

# **ETMA: A New Software for Event Tree Analysis with Application to Power Protection**

Mohamed Abdelghany, Waqar Ahmad, Sofiène Tahar, and Sowmith Nethula

Department of Electrical and Computer Engineering,  
Concordia University, Montréal, QC, Canada

{m\_eldes,waqar,tahar,s\_nethul}@ece.concordia.ca

**TECHNICAL REPORT**

June 2020

arXiv:2006.12383v1 [eess.SP] 18 Jun 2020

## Abstract

Event Tree (ET) analysis is a widely used forward deductive safety analysis technique for decision-making at a system design stage. Existing ET tools usually provide Graphical Users Interfaces (GUI) for users to manually draw system level ET diagrams, which consist of nodes and branches, describing all possible success and failure scenarios. However, these tools do not include some important ET analysis steps, e.g., the automatic generation and reduction of a complete system ET diagram. In this paper, we present a new *Event Trees Modeling and Analysis* ( $\mathcal{ETMA}$ ) tool to facilitate users to conduct a complete ET analysis of a given system. Some key features of  $\mathcal{ETMA}$  include: (i) automatic construction of a complete ET model of real-world systems; (ii) deletion/reduction of unnecessary ET nodes and branches; (iii) partitioning of ET paths; and (iv) probabilistic analysis of the occurrence of a certain event. For illustration purposes, we utilize our  $\mathcal{ETMA}$  tool to conduct the ET analysis of a protective fault trip circuit in power grid transmission lines. We also compared the  $\mathcal{ETMA}$  results with Isograph, which is well-known commercial tool for ET analysis.

**Keywords**— Event Tree, Modeling, Analysis, Python, Isograph, Power Grid Transmission Lines.

## 1 Introduction

Nowadays, the fulfillment of stringent safety requirements for highly critical systems, which are prevalent, e.g., in smart grids and autonomous systems, has been encouraging safety design engineers to utilize dependability analysis techniques as per recommendations of the safety standards, such as IEC 61850 [1] and ISO 26262 [2]. Event Tree (ET) analysis is a well-known dependability analysis technique that enumerates all possible combinations of component states and external events and thus provides a detailed system view [3]. The building of a graphical diagram of a system ET model starts with an initiating node and sequentially drawing all the system components and their operating states [4]. In the ET analysis, the probabilistic assessment of the occurrence of a certain event can be used for decision-making at the systems design stage. The results of the ET analysis are extremely useful for safety analysts to quantify systems improvement.

Existing commercially available ET tools, such as ITEM [5], Isograph [6], and EC Tree [7], provide many powerful features, including user-friendly editors, a commonly used events library and the coloring of diagram elements for easier viewing. For instance, the EC Tree tool, which is developed by NASA's IM&S (Integrated Modeling and Simulation) Team, provides an Excel sheet for potential users. It can be easily used to input a given system ET model with a little training. All these tools require a system ET diagram from the user, which is then followed by assigning the probability to each branch of an ET diagram. Prior to utilizing these tools for ET analysis, the users must draw a given system ET diagram manually, may be on a paper. However, this manual approach may introduce errors from the start since an ET diagram

becomes significantly large as the number of system components and their operational states increases. Moreover, an important feature of partitioning an ET with respect to an event occurrence and then to calculate its corresponding probability is not available in any existing ET analysis tools.

To overcome the above-mentioned limitations of existing ET tools, we develop a new Event Trees Modeling and Analysis ( $\mathcal{ETMA}$ ) tool. It is mainly inspired from the work of Papazoglou [8], who was among the first ones to describe the sound mathematical foundations of ET analysis during late 90's. The development of  $\mathcal{ETMA}$  starts from a recursive function describing the pattern of generating an ET diagram from the given list of all possible failure and success modes of given system components. Most importantly,  $\mathcal{ETMA}$  offers a reduction feature, which deletes unnecessary nodes and branches from the automatically generated ET diagram and return an ET model representing the actual behavior of a given system.  $\mathcal{ETMA}$  has an intriguing feature of partitioning the ET paths according to the system components failure and success modes. It also provides the probabilistic analysis feature by allowing users to assign the probability to each components states. Moreover, the  $\mathcal{ETMA}$  results can be used to identify critical components and make decisions about adding redundancy in a system. All these  $\mathcal{ETMA}$  features are developed in the Python programming language [9], which offers extensive built-in libraries for displaying, list manipulations and arithmetic calculations.

It is worth mentioning that our  $\mathcal{ETMA}$  tool can handle large and complex real-world systems with an arbitrary number of system components and their operating states. For illustration purposes, we utilize  $\mathcal{ETMA}$  to conduct the ET analysis of a protective trip circuit in power grid transmission lines consisting of several critical components, such as relays and current transformers[10].

Our main novel contributions in this paper are as follows:

- Automatic generation of complete system ET model from a given list of system components and their operating states
- Deletion of unnecessary nodes and branches to generate a reduced ET model
- Partitioning of ET with respect to an event occurrence for probabilistic analysis
- Implementation in Python of a comprehensive tool for ET modeling and analysis:  $\mathcal{ETMA}$
- Step-wise ET analysis of a protection fault trip circuit in power grid transmission lines with a decision analysis to add redundancy for some critical components
- Comparison between the results of  $\mathcal{ETMA}$  with the commercial Isograph ET analysis software

The rest of the paper is organized as follows: In Section II, we briefly summarize the fundamentals of ETs and the theoretical foundations of the  $\mathcal{ETMA}$  tool. Section III describes the ET modeling and analysis features of  $\mathcal{ETMA}$ . Section IV presents the step-wise process of ET analysis of a protective trip circuit in power grid transmission lines using  $\mathcal{ETMA}$  and the decision analysis of the trip circuit critical components based on  $\mathcal{ETMA}$  results. Section V provides a comparison between  $\mathcal{ETMA}$  and the Isograph software. Lastly, Section VI concludes the paper.

## 2 Event Trees

An ET diagram starts by a single initiating event called *Node* and then all possible outcomes of an event are drawn as branches. This process is continuously repeated in the forward direction until all event nodes and their branches are drawn resulting in a complete ET diagram of the system. Fig. 1 depicts a generic ET diagram.

Nodes model the occurrence of different possibilities for an event or modes of operation of system components, which is known as event outcome space in the literature [8]. A *Node* is usually represented by a circle with multiple line segments. For instance, in Fig. 1, X, Y and Z are nodes. Branches originating from a node represent each of the next possible component states. A *Branch* is usually designated by a line segment associated with a preceding node. For instance,  $X_1, \dots, X_N$  and  $Y_1, \dots, Y_M$  are branches, as shown in Fig. 1. A complete ET diagram draws all possible paths that represent a specific system. Each path consists of a unique sequence of events, i.e.,  $(X_N Y_2 Z_1 \dots)$  is one of the ET paths in Fig. 1.

