Fault tree analysis in the R programming environment. Treatmentof common cause failures

Alexander V. Antonov, *JSC RASU, Moscow, Russia*
Evgeny Yu. Galivets, *JSC RASU, Moscow, Russia*
Valery A. Chepurko, *JSC RASU, Moscow, Russia*
Alexey N. Cherniaev, *JSC RASU, Moscow, Russia*

Alexander V. Antonov

Evgeny Yu. Galivets

Valery A. Chepurko

Alexey N. Cherniaev

Abstract. **Aim.** *This paper is the continuation of [1] that proposes using the R programming language for fault tree analysis (FTA). In [1], three examples are examined: fault tree (FT) calculation per known probabilities, dynamic FT calculation per known distributions of times to failure for a system'selements. In the latter example, FTA is performed for systems with elements that are described by different functional and service models. Fault tree analysis (FTA) is one of the primary methods of dependability analysis of complex technical systems. This process often utilizes commercial software tools like Saphire, Risk Spectrum, PTC Windchill Quality, Arbitr, etc. Practically each software tool allows calculating the dependability of complex systems subject to possible common cause failures (CCF). CCF are the associated failures of a group of several elements that occur simultaneously or within a short time interval (i.e. almost simultaneously) due to one common cause (e.g. a sudden change in the climatic service conditions, flooding of the premises, etc.). An associated failure is a multiple failure of several system elements, of which the probability cannot be expressed simply as the product of the probabilities of unconditional failures of individual elements. There are several generally accepted models used in CCF probability calculation: the Greek letters model, the alpha, beta factor models, as well as their variations. The beta factor model is the most simple in terms of associated failures simulation and further dependability calculation. The other models involve combinatorial search associated events in a group of n events, that becomes labor-consuming if the number n is large. Therefore, in the above software tools there are some restrictions on the n, beyond which the probability of CCF is calculated approximately. In the current R FaultTree package version there are no above CCF models, therefore all associated failures have to be simulated manually, which is not complicated if the number of associated events is small, as well as useful in terms of understanding the various CCF models. In this paper, for the selected diagram a detailed analysis of the procedure of associated failures simulation is performed for alpha and beta factor models. The* **Purpose***of this paper consists in the detailed analysis of the alpha and beta factor methods for a certain diagram, in the demonstration of fault tree creation procedure taking account of CCF using R's FaultTree package.* **Methods.** *R's FaultTree scripts were used for the calculations and FTA capabilities demonstration.* **Conclusions.** *Two examples are examined in detail. In the first example, for the selected block diagram that contains two groups of elements subject to associated failures, the alpha factor model is applied. In the second example, the beta factor model is applied. The deficiencies of the current version of FaultTree package are identified. Among the main drawbacks we should indicate the absence of some basic logical gates.*

Keywords: *fault tree, fault tree analysis, CCF, total cause failure, independent failures, dependent failures, antithetic events, alpha factor, beta factor.*

For citation: *Antonov AV, GalivetsEYu, Chepurko VA, Cherniaev AN. Fault tree analysis in the R programming environment. Accounting for common cause failures. Dependability 2018; 3: 3-9. DOI: 10.21683/1729-2646-2018-18-3-3-9*

Introduction

This paper is the continuation of [1] dedicated to the overview of the capabilities of the FaultTree package developed for the R programming environment. R is a programming language for statistical processing of graphics, as well as a free open-source programming environment developed as part of the GNU project. R supports a wide range of statistical and numerical methods and a large number of extension packages. Packages are libraries that support specific functions and subprograms or special applications. The paper continues the analysis of the capabilities of the package for creation, calculation and output of fault trees, the FaultTree package, in terms of the common cause failures (CCF).

Fault tree analysis (FTA) is a method of complex systems dependability analysis, in which the system failures are analyzed using the methods of Boolean algebra, summarizing the sequence of the subordinate events (low level failures) that cause the failure of the entire system. Sequences of random events are identified that may cause the system to fail, ways of reducing risks are defined and the rates of system failures are determined. In the most simple cases the fault trees form independent events. However, situations are possible when failures occur due to common causes, i.e. depend on a certain internal or external factor. Internal factors include general design, process and other internal causes, external factors include the effects of natural phenomena and/or human activity [2-4].

