Battery charger failure} \quad (14)$$

The logics in Eqs. (13) and (14) generate Type A logical loop on the Battery charger failure (*X*)

$$X = a + \textit{Battery failure} \times X \quad (15)$$

Since there is no available offsite and onsite power when the diesel generator is starting, battery charger failure is trimmed away from the Boolean equations in Eq. (14) as

$$\textit{DG failure to start} = c + d \times (e + \textit{Battery failure}) \quad (16)$$

FTREX[9] has a reporting capability of logical loops. The logical loop information of the latest Ulchin 3/4 NPP core damage fault tree is summarized in Table 1. The fault tree is developed for the input to the risk monitor based on the PSA study [11].

Table 1. Logical loops KSNP PSA

X that has a logical loop structure X=a+bX Type A: Battery charger failure (X) and Battery failure (b) in SBO Type B or C: Component cooling water pumps 1A (X) and 2A (b) Emergency diesel generators 1A/1B (X) and AAC (b)	17
X that has a logical loop structure X=A+X	46
Number of logical loops	2,650

5. Conclusion

In this study, the analytical method and its background to break logical loops are explained. Furthermore, the procedure for breaking logical loops is presented. A careful attention should be paid to the manual breaking of the logical loops, since it is a very complicated task. Therefore, it is recommended to use dedicated computational tools such as KIRAP or FTREX to automatically break the logical loops.

REFERENCES

[1] D.D. Carlson, Interim Reliability Evaluation Program Procedure Guide, NUREG/CR-2728, SAND82-1100, U.S. Nuclear Regulatory Commission, Washington, DC, 1983.

[2] G.A. Coles, Powers TB, "Breaking the Logic Loop to Complete the Probabilistic Risk Assessment," Proceeding of PSA 89: International Topical Meeting Probability, Reliability, and Safety Assessment, pp.1155-1160, Pennsylvania, USA, 1989.

[3] D.I. Kang, Hwang MJ, Yang JE, Jin YH, Shapiro HS, "Breaking the Complex Logical Loops in Wolsong 2/3/4 PSA," PSA '95, Seoul, Korea, 1995.

[4] M. Demichela, N. Piccinini, I. Ciarambino, S. Contini, "How to Avoid the Generation of Logic Loops in the Construction of Fault Trees," Reliability Engineering and Safety, Vol. 84, pp. 197-207, 2004.

[5] ASME, "Standard for Probabilistic Risk Assessment for Nuclear Power Plant Applications," ASME RS-S-2002, 2002.

[6] J.E. Yang, S.H. Han, J.H. Park, Y.H. Jin, "Analytic Method to Break Logical Loops Automatically in PSA," Reliability Engineering and Safety, Vol. 56, pp. 101-105, 1997.

[7] S.H. Han, "PC-workstation based level 1 PRA code package-KIRAP," Reliability Engineering and System Safety, Vol. 30, pp.313-322, 1990.

[8] W.S. Jung, S.H. Han, J.J. Ha, "A Fast BDD Algorithm for Large Coherent Fault Trees Analysis," Reliability Engineering and System Safety, Vol. 83, pp. 369–374, 2004.

[9] W.S. Jung, S.H. Han, J.J. Ha, "Development of an Efficient BDD Algorithm to Solve Large Fault Trees," Proceedings of the 7th International Conference on Probabilistic Safety Assessment and Management, June, Berlin, Germany, 2004.

[10] W.S. Jung, S.H. Han, "Development of an analytical method to break logical loops at the system level," Reliability Engineering and System Safety, Vol. 90, pp. 37-44, 2005.

[11] Korea Electric Power Corporation. Ulchin Units 3 and 4 Final Probabilistic Safety Assessment Report, 1997.