The probability of each path in an ET diagram is evaluated for decision-making at the systems design stage. These probabilities represent the likelihood of each outcome or condition that can happen in a system. The assessment of these probabilities depends upon the occurrence of previous events in an ET. The probability of each path is usually computed by multiplying the probabilities of events associated with all nodes in a path. For example, the probability of the path  $(X_N Y_2 Z_1 \dots)$  in Fig. 1 is expressed mathematically as:

$$\mathcal{P}(X_N Y_2 Z_1 \dots) = \mathcal{P}(X_N) * \mathcal{P}(Y_2) * \mathcal{P}(Z_1) * \dots \quad (1)$$

Also, all events in a path including the initiating node are assumed to be mutually exclusive. This implies that the cumulative probability of all branches connected to a certain node must be equal to 1 as:

$$\sum_{i=1}^N \mathcal{P}(X_i) = 1, \sum_{j=1}^M \mathcal{P}(Y_j) = 1, \sum_{h=1}^K \mathcal{P}(Z_h) = 1, \dots \quad (2)$$

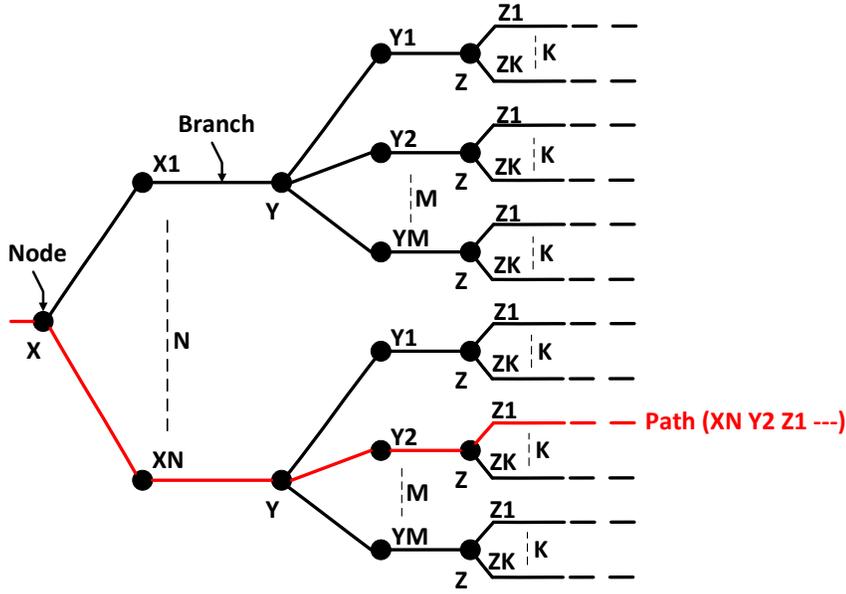


Figure 1: A generic ET diagram

## 2.1 Theoretical Foundation

The underline mathematics of ET analysis in  $\mathcal{ETMA}$  are mainly inspired from the work of Papazoglou [8] that are briefly described as follows:

An event outcome space ( $\mathcal{W}$ ) is referred to as a list of all possible outcomes of an event. Each node of an ET is associated with an event outcome space must satisfy following constraints

1. *Distinct*: All outcomes in an event outcome space must be unique.
2. *Disjoint (mutually exclusive)*: Any pair of events from a set of events outcome space cannot occur at the same time.
3. *Complete*: An event outcome space must contain all possible events that can occur.
4. *Finite*: An event outcome space must consists of a finite number of elements.

$$\mathcal{W} = [\omega_j] \quad j = 1, 2, \dots, \mathcal{N} \quad (3)$$

Consider a system having two events, say  $E_1$  and  $E_2$ , with two event outcome spaces  $\mathcal{W}_1$  and  $\mathcal{W}_2$ , respectively. The Cartesian product ( $\otimes$ ) of these event outcome spaces returns a list of  $(\mathcal{N}_1 \times \mathcal{N}_2)$  pairs containing all possible outcome pairs for the occurrence of  $E_1$  and  $E_2$  together (i.e.,  $\mathcal{W}_1 \otimes \mathcal{W}_2$ ). In ET, the resulting event outcome space from the Cartesian product of two event outcome spaces must also satisfy the above-mentioned constraints. We program this concept in  $\mathcal{ETMA}$  in two steps as follows:

*Step 1:* We first construct a list of pairs by taking each element from the event outcome spaces  $\mathcal{W}_1$  and  $\mathcal{W}_2$ .

*Step 2:* We ensure that the obtained duets from Step 1 are mutually exclusive. For instance, consider two arbitrary outcomes  $(\omega_{1m} \omega_{2n})$  and  $(\omega_{1k} \omega_{2l})$ , at least  $(m \neq k)$  or  $(n \neq l)$  must be true.

One of our main objectives, in this work, is to take an arbitrary list of given system components with their operating states and automatically generate the corresponding ET diagram (i.e.,  $\mathcal{W}_1 \otimes \mathcal{W}_2 \otimes \dots \otimes \mathcal{W}_N$ ). For this purpose, we developed a Python function in that can recursively perform *Steps 1* and *2* on a given list of event outcome spaces representing the system components and their operational states.

To present a clear understanding of the above-mentioned automatic ET generation feature of  $\mathcal{ETMA}$ , consider a system having three events, say  $E_1$ ,  $E_2$  and  $E_3$ , with three event outcome spaces  $\mathcal{W}_1$ ,  $\mathcal{W}_2$  and  $\mathcal{W}_3$ , respectively. The resulting ET diagram is shown in Fig. 2 and the collection of all possible ET paths in a list of strings are as:

$$\begin{aligned} Path_0 &= [A_1, B_1, C_1], Path_1 &= [A_1, B_1, C_2], \\ Path_2 &= [A_1, B_2, C_1], Path_3 &= [A_1, B_2, C_2], \\ Path_4 &= [A_2, B_1, C_1], Path_5 &= [A_2, B_1, C_2], \\ Path_6 &= [A_2, B_2, C_1], Path_7 &= [A_2, B_2, C_2], \\ Path_8 &= [A_3, B_1, C_1], Path_9 &= [A_3, B_1, C_2], \\ Path_{10} &= [A_3, B_2, C_1], Path_{11} &= [A_3, B_2, C_2] \end{aligned}$$

The order of the outcomes in a path is irrelevant when evaluating the probabilities of a given path [11], i.e., the probability of path  $[A_3, B_1, C_2]$  is  $\mathcal{P}(A_3) * \mathcal{P}(B_1) * \mathcal{P}(C_2)$ , which is exactly equivalent to the probability of path  $[C_2, B_1, A_3]$  due to the commutative property of multiplication. However, in many cases, it is useful to preserve the order of outcomes in the ET paths. For instance, it can facilitate the thinking process in certain critical situations, but it has no relation to the dynamic of the system components [8]. Another benefit of introducing a sequence is that, in some critical systems, if the main component fails, then the probability of this path depends on the failure of the main component only without considering the next components state. Therefore, we believe that a sequence preserving generation of ETs must be adopted.

## 2.2 Branch or Node Deletion

During ET analysis, we may require to model the exact logical behavior of systems in the sense that the irrelevant nodes and branches should be removed from a complete ET of a system. This can be done by deleting some specific branch or nodes corresponding to the occurrence of certain events, which are known as *Complete Cylinders* (CCs). These cylinders are ET paths consisting of  $\mathcal{N}$  events and they are conditional on the occurrence of  $\mathcal{K}$  events in their respective paths and not conditional on the occurrence of the remaining  $(\mathcal{N} - \mathcal{K})$  events [11]. These cylinders are also referred to as CCs with respect to  $\mathcal{K}$  *Conditional Events* (CEs).