Calculations of CCF probabilities commonly involve various mathematical models that establish linear connection between the probabilities of dependent failure of a subset of elements affected to CCF with the probability of failure due to total causes. Failure due to total cause is essentially a complete group that includes independent failures of each element, CCF of two, three, etc. elements. The sufficiently simple, from the implementation point of view, beta factor model implies that in a set of elements exposed to CCF the failures can only be of two types: independent single failures of elements and dependent CCF of the entire group occurring simultaneously or almost simultaneously. In this case these events can be easily introduced into the fault tree manually. It should be taken into consideration that they must be incompatible, i.e. the connecting logical operations must make allowance for this fact. Under relatively low probabilities of failure, operator "or" can be used, while the calculation error is small.

The beta factor model is a special case of the more common Greek letters and alpha factor models. Let us note that the latter has several modifications. The basic difference between the generalized models and the beta factor model is that dependent failures can affect any subsets out of a set of elements affected by CCF. The choice of such subsets must be substantiated by the fact that their combination must cause system failure. It is clear that in this case we are dealing with a combinatorial enumeration of such situations, that, in case of small size of the set (two, three elements) can be done manually. However, if the set is large, computer technology has to be used, more precisely specialized software products: Windchill PTC, Risk Spectrum, Arbitr, etc. In the software tools there are some restrictions on the size of sets, beyond which the calculations are conducted approximately. That is due to the fact that as the size of the set of elements affected by CCF grows, the computational costs increase incommensurably.

As to the FaultTree package, its current version does not yet have CCF calculation models, therefore in the generalized models all enumerations have to be performed manually. That causes other problems associated with a deficiency in the required logical operations and/or event categories thatwill be covered in this article.

Let us examine some basic CCF capabilities supported by FaultTree.

Treatment of common cause failures

For the purpose of demonstrating the CCF capabilities, let us consider four different models: beta factor, alpha factor (with staggered and non-staggered tests) and the Greek letters model [5-7]. As the initial scheme let us consider the circuit shown in Figure 1 as per [1].

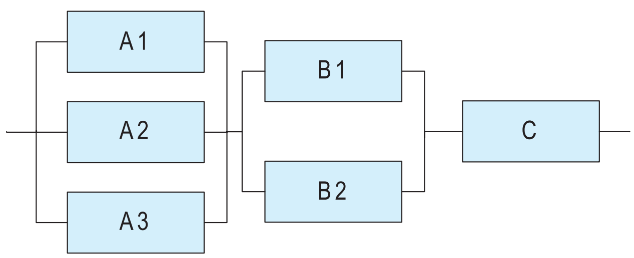

Figure 1. System diagram

Let us assume that the elements of group A (A1, A2, A3) and the elements of group B (B1, B2) may fail due to common causes. Let us introduce the following designations:

$I_1(A)$, $I_2(A)$, $I_3(A)$ are independent (single) failures of the elements of group A;

$C_{12}(A)$, $C_{23}(A)$, $C_{13}(A)$ are the CCFs of exactly two elements of group A;

$C_{123}(A)$ are CCFs of all thee elements of group A;

$I_1(B)$, $I_2(B)$ are independent failures of elements of group B;

$C_{12}(B)$ are CCFs of all the elements of group B;

$F(C)$ is the failure of element C.

The basic parametric model of CCF analysis determines the following events:

$$\begin{gathered} 1_t(A)= I_1(A)+ C_{12}(A)+C_{13}(A)+C_{123}(A): \\ 2_t(A)= I_2(A)+ C_{12}(A)+C_{23}(A)+C_{123}(A); \\ 3_t(A)= I_3(A)+ C_{13}(A)+C_{23}(A)+C_{123}(A); \\ 1_t(B)= I_1(B)+ C_{12}(B); \\ 2_t(B)= I_2(B)+ C_{12}(B). \end{gathered} \tag{1}$$

For instance, the first event will indicate a failure due to total causes related to the failure of the first element of group A. Let us designate the probabilities of such events:

$$Q_t(A)=\Pr(1_t(A))=\Pr(2_t(A))=\Pr(3_t(A));$$
$$Q_1(A)=\Pr(I_1(A))=\Pr(I_2(A))=\Pr(I_3(A));$$
$$Q_2(A)=\Pr(C_{12}(A))=\Pr(C_{13}(A))=\Pr(C_{23}(A)); \quad (2)$$
$$Q_3(A)=\Pr(C_{123}(A));$$
$$Q_t(B)=\Pr(1_t(B))=\Pr(2_t(B));$$
$$Q_1(B)=\Pr(I_1(B))=\Pr(I_2(B));$$
$$Q_2(B)=\Pr(C_{12}(B)).$$

Equations (2) are substantiated by that fact that hypothetically all elements of the same group are identical and are operated under identical conditions, and, therefore, their dependability indicators are identical as well.