A reduced ET can be obtained in two ways: (1) eliminate certain branches with all their successor nodes; and (2) delete only nodes from specific branches leaving the successor nodes connected to these branches. The reduction process can be explained as follows:

### 2.2.1 Branch Deletion

If the paths {8; 9; 10; 11}, shown in Fig. 2, are CCs with respect to the event  $A_3$  (i.e., not conditional on the occurrence of neither  $\mathcal{W}_2$  nor  $\mathcal{W}_3$  event outcome spaces), then the branches  $[A_3, B_1]$  and  $[A_3, B_2]$  should be deleted. The resulting ET paths after deletion are as follows:

$$\begin{aligned} Path_0 &= [A_1, B_1, C_1], Path_1 = [A_1, B_1, C_2], \\ Path_2 &= [A_1, B_2, C_1], Path_3 = [A_1, B_2, C_2], \\ Path_4 &= [A_2, B_1, C_1], Path_5 = [A_2, B_1, C_2], \\ Path_6 &= [A_2, B_2, C_1], Path_7 = [A_2, B_2, C_2], \\ Path_8 &= [A_3] \end{aligned}$$

### 2.2.2 Node Deletion

If the paths {0; 1; 2; 3}, shown in Fig. 2, are CCs with respect to the event  $A_1$  and  $\mathcal{W}_3$  event outcome space (i.e., not conditional on the occurrence of  $\mathcal{W}_2$  event outcome space only), then the node  $\mathcal{W}_2$  from the branch  $A_1$  needs to be deleted only while keeping the  $\mathcal{W}_3$  event outcome space connected with the event  $A_1$ . The resulting ET paths after deletion are as follows:

$$\begin{aligned} Path_0 &= [A_1, C_1], Path_1 = [A_1, C_2], \\ Path_2 &= [A_2, B_1, C_1], Path_3 = [A_2, B_1, C_2], \\ Path_4 &= [A_2, B_2, C_1], Path_5 = [A_2, B_2, C_2], \\ Path_6 &= [A_3] \end{aligned}$$

## 3 Event Tree Analysis in $\mathcal{ETMA}$

The flowchart describing the  $\mathcal{ETMA}$  tool for ETs modeling and analysis is depicted in Fig. 3 and mainly consists of 4 steps as follows: (1) identify the given system components and their operating states representing the behavior of the system, then automatically generate a complete ET model describing all system components states and also produce a complete outcome space with all possible scenarios of different levels of failure and success; (2) optionally, reduce manually some nodes/branches from the generated complete ET diagram to regenerate a smaller model exhibiting the exact behavior of the given system; (3) partition the ET paths according to the system components failure and success modes; and (4) evaluate the probability of occurrence for certain events in the system after partitioning the ET paths.

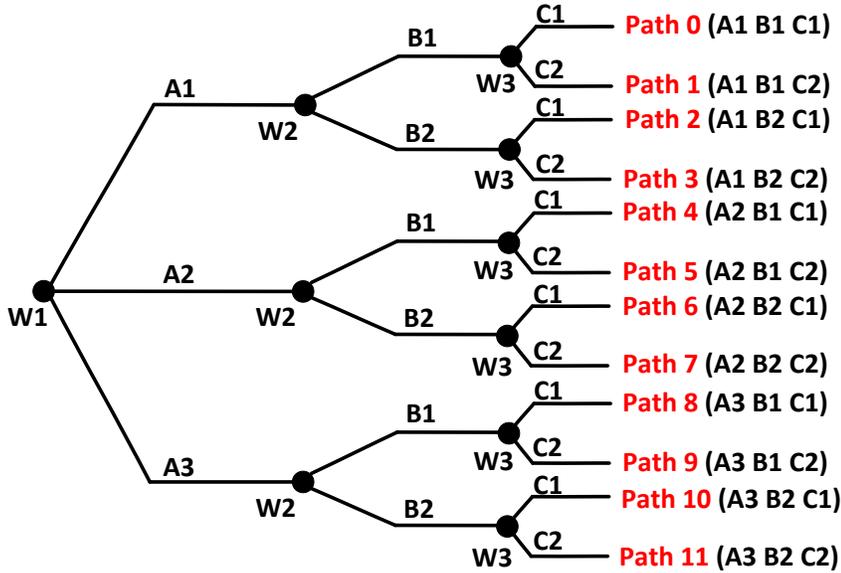


Figure 2: ET represents the event space outcomes

The details of the  $\mathcal{ETMA}$  functions that perform the above-mentioned operations are described in Algorithm 1. We provide pop-up input windows for each of these functions in order to facilitate the users interaction with the  $\mathcal{ETMA}$  tool. It can be noticed from Algorithm 1 that the reduction  $\mathcal{ETMA}$  feature can be bypassed, in case the deletion of nodes or branches is not required. Also, to ensure that  $\mathcal{ETMA}$  is capable of generating complex and scalable ETs, we have implemented the steps of Algorithm 1 using the *PyGraphviz* Python package [12], which provides several methods for layout and drawing of complex graphs.

In reliability engineering, the decisions to add redundancy for critical components or functions in a system are very crucial since it significantly increases the cost of the system. Redundancy is often used in the form of a backup or fail-safe in order to improve actual system performance. Decision tree [13], is a tree-like model that helps safety engineers to conduct decision analysis and make effective decisions, like adding redundancy to critical system components. Fig. 3 shows the procedure of making a decision for redundancy of a critical component in a system. If the level of the probabilistic analysis evaluated from the ET model is satisfied, then this component is duplicated. If the results are not acceptable, then another critical component is selected for redundancy from the system and  $\mathcal{ETMA}$  is used for re-construct the new ET model.

In the next section, we apply our algorithm and tool, which can be downloaded from [14], on a real-world system for the domain of power protection and the results of  $\mathcal{ETMA}$ , in detail, are uploaded on the same above link.

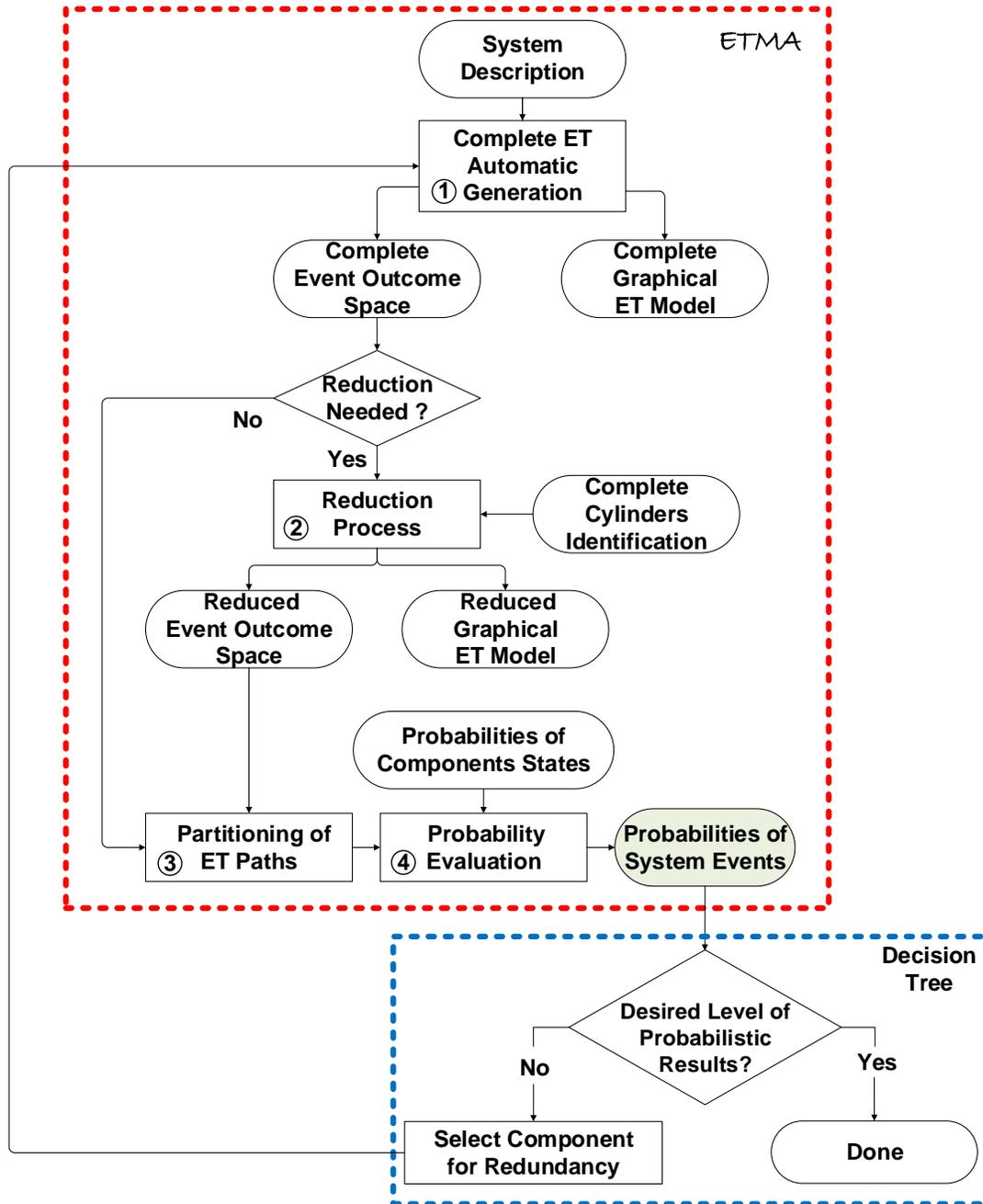


Figure 3: *ETMA* Analysis flowchart

---

**Algorithm 1** *ETMA*

---

```
1: procedure
2: S1: complete_gen
3:   Input: system name
4:     system components
5:     each system component states
6:   Output: complete ET model
7:     complete event outcome space
8: If Reduction of ET model needed?
9:   then
10:    S2: reduction_process
11:      Input: CCs identification
12:      Output: reduced ET model
13:        reduced event outcome space
14: S3: partitioning_paths
15:   Input: component event name(s)
16:     ET path number(s)
17:   Output: system events ET paths
18: S4: probability_eval
19:   Input: probabilities of components states
20:   Output: Occurrence probability of an event
21: end procedure
```

---

## 4 Trip Circuit Analysis

Consider a trip circuit in a power grid system, which is used to isolate a specific transmission line from the rest of a power grid, in case a fault occurs. The cascaded failure for many transmission lines could lead to a blackout situation for the whole grid, like what happened in San Diego in 2011 [15]. Therefore, a detailed ET analysis of the trip circuit is essential.

The power grid consists of one generator, 9 circuit breakers (CB), 4 bus bars (BB), 2 transmission lines (TL), 2 loads, 2 (on step up and one step down) transformers (Trans), 2 trip circuits (TC) with 1 relay (R) and 1 current transformer (CT), as shown in Fig. 4. During normal operation, all CBs are in a closed position. If a fault (F) occurs on  $TL_1$ , a primary current ( $I_p$ ) spike rises to about 20 times from a normal current level. Then, the CT detects that there is a fault in  $TL_1$  and the secondary current ( $I_s$ ) also rises with the same ratio simultaneously. Consequently, the relay coil increases the magnetic field and attracts the relay contacts, which are connected to the two separated trip circuits 1 and 2. Each trip circuit is provided with a battery. So, when the relay contact closes, it becomes a closed loop. Finally, the magnetic field produced by the trip coils 1 and 2 will push  $CB_1$  and  $CB_2$  to open and isolate  $TL_1$ . If all components of the trip circuit work correctly, then the fault becomes isolated and the grid is safe. If not, then the grid is in a risk situation of a blackout and back-up

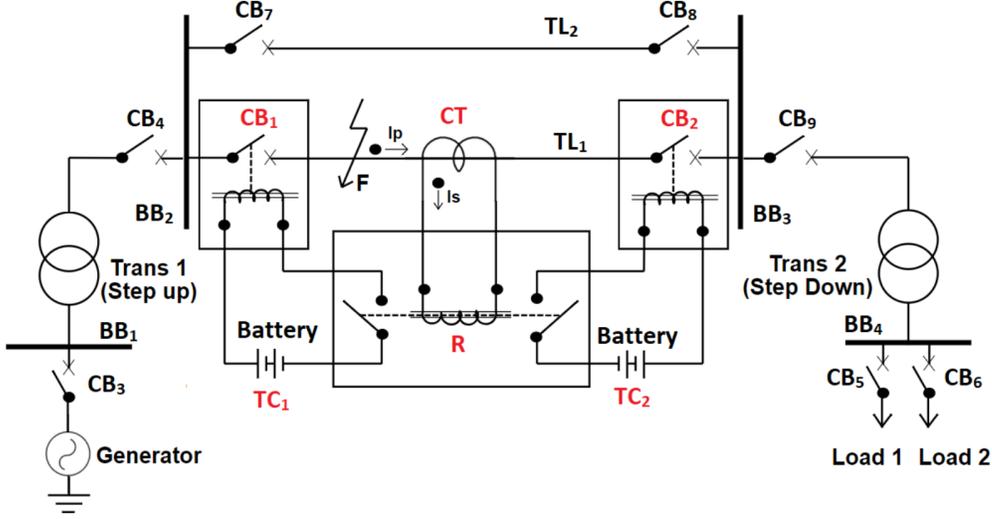


Figure 4: Single line diagram of a trip circuit in a power grid

decisions should be made. In this paper, we study the ET-based probabilistic analysis of all scenarios of failure and success that can occur in the trip circuit.

#### 4.1 Event Tree Analysis

We start the ET analysis of the trip circuit in  $\mathcal{ETMA}$  by first generating a complete ET model. Then, we delete the unnecessary nodes and branches to obtain a reduced ET that models the actual behavior of the trip circuit. Afterwards, we estimate the probabilities of different events that can occur in the trip circuit, for instance, the probability of both breakers  $CB_1$  and  $CB_2$  failing. Following are the steps required to conduct the trip circuit ET analysis in  $\mathcal{ETMA}$ :

*Step 1 (Complete ET Generation):*

We enter the details of the trip circuit components consisting of one  $CT$ , one  $R$ , two TCs ( $TC_1$  and  $TC_2$ ) and two CBs ( $CB_1$  and  $CB_2$ ) and each having two operational states, i.e., operating or failing, as shown in Fig. 5. However, we can assign different levels of failure associated with each component. The entered details are sufficient for  $\mathcal{ETMA}$ 's function to automatically generate the complete graph ET model, see Fig. 6 for a snapshot of a portion of the complete ET. This model shows the whole possible scenarios of failure and success for the trip circuit components states.  $\mathcal{ETMA}$  also automatically produces a complete event outcome space (64 paths from 0 to 63) from the complete ET model as:

$$\begin{aligned}
 Path_0 &= [CT_O, R_O, TC_{1O}, TC_{2O}, CB_{1O}, CB_{2O}] \\
 Path_1 &= [CT_O, R_O, TC_{1O}, TC_{2O}, CB_{1O}, CB_{2F}] \\
 &\vdots \\
 Path_{62} &= [CT_F, R_F, TC_{1F}, TC_{2F}, CB_{1F}, CB_{2O}] \\
 Path_{63} &= [CT_F, R_F, TC_{1F}, TC_{2F}, CB_{1F}, CB_{2F}]
 \end{aligned}$$