Due to the incompatibility of events in the right part of each equations (1), we obtain:

$$Q_t(A)=Q_1(A)+2Q_2(A)+Q_3(A); \quad (3)$$
$$Q_t(B)=Q_1(B)+Q_2(B).$$

The probabilities of the right parts of equations (3) are determined differently depending on specific models.

Greek letters model

Thus, for the Greek letters model the following assumption is true:

$$Q_k^{(m)}=\left(\frac{1-p_{k+1}}{C_{m-1}^{k-1}}\prod_{i=1}^{k}p_i\right)Q_t,\ p_1=1,\ldots,p_{m+1}=0. \quad (4)$$

In our case if we designate: $p_2=\beta$, $p_3=\gamma$, from (4) easily follows:

$$Q_1^{(3)}(A)=(1-\beta(A))Q_t(A);$$
$$Q_2^{(3)}(A)=\frac{1}{2}\beta(A)(1-\gamma(A))Q_t(A); \quad (5)$$
$$Q_3^{(3)}(A)=\beta(A)\gamma(A)Q_t(A);$$
$$Q_1^{(2)}(B)=(1-\beta(B))Q_t(B);$$
$$Q_2^{(2)}(B)=\beta(B)Q_t(B).$$

Alpha factor model (not-staggered testing)

In this case the general formula for the probabilities is as follows:

$$Q_k^{(m)}=\left(\frac{k\alpha_k^{(m)}}{C_{m-1}^{k-1}\alpha_t}\right)Q_t,\ \text{where } \alpha_t=\sum_{k=1}^{m}k\alpha_k^{(m)}. \quad (6)$$

For groups of 3 and 2 events, we thus obtain:

$$Q_1^{(3)}(A)=\frac{\alpha_1^{(3)}}{\alpha_t^{(3)}}Q_t(A);$$
$$Q_2^{(3)}(A)=\frac{\alpha_2^{(3)}}{\alpha_t^{(3)}}Q_t(A); \quad (7)$$
$$Q_3^{(3)}(A)=\frac{3\alpha_3^{(3)}}{\alpha_t^{(3)}}Q_t(A);\ \textit{где}\ \alpha_t^{(3)}=\alpha_1^{(3)}+2\alpha_2^{(3)}+3\alpha_3^{(3)};$$
$$Q_1^{(2)}(B)=\frac{\alpha_1^{(2)}}{\alpha_t^{(3)}}Q_t(B);$$
$$Q_2^{(2)}(B)=\frac{2\alpha_2^{(2)}}{\alpha_t^{(2)}}Q_t(B);\ \textit{где}\ \alpha_t^{(2)}=\alpha_1^{(2)}+2\alpha_2^{(2)}.$$

Alpha factor model (staggered testing)

In this case the general formula for the probabilities is as follows:

$$Q_k^{(m)}=\left(\frac{\tilde{\alpha}_k^{(m)}}{C_{m-1}^{k-1}}\right)Q_t,\ \text{where } \sum_{k=1}^{m}\tilde{\alpha}_k^{(m)}=1. \quad (8)$$

For groups of 3 and 2 events, we thus obtain:

$$Q_1^{(3)}(A)=\alpha_1^{(3)}Q_t(A);$$
$$Q_2^{(3)}(A)=\frac{1}{2}\tilde{\alpha}_2^{(3)}Q_t(A); \quad (9)$$
$$Q_3^{(3)}(A)=\tilde{\alpha}_3^{(3)}Q_t(A)\ \textit{where}\ \tilde{\alpha}_1^{(3)}+\tilde{\alpha}_2^{(3)}+\tilde{\alpha}_3^{(3)}=1;$$
$$Q_1^{(2)}(B)=\tilde{\alpha}_1^{(2)}Q_t(B);$$
$$Q_2^{(2)}(B)=\tilde{\alpha}_2^{(2)}Q_t(B)\ \textit{where}\ \tilde{\alpha}_1^{(2)}+\tilde{\alpha}_2^{(2)}=1.$$

Beta factor model

One of the simplest CCF models is as follows:

$$Q_k^{(m)}=\begin{cases}(1-\beta)Q_t, & k=1;\\ 0, & 1<k<m;\\ \beta Q_t, & k=m.\end{cases} \quad (10)$$

In our case we obtain:

$$Q_1^{(3)}(A)=(1-\beta(A))Q_t(A);$$
$$Q_2^{(3)}(A)=0; \quad (11)$$
$$Q_3^{(3)}(A)=\beta(A)Q_t(A);$$
$$Q_1^{(2)}(B)=(1-\beta(B))Q_t(B);$$
$$Q_2^{(2)}(B)=\beta(B)Q_t(B).$$