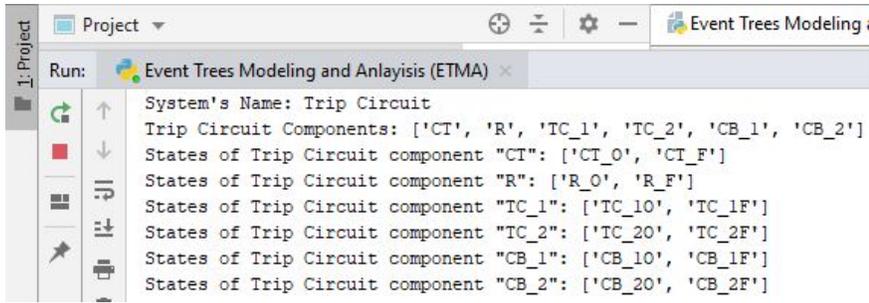


Figure 5: *ETMA*: Trip circuit identification O (Operates) / F (Fails to operate)

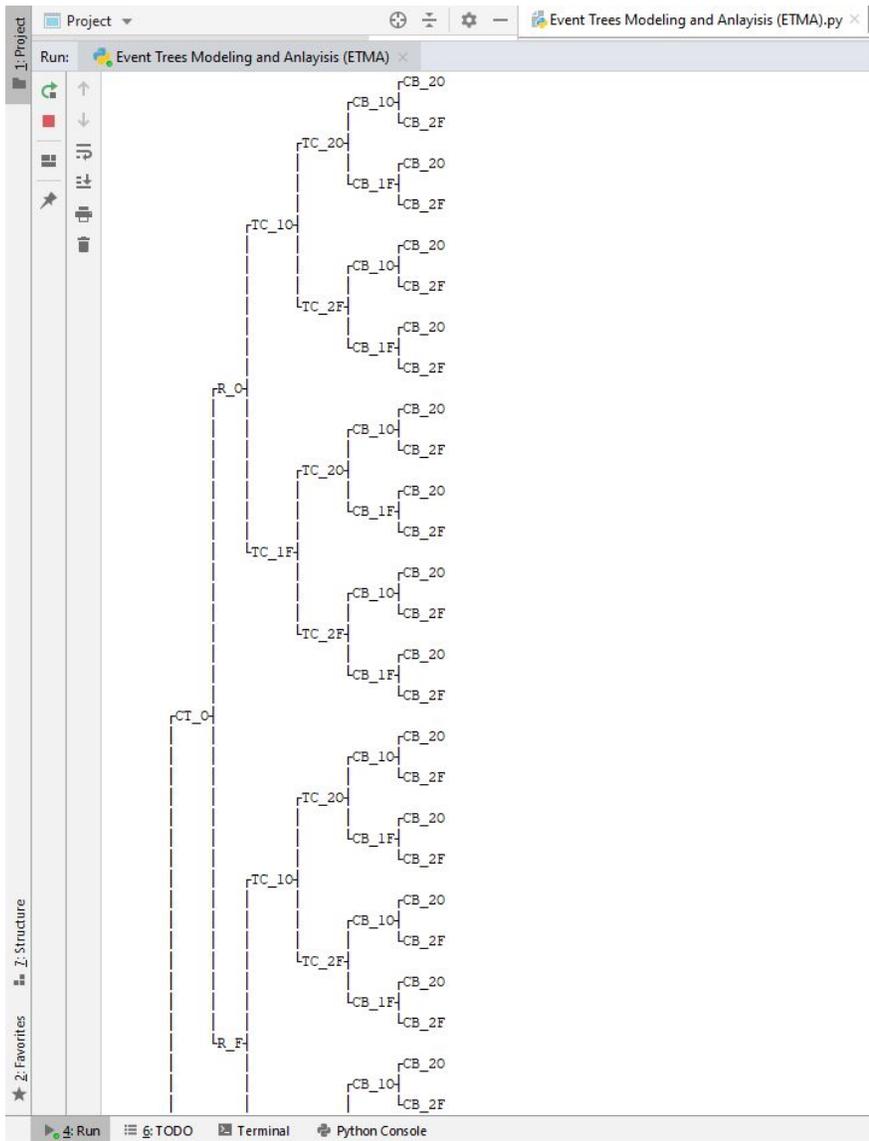


Figure 6: *ETMA*: Trip circuit complete ET model

Table 1: Trip circuit ET complete cylinders

CCs	ET Paths	CEs	Type of Deletion
$CC_1$	32, ..., 63	$CT_F$	Branch
$CC_2$	16, ..., 31	$CT_O, R_F$	Branch
$CC_3$	12, ..., 15	$CT_O, R_O, TC_{1F}, TC_{2F}$	Branch
$CC_4$	8, ..., 11	$CT_O, R_O, TC_{1F}, TC_{2O}, CB_2$	Node ( $CB_1$ )
$CC_5$	4, ..., 7	$CT_O, R_O, TC_{1O}, TC_{2F}, CB_1$	Branch

*Step 2 (ET Reduction Process):*

If the user desires to take into consideration the complete ET model generated in *Step 1*, then  $\mathcal{ETMA}$  provides a bypassing option for *Step 2* (i.e, ET reduction process). Otherwise, the next step is to define the CCs and their CEs (Table 1) to model the exact logical behavior of the trip circuit system. For instance, consider the paths from 32 to 63, if the CT fails then the likelihood or probability of occurrence of these paths are equal to the probability of CT failure only, regardless of the status of other components (i.e, the paths from 32 to 63 are CCs with respect to  $CT_F$ ). So, in  $\mathcal{ETMA}$ , we deleted the branches  $[CT_F, R_O]$  and  $[CT_F, R_F]$  from the complete ET (64 paths) in order to model the exact logical behavior of the trip circuit, as shown in Fig. 7. The reduced event outcome space (11 paths from 0 to 10) produced from the reduced ET model is as:

$$\begin{aligned}
 Path_0 &= [CT_O, R_O, TC_{1O}, TC_{2O}, CB_{1O}, CB_{2O}] \\
 Path_1 &= [CT_O, R_O, TC_{1O}, TC_{2O}, CB_{1O}, CB_{2F}] \\
 Path_2 &= [CT_O, R_O, TC_{1O}, TC_{2O}, CB_{1F}, CB_{2O}] \\
 Path_3 &= [CT_O, R_O, TC_{1O}, TC_{2O}, CB_{1F}, CB_{2F}] \\
 Path_4 &= [CT_O, R_O, TC_{1O}, TC_{2F}, CB_{1O}] \\
 Path_5 &= [CT_O, R_O, TC_{1O}, TC_{2F}, CB_{1F}] \\
 Path_6 &= [CT_O, R_O, TC_{1F}, TC_{2O}, CB_{2O}] \\
 Path_7 &= [CT_O, R_O, TC_{1F}, TC_{2O}, CB_{2F}] \\
 Path_8 &= [CT_O, R_O, TC_{1F}, TC_{2F}] \\
 Path_9 &= [CT_O, R_F] \\
 Path_{10} &= [CT_F]
 \end{aligned}$$

*Step 3 (Partition Outcome Space):*

The partitioning of the outcome space is essential as we are only interested in the occurrence of certain events in an ET. Suppose, we are only focusing on the failure of  $CB_1$ , then paths 2, 3, and 5-10 are obtained. Similarly, different sets of paths can be obtained by observing the behavior of the trip circuit components as:

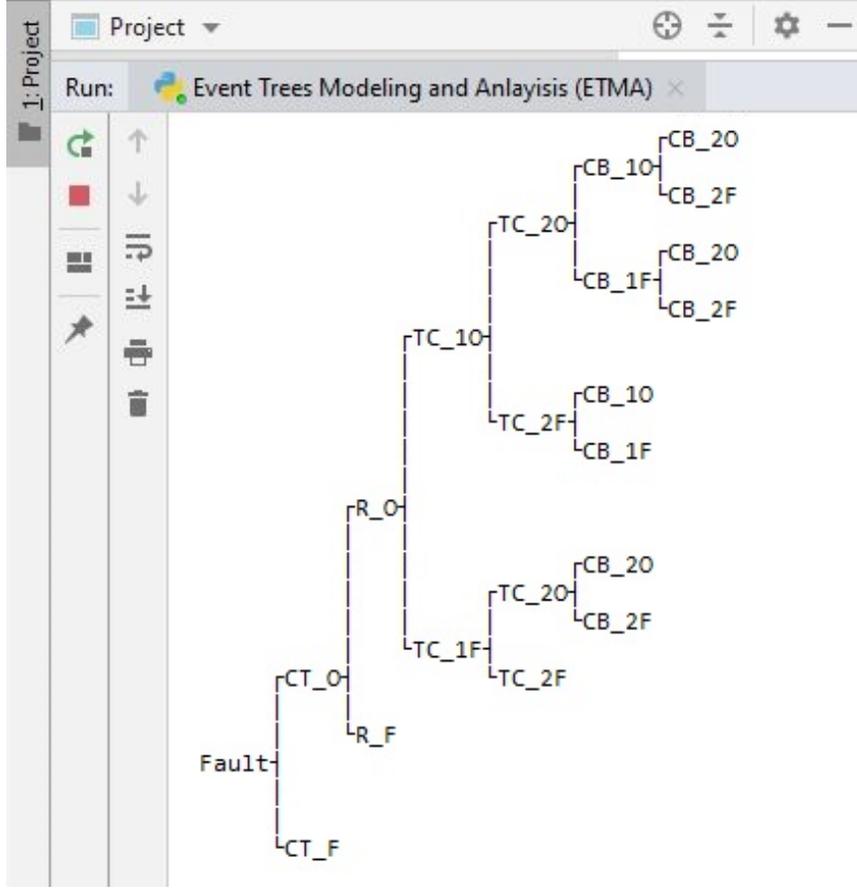


Figure 7:  $\mathcal{ETMA}$ : Trip circuit reduced ET model

- $\mathcal{P}(CB_1 \text{ Only Fails}) = \sum \mathcal{P}(2, 3, 5 - 10)$
- $\mathcal{P}(CB_1 \text{ Only Operates}) = \sum \mathcal{P}(0, 1, 4)$
- $\mathcal{P}(CB_2 \text{ Only Fails}) = \sum \mathcal{P}(1, 3 - 5, 7 - 10)$
- $\mathcal{P}(CB_2 \text{ Only Operates}) = \sum \mathcal{P}(0, 2, 6)$
- $\mathcal{P}(\text{Both } CB_1 \text{ and } CB_2 \text{ Fail}) = \sum \mathcal{P}(3, 5, 7 - 10)$
- $\mathcal{P}(\text{Both } CB_1 \text{ and } CB_2 \text{ Operate}) = \sum \mathcal{P}(0)$

To the best of our knowledge this feature is not found in any other existing ET analysis tool.

*Step 4 (Probability Evaluation):*

To estimate the probability of events associated with the trip circuit components, we assign probability values to each operational state of the components, as shown in Table 2. Assume that the times to failure of the trip circuit components are exponentially

Table 2: Trip circuit probability of components states

Component	$\lambda$ ( <i>f/yr</i> )	Prob. of Failure (%) After 6 Months	Prob. of Success (%) After 6 Months
CT	0.06	$CT_F$ (3%)	$CT_O$ (97%)
R	0.04	$R_F$ (2%)	$R_O$ (98%)
$TC_1$	0.08	$TC_{1F}$ (4%)	$TC_{1O}$ (96%)
$TC_2$	0.08	$TC_{2F}$ (4%)	$TC_{2O}$ (96%)
$CB_1$	0.06	$CB_{1F}$ (3%)	$CB_{1O}$ (97%)
$CB_2$	0.06	$CB_{2F}$ (3%)	$CB_{2O}$ (97%)

distribution with failure rate  $\lambda$  and time index  $t$ . Then the unreliability function or the probability of failure can be computed as [16]:

$$F(t) = \mathcal{P}(X \leq t) = 1 - e^{-\lambda t} \quad (4)$$

where  $X$  is a time-to-failure random variable. Similarly, the reliability of a component can be estimated by taking the complement of unreliability function with respect to the probability space as [16]:

$$R(t) = \mathcal{P}(X > t) = 1 - F(t) \quad (5)$$

The probabilities of the different trip circuit events, which are calculated using  $\mathcal{ETMA}$  are as follows:

- $\mathcal{P}$  (Both  $CB_1$  and  $CB_2$  Fail) = 5.389960806400000%
- $\mathcal{P}$  (Both  $CB_1$  and  $CB_2$  Operate) = 82.429704806399980%
- $\mathcal{P}$  ( $CB_1$  Only Fails) = 11.480127999999999%
- $\mathcal{P}$  ( $CB_1$  Only Operates) = 88.519871999999980%
- $\mathcal{P}$  ( $CB_2$  Only Fails) = 11.480127999999999%
- $\mathcal{P}$  ( $CB_2$  Only Operates) = 88.519871999999980%

It can be observed that the probability of both circuit breakers  $CB_1$  and  $CB_2$  failing is evaluated as 5.389960806400000%. If we want to decrease their probability to 2.5% or less, then we may add redundancy to these components. However, to ensure that the redundancy to these components are a correct decision, we need to conduct the decision analysis of the trip circuit, which is presented in the next section.

## 4.2 Decision Analysis

In the trip circuit, we can identify that the critical components are CT and R since the failure of these components may cause overall trip circuit failure. A decision-tree

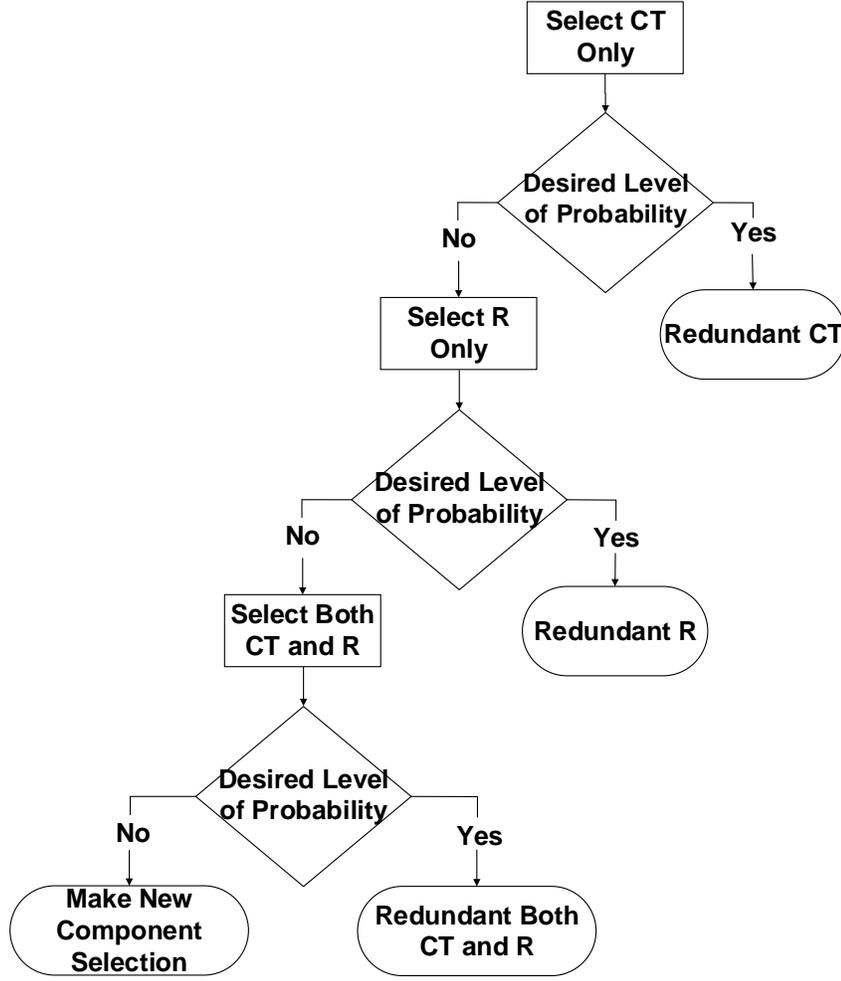


Figure 8: Decision tree for the trip circuit

describing the process of selecting the redundancy for critical trip circuit components is shown in Fig. 8. First, we select CT only for redundancy (i.e., adding  $CT_2$ ) assuming the same probability of failure and success of  $CT_1$ . If the probability of both circuit breakers  $CB_1$  and  $CB_2$  failing together, after re-evaluation in  $\mathcal{ETMA}$ , is equal to 2.5 % or less as required, then this is a correct decision. If not, then we select the critical component R for redundancy. If we still do not achieve the desired level of probability, then we select both CT and R together. If the results are not acceptable, then we make a new component selection from the trip circuit. We use  $\mathcal{ETMA}$  to generate the new reduced ET model after adding redundant  $CT_2$  and obtain the following event outcome space (31 paths only out of 128 complete paths):

$$\begin{aligned}
 Path_0 &= [CT_{1O}, CT_{2O}, R_O, TC_{1O}, TC_{2O}, CB_{1O}, CB_{2O}] \\
 Path_1 &= [CT_{1O}, CT_{2O}, R_O, TC_{1O}, TC_{2O}, CB_{1O}, CB_{2F}] \\
 Path_2 &= [CT_{1O}, CT_{2O}, R_O, TC_{1O}, TC_{2O}, CB_{1F}, CB_{2O}] \\
 Path_3 &= [CT_{1O}, CT_{2O}, R_O, TC_{1O}, TC_{2O}, CB_{1F}, CB_{2F}]
 \end{aligned}$$

$$\begin{aligned}
Path_4 &= [CT_{1O}, CT_{2O}, R_O, TC_{1O}, TC_{2F}, CB_{1O}] \\
Path_5 &= [CT_{1O}, CT_{2O}, R_O, TC_{1O}, TC_{2F}, CB_{1F}] \\
Path_6 &= [CT_{1O}, CT_{2O}, R_O, TC_{1F}, TC_{2O}, CB_{2O}] \\
Path_7 &= [CT_{1O}, CT_{2O}, R_O, TC_{1F}, TC_{2O}, CB_{2F}] \\
Path_8 &= [CT_{1O}, CT_{2O}, R_O, TC_{1F}, TC_{2F}] \\
Path_9 &= [CT_{1O}, CT_{2O}, R_F] \\
Path_{10} &= [CT_{1O}, CT_{2F}, R_O, TC_{1O}, TC_{2O}, CB_{1O}, CB_{2O}] \\
Path_{11} &= [CT_{1O}, CT_{2F}, R_O, TC_{1O}, TC_{2O}, CB_{1O}, CB_{2F}] \\
Path_{12} &= [CT_{1O}, CT_{2F}, R_O, TC_{1O}, TC_{2O}, CB_{1F}, CB_{2O}] \\
Path_{13} &= [CT_{1O}, CT_{2F}, R_O, TC_{1O}, TC_{2O}, CB_{1F}, CB_{2F}] \\
Path_{14} &= [CT_{1O}, CT_{2F}, R_O, TC_{1O}, TC_{2F}, CB_{1O}] \\
Path_{15} &= [CT_{1O}, CT_{2F}, R_O, TC_{1O}, TC_{2F}, CB_{1F}] \\
Path_{16} &= [CT_{1O}, CT_{2F}, R_O, TC_{1F}, TC_{2O}, CB_{2O}] \\
Path_{17} &= [CT_{1O}, CT_{2F}, R_O, TC_{1F}, TC_{2O}, CB_{2F}] \\
Path_{18} &= [CT_{1O}, CT_{2F}, R_O, TC_{1F}, TC_{2F}] \\
Path_{19} &= [CT_{1O}, CT_{2F}, R_F] \\
Path_{20} &= [CT_{1F}, CT_{2O}, R_O, TC_{1O}, TC_{2O}, CB_{1O}, CB_{2O}] \\
Path_{21} &= [CT_{1F}, CT_{2O}, R_O, TC_{1O}, TC_{2O}, CB_{1O}, CB_{2F}] \\
Path_{22} &= [CT_{1F}, CT_{2O}, R_O, TC_{1O}, TC_{2O}, CB_{1F}, CB_{2O}] \\
Path_{23} &= [CT_{1F}, CT_{2O}, R_O, TC_{1O}, TC_{2O}, CB_{1F}, CB_{2F}] \\
Path_{24} &= [CT_{1F}, CT_{2O}, R_O, TC_{1O}, TC_{2F}, CB_{1O}] \\
Path_{25} &= [CT_{1F}, CT_{2O}, R_O, TC_{1O}, TC_{2F}, CB_{1F}] \\
Path_{26} &= [CT_{1F}, CT_{2O}, R_O, TC_{1F}, TC_{2O}, CB_{2O}] \\
Path_{27} &= [CT_{1F}, CT_{2O}, R_O, TC_{1F}, TC_{2O}, CB_{2F}] \\
Path_{28} &= [CT_{1F}, CT_{2O}, R_O, TC_{1F}, TC_{2F}] \\
Path_{29} &= [CT_{1F}, CT_{2O}, R_F] \\
Path_{30} &= [CT_{1F}, CT_{2F}]
\end{aligned}$$

The new probabilities values evaluated using  $\mathcal{ETMA}$  describing the occurrence of failure and success in the trip circuit components are as follows:

$$\begin{aligned}
\mathcal{P}(\text{Both } CB_1 \text{ and } CB_2 \text{ Fail}) &= 2.255165963059199\% \\
\mathcal{P}(\text{Both } CB_1 \text{ and } CB_2 \text{ Operate}) &= 84.902595950591990\% \\
\mathcal{P}(CB_1 \text{ Only Fails}) &= 8.824531840000000\% \\
\mathcal{P}(CB_1 \text{ Only Operates}) &= 91.175468160000000\% \\
\mathcal{P}(CB_2 \text{ Only Fails}) &= 8.824531840000000\% \\
\mathcal{P}(CB_2 \text{ Only Operates}) &= 91.175468160000000\%
\end{aligned}$$

By comparing these values with those in Section 4 (*Step 4*), we can clearly observe that the trip circuit performance has been improved. Fig. 9 shows the comparison among these values in a histogram plot. It can be seen that the probability percentage of the circuit breakers  $CB_1$  and  $CB_2$  failing together is decreased from 5.38996% to 2.25517% by an amount of 3.13479%. Similarly, the proportion of the circuit breakers  $CB_1$  and  $CB_2$  succeeding together is also increased from 82.42970% to 84.90259% with an increment of 2.47289%.