It is not difficult to show that by substituting (5), (7), (9), (11) into (3) an identical equation is obtained: $Q_t(A)=Q_t(A)$, $Q_t(B)=Q_t(B)$, however, this will be definitely true under large m as well. Thus, the difference between the approaches employed by the models consists only in the different understanding of the correlations between the probabilities $Q_1^{(m)},Q_2^{(m)},\ldots,Q_m^{(m)}$. Frequently, different models may provide sufficiently close results. For that purpose transfer equations can be used [5] (see Table A-2-A-4 of annex A). In addition, [4] (Table 5.11, p. 75) provides reference statistical information of the parameters $\tilde{\alpha}_k^{(m)}$ for the alpha factor model (8). Thus, for parallel series of two elements B1, B2 sample medians of the parameters (50% of point) are equal respectively

$$med\left(\tilde{\alpha}_1^{(2)}\right)=0,953,\ med\left(\tilde{\alpha}_2^{(2)}\right)=0,047. \quad (12)$$

For subseries A of three elements A1, A2, A3

$$med\left(\tilde{\alpha}_1^{(3)}\right)=0,9500,\ med\left(\tilde{\alpha}_2^{(3)}\right)=0,0242,$$
$$med\left(\tilde{\alpha}_3^{(3)}\right)=0,0258. \quad (13)$$

Let us take these numbers as the values of the parameters of model (8). Probabilities (9) will be as follows

$$\begin{aligned} Q_1^{(3)}(A) &= 0,9500 \cdot Q_t(A); \\ Q_2^{(3)}(A) &= 0,0121 \cdot Q_t(A); \\ Q_3^{(3)}(A) &= 0,0258 \cdot Q_t(A); \\ Q_1^{(2)}(B) &= 0,953 \cdot Q_t(B); \\ Q_2^{(2)}(B) &= 0,047 \cdot Q_t(B). \end{aligned} \quad (14)$$

Flow tables can be used, but it is not difficult to guess, that in model (5)

$$\beta(A) = 0,05, \gamma(A) = \frac{0,0258}{\beta(A)} = 0,516, \beta(B) = 0,047.$$

In the alpha factor model (staggered testing), a simple transformation provides the following result:

$$\alpha_1^{(3)} = 0,95, \alpha_2^{(3)} = 0,0121, \alpha_3^{(3)} = 0,0086, \alpha_t^{(3)} = 1,$$

$$\alpha_1^{(2)} = 0,953, \alpha_2^{(2)} = 0,0235, \alpha_t^{(2)} = 1.$$

Under the deduced values of the parameters the results of both the alpha factor and Greek letters models will provide identical results. For the sufficiently rough, yet simpler beta factor model the results will be different, since the beta factor model uses only one input parameter. Nevertheless, let us take it identical to the corresponding Greek letter, 0.05.

Now let us proceed to the calculations. In order to simplify the fault tree let us avoid using different dependability models for different elements, but assume that the probabilities of failure of elements A, B and C are respectively

$$Q_t(A) = 0.3, Q_t(B) = 0,2, Q_t(C) = 0,1. \quad (15)$$

The probability of failure without regard to the CCF will be equal to:

$$Q_S = 1 - \left(1 - Q_t^3(A)\right)\left(1 - Q_t^2(B)\right)\left(1 - Q_t(C)\right) = 0,1593. \quad (16)$$

Let us perform calculations taking the CCF into account. The circuit will fail under the following combinations of events presented as eight minimum sections:

$$\begin{gathered} \{I_1(A) \cap I_2(A) \cap I_3(A)\}, \{I_1(A) \cap C_{23}(A)\}, \\ \{I_2(A) \cap C_{13}(A)\}, \{I_3(A) \cap C_{12}(A)\}, \{C_{123}(A)\}, \quad (17) \\ \{I_1(B) \cap I_2(B)\}, \{C_{12}(B)\}, \{F(C)\}. \end{gathered}$$

Let us compose the calculation script. Unlike in the the specialized packages mentioned above, in the current version of the package under consideration CCF is not taken account of, therefore all the events of (17) have to be developed and introduced manually. Let us note that in (17) there is a group of incompatible (thus dependent) sections, for example, the first and the second, the first and the third, etc. There is also a group of independent sections, for example, $\{I_1(A) \cap I_2(A) \cap I_3(A)\}$ and $\{C_{12}(B)\}$, i.e. sections belonging to different CCF groups. Correct calculation of the probabilities of failure of this group requires using the specialized logic node "or" that calculates the probability of a sum of antithetical events. On the other hand, an additional type can be introduced for the group of incompatible events contained in one CCF group. Probably, the optimal solution consists in the development of a module for taking account of CCF, that, probably without a graphic representation in the fault tree, would automatically and correctly calculate the dependability indicators when highlighting CCF event groups and selecting the appropriate model. Unfortunately, such capabilities are not yet implemented in R. Therefore, in the calculation we will be using regular "or".