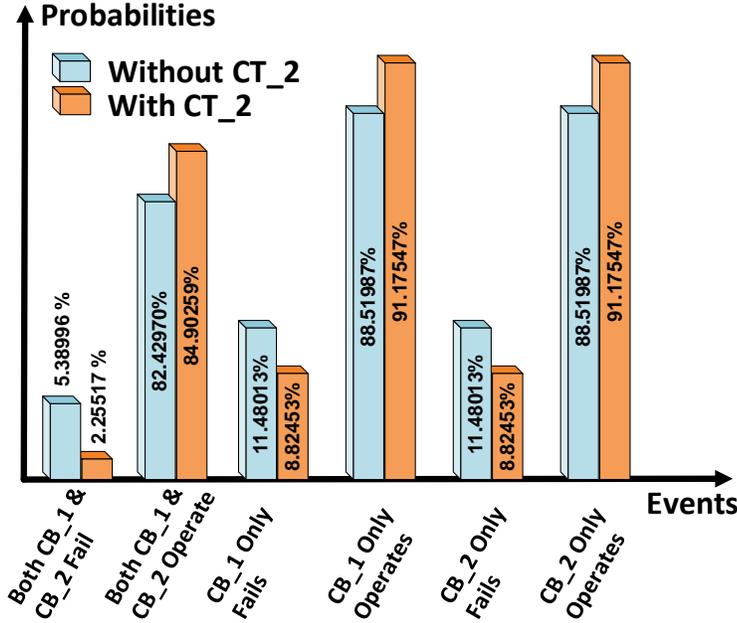


Figure 9: Trip circuit events probabilities evaluation

## 5 Comparison with Isograph

To ensure the accuracy of the  $\mathcal{ETMA}$  computation, we compare the trip circuit analysis results with the commercial Isograph ET analysis tool [6]. We analyze the trip circuit without any redundancy in the critical components using Isograph. It is important to mention that, unlike  $\mathcal{ETMA}$ , Isograph requires from the users to manually draw the trip circuit actual ET model ( $\mathcal{ETMA}$  Step 2) and assign the probability to each event as shown in Fig. 10. Since the partitioning process of the ET paths is not available in Isograph, we used the manual calculation of the paths probabilities that represent the occurrence of the the trip circuit events. The comparison in the probabilistic analysis of the trip circuit between  $\mathcal{ETMA}$  and Isograph is presented in Table 3.

It can be observed that the probabilities obtained from  $\mathcal{ETMA}$  are approximately equivalent to the corresponding ones calculated using Isograph. This clearly demonstrates that  $\mathcal{ETMA}$  is not only providing the correct results but also a complete ET analysis compared to existing ET analysis tools. Moreover, the CPU time for the trip circuit step-wise ET analysis in  $\mathcal{ETMA}$  is much faster than Isograph, as shown in Table 4. The experiments were performed on a single-core i5, 2.20 GHz, Linux VM with 1 GB of RAM device. Also,  $\mathcal{ETMA}$  is providing several additional features, including the automation of complete ET generation and the partitioning of ET paths for events probabilistic analysis, that are not available in any other existing reliability analysis tool. All these features of  $\mathcal{ETMA}$  are extremely useful for safety analysts and reliability engineers to quantify system improvements with fast and accurate decisions.

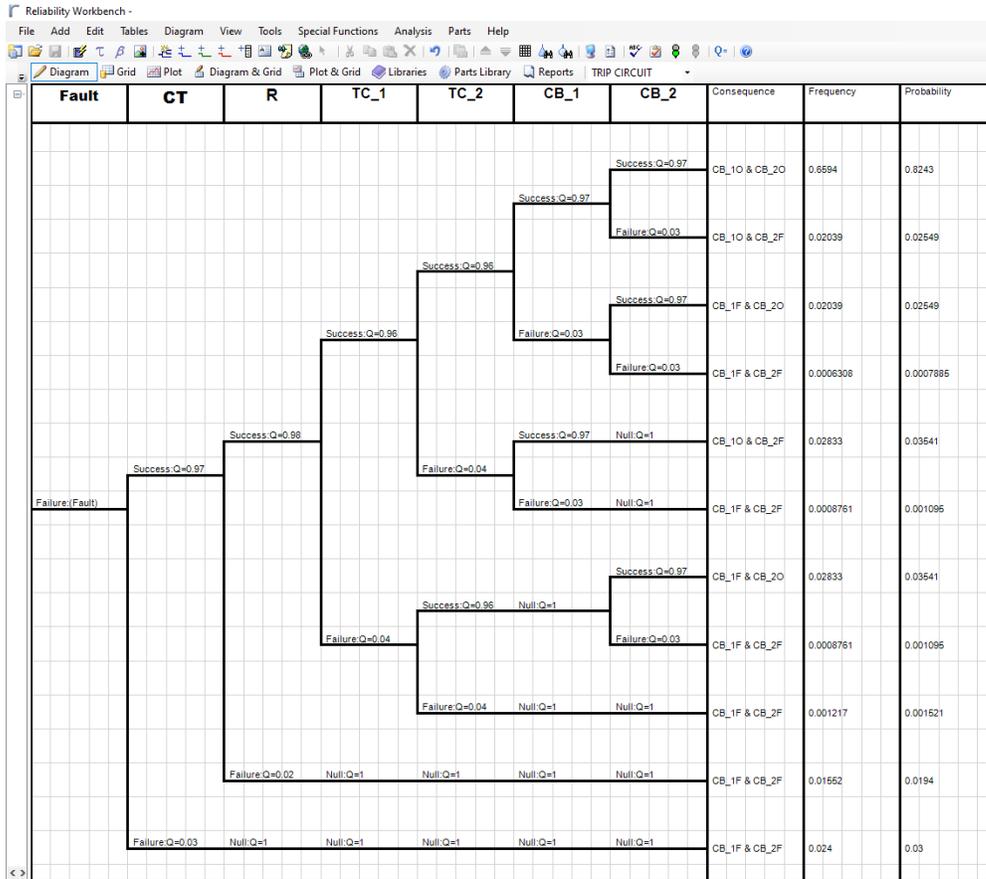


Figure 10: Isograph: Trip circuit ET model

Table 3: Comparison between  $\mathcal{ETMA}$  and Isograph

Trip Circuit Events	% Prob. from Isograph	% Prob. from $\mathcal{ETMA}$
Both $CB_1$ and $CB_2$ Fail	5.38996 %	5.389960806400000 %
Both $CB_1$ and $CB_2$ Operate	82.43 %	82.429704806399980 %
$CB_1$ Only Fails	11.48 %	11.480127999999999 %
$CB_1$ Only Operates	88.52 %	88.519871999999980 %
$CB_2$ Only Fails	11.48 %	11.480127999999999 %
$CB_2$ Only Operates	88.52 %	88.519871999999980 %

Table 4:  $\mathcal{ETMA}