Example 1. CCF. Alpha factor model

```
library(FaultTree)
tree4 <- ftree.make(type="or", name="Example 4.", name2="CCF")
tree4 <- addLogic(tree4, at=1, type="and", name="I1(A)*I2(A)*I3(A)")
tree4 <- addLogic(tree4, at=2, type="inhibit", name="Independent", name2="failure Ai")
tree4 <- addProbability(tree4, at=3, prob=.95, name="Parameter", name2="models")
tree4 <- addProbability(tree4, at=3, prob=.3, name="Failure Ai", name2="(total)")
tree4 <- addDuplicate(tree4, at=2, dup_id=3)
tree4 <- addDuplicate(tree4, at=2, dup_id=3)
tree4 <- addLogic(tree4, at=1, type="and", name="Ii(A)*Cjk(A)")
tree4 <- addDuplicate(tree4, at=12, dup_id=3)
tree4 <- addLogic(tree4, at=12, type="inhibit", name="CCF", name2="failure Aj, Ak")
tree4 <- addProbability(tree4, at=16, prob=.0121, name="Parameter", name2="models")
tree4 <- addProbability(tree4, at=16, prob=.3, name="Failure Ai", name2="(total)")
tree4 <- addDuplicate(tree4, at=1, dup_id=12)
tree4 <- addDuplicate(tree4, at=1, dup_id=12)
tree4 <- addLogic(tree4, at=1, type="inhibit", name="CCF C123(A)", name2="failure A1,A2,A3")
tree4 <- addProbability(tree4, at=33, prob=.0258, name="Parameter", name2="models")
tree4 <- addProbability(tree4, at=33, prob=.3, name="Failure Ai", name2="(total)")
tree4 <- addLogic(tree4, at=1, type="and", name="I1(B)*I2(B)")
```

```
  tree4 <- addLogic(tree4, at=36,
type=”inhibit”, name=”Independent”,
name2=”failure Bi”)
  tree4 <- addProbability(tree4,
at=37, prob=.953, name=”Parameter”,
name2=”models”)
  tree4 <- addProbability(tree4, at=37,
prob=.2, name=”Failure Bi”, name2=”(total”)
  tree4 <- addDuplicate(tree4, at=36, dup_
id=37)
  tree4 <- addLogic(tree4, at=1, type=”inhibit”,
name=”CCF C12(B)”, name2=”failure B1,B2”)
  tree4 <- addProbability(tree4,
at=43, prob=.047, name=”Parameter”,
name2=”models”)
  tree4 <- addProbability(tree4, at=43, prob=.2,
name=”Failure Bi”, name2=”(total)”)
  tree4 <- addProbability(tree4, at=1,
prob=.1, name=”Failure C”, name2=”(total)”)
  tree4 <- ftree.calc(tree4)
  ftree2html(tree4, write_file=TRUE)
  browseURL(“tree4.html”)
```

We will provide no detailed comments regarding this script. Let us focus on lines nos. 4, 11, …. When a logical elements is added, an inhibitory gate is used. As it is known [5-7], in this case the output event occurs, if both input events occur, one of which is a restraint event. The role of condition is performed by the coefficient of the alpha, beta factor or Greek letters model, as these coefficients really play the role of conditional probabilities.

It would appear that calculating the beta factor model two insignificant corrections would suffice. In the 12-th line the probability of 0 and in the 17-th line the probability of 0.05 would need to be specified. However, in this case the fault tree calculation results in an error due to the fact that one of the probabilities is equal to 0. Most probably, in the future this error will be corrected. For now, at least two approaches are possible. One of them consists in specifying zero probability as extremely low. The other one is to remove the branches with a zero probability. The following example demonstrates this exact approach.

Example 2. CCF. Beta factor model

```
  library(FaultTree)
  tree4 <- ftree.make(type=”or”, name=”Example
4.”, name2=”CCF”)
  tree4 <- addLogic(tree4, at=1, type=”and”,
name=”I1(A)*I2(A)*I3(A)”)
  tree4 <- addLogic(tree4, at=2, type=”inhibit”,
name=”Independent”, name2=”failure Ai”)
  tree4 <- addProbability(tree4, at=3,
prob=.95, name=”Parameter”, name2=”models”)
  tree4 <- addProbability(tree4, at=3, prob=.3,
name=”Failure Ai”, name2=”(total)”)
  tree4 <- addDuplicate(tree4, at=2, dup_
id=3)
  tree4 <- addDuplicate(tree4, at=2, dup_
id=3)
  tree4 <- addLogic(tree4, at=1,
type=”inhibit”, name=”CCF C123(A)”,
name2=”failure A1,A2,A3”)
  tree4 <- addProbability(tree4, at=12,
prob=.05, name=”Parameter”, name2=”models”)
  tree4 <- addProbability(tree4, at=12, prob=.3,
name=”Failure Ai”, name2=”(total)”)
  tree4 <- addLogic(tree4, at=1, type=”and”,
name=”I1(B)*I2(B)”)
  tree4 <- addLogic(tree4, at=15,
type=”inhibit”, name=”Independent”,
name2=”failure Bi”)
  tree4 <- addProbability(tree4,
at=16, prob=.953, name=”Parameter”,
name2=”models”)
  tree4 <- addProbability(tree4, at=16,
prob=.2, name=”Failure Bi”, name2=”(total”)
  tree4 <- addDuplicate(tree4, at=15, dup_
id=16)
  tree4 <- addLogic(tree4, at=1, type=”inhibit”,
name=”CCF C12(B)”, name2=”failure B1,B2”)
  tree4 <- addProbability(tree4,
at=22, prob=.047, name=”Parameter”,
name2=”models”)
  tree4 <- addProbability(tree4, at=22, prob=.2,
name=”Failure Bi”, name2=”(total)”)
  tree4 <- addProbability(tree4, at=1,
prob=.1, name=”Failure C”, name2=”(total)”)
  tree4 <- ftree.calc(tree4)
  ftree2html(tree4, write_file=TRUE)
  browseURL(“tree4.html”)
```

Let us conduct calculations analytically. First, let us calculate the alpha factor and Greek letters models. By fitting model coefficient we obtained identical results. By virtue of (2) and independence of events, the precise probability of failure due to all causes (both common causes, and independently) will be equal to

$$Q_{S(CCF)}=1-Pr\left(\bar{A}\cap\bar{B}\cap\bar{C}\right)=1-Pr\left(\bar{A}\right)Pr\left(\bar{B}\right)Pr\left(\bar{C}\right), \quad (18)$$

where events $\bar{A}$ ={elements of group A did not fail}, $\bar{B}$ = {elements of group B did not fail}, $\bar{C}$ = {elements of the group C did not fail}.

Since the independent failures and common cause failures are incompatible, thus mutually dependent, then

$$\begin{cases} Pr\left(\bar{A}\right)=1-Q_1^3\left(A\right)-3Q_1\left(A\right)Q_2\left(A\right)-Q_3\left(A\right), \\ Pr\left(\bar{B}\right)=1-Q_1^2\left(B\right)-Q_2\left(B\right). \end{cases} \quad (19)$$

Numerical value $Q_{S(CCF-\alpha)}$ = 0.170350.Calculated approximate value $Q_{S(CCF-\alpha)}$ = 0.16981 (see Fig. 2).The figure shows the incomplete fault tree with a number of “collapsed” branches due to its awkwardness.

The logical node “or” does not take into consideration the fact of dependence of minimum sections in (17) and calculates $Pr\left(\bar{A}\right)$ and $Pr\left(\bar{B}\right)$ using the following formulas:

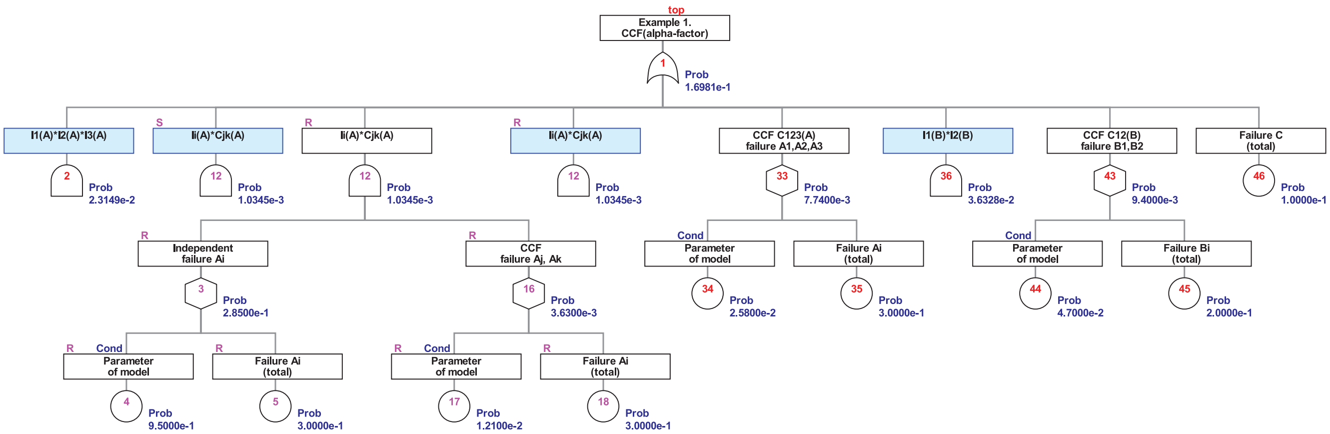

Figure 2. Fault tree for example 4 (alpha factor model)

$$\begin{cases} Pr(\bar{A}) = (1-Q_1^3(A))(1-Q_1(A)Q_2(A))^3(1-Q_3(A)), \\ Pr(\bar{B}) = (1-Q_1^2(B))(1-Q_2(B)). \end{cases} \quad (20)$$

For the beta factor model (10) the precise formula of probabilities calculation $Pr(\bar{A})$ and $Pr(\bar{B})$ will be somewhat simpler:

$$\begin{cases} Pr(\bar{A}) = 1-Q_1^3(A)-Q_3^*(A), \\ Pr(\bar{B}) = 1-Q_1^2(B)-Q_2(B). \end{cases} \quad (21)$$

Approximation formula:

$$\begin{cases} Pr(\bar{A}) = (1-Q_1^3(A))(1-Q_3^*(A)), \\ Pr(\bar{B}) = (1-Q_1^2(B))(1-Q_2(B)). \end{cases} \quad (22)$$

In (21) and (22), $Q_3^*(A) = 0.05Q_t(A)$.Precise and approximated values $Q_{S(CCF-\beta)}$ = 0,17392 and 0,17333 respectively. The approximate probability matches the estimated one (see Fig. 3).

As expected, the beta factor model turned out to be more pessimistic.

Figure shows the fault tree with a number of "collapsed" branches due to its awkwardness.

In conclusion, it should be noted the current version of FaultTreehas several essential deficiencies of in terms of its applicability in complex systems dependability calculation.

The selection of logical operations (gates) for work with events is quite limited. Thus, there are no modules "mutually excluding or", "priority and", "negation", etc. This substantially limits the package's capabilities.

The package does not allow duplicating a basic event in different branches of a tree. The addDuplicate() script only simplifies the construction of complex event trees while duplicating branches, structure. Yet it is impossible to take into consideration the fact of dependence, incompatibility of events. It is not possible to "manually", with the help of the available logical operations, create a tree that would contain such events. This was covered above.

Additional scripts for different models (alfa, beta factor, Greek letters, etc.) could significantly help taking account of CCF.

The wide range of tools of the R language allows for more flexible setting of calculations and unlike the rigid schemes of specialized packages enables independently performing certain procedures with input data. And, certainly, the most important advantage of R is that unlike the specialized packages intended for the analysis of event tree only it provides by far more capabilities to perform data analysis procedures.

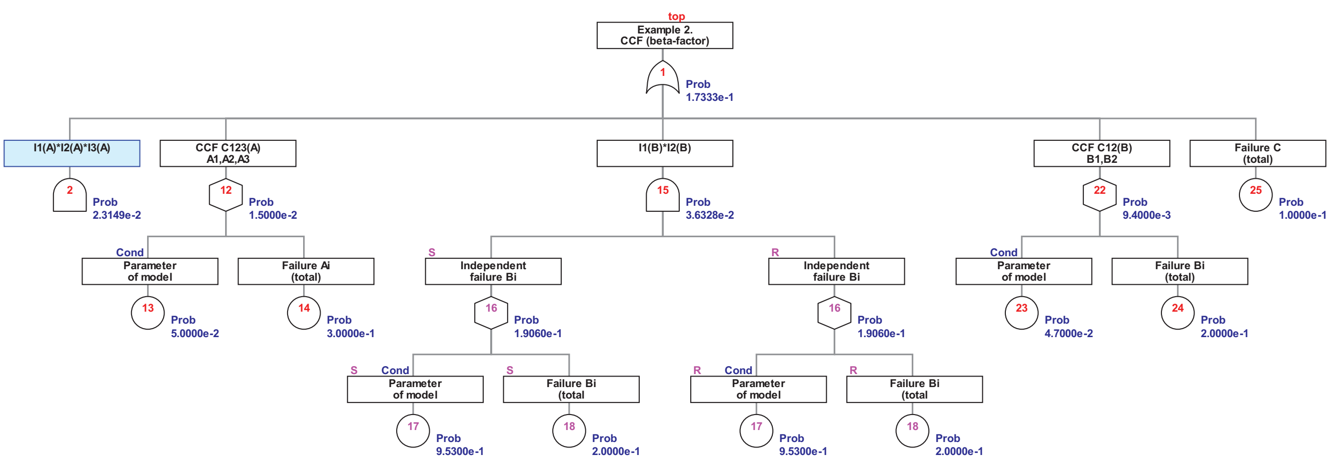

Figure 3. Fault tree for example 4 (beta factor model)

Thus, the vital improvement of FaultTree package aiming to eliminate the above shortcomings will indeed provide experts with a powerful tool for not only fault trees analysis, but also for more advanced statistical analysis. As to the further development of package, it should also be improved in terms of development of functionality related to the calculation of various importance factors according to Birnbaum, Fussell-Vesely, etc., uncertainty analysis.

Conclusion

This paper is dedicated to the demonstration of the fault tree construction and analysis capabilities of the actively developing statistical computing language R and its FaultTree package. Fault trees are used for dependability analysis of complex systems. The paper sets forth and analyses in detail some models of CCF management, two examples are given. In the first example, CCF is taken account of per alpha factor model. The second example is dedicated to the beta factor model. The deficiencies and optimal development strategy of the FaultTree package are identified.

References

[1]. Antonov AV, Galivets EYu, Chepurko VA, Cherniaev AN. Fault tree analysis in the R programming environment. Dependability 2018;(1):4-13. DOI: 10.21683/1729-2646-2018-18-1-4-13

[2]. Pereguda AI, Pereguda AA, Timashev D.A. The mathematical model of computer networks' reliability. Dependability 2013;(4):31-43. DOI:10.21683/1729-2646-2013-0-4-18-43

[3]. Alpeev AS. Dependability of control systems software and safety of nuclear power plants. Dependability 2015;(4):78-80. DOI:10.21683/1729-2646-2015-0-4-75-80

[4]. Ostreykovsky VA. Shvyriaev YuV. Bezopasnost' atomnykhstantsiy. Veroyatnostnyyanaliz [Safety of nuclear powerplants. Probabilistic analysis].Moscow: Fizmatlit; 2008 [in Russian].

[5]. Mosleh A et al. Procedures Guidelines in Modeling Common Cause Failures in Probabilistic Risk Assessment (NUREG/CR-5485); 1998.

[6]. Programmnyy kompleks avtomatizirovannogo strukturno-logicheskogo modelirovaniya i rascheta nadezhnosti i bezopasnosti ASUTP na stadii proektirovaniya (PK ASM SZMA) [Software system for automated structural and logical modeling and dependability and safety calculation of ACS at the design stage (PK ASM SZMA)]. Technical documentation. Saint-Petersburg: OAO SPIK SZMA; 2003.

[7]. Smith CL, Wood ST, Galyean WJ, Schroeder JA, Sattison MB. Systems Analysis Programs for Hands-on Integrated Reliability Evaluations (SAPHIRE). Version 8. Vol. 2, NUREG/CR-7039 INL/EXT-09-17009; June 2011.

About the authors

Alexander V. Antonov, Doctor of Engineering, Professor, Chief Expert, Division for Justifying Calculations of Design Solutions, JSC RZSU, e-mail: AlVlaAntonov@rasu.ru

Evgeny Yu. Galivets, Deputy Director of Department, Head of Design Division, JSC RASU, e-mail:EYGalivets@rasu.ru

Valery A. Chepurko, Candidate of Physics and Mathematics, Associate Professor, Chief Specialist, Division for Justifying Calculations of Design Solutions, JSC RZSU, e-mail: VAChepurko@rasu.ru

Alexey N. Cherniaev, Candidate of Engineering, Deputy Technical Director, Director of Design Department, JSC RASU, e-mail: AlNChernyaev@rasu.ru

Received on: 19.03.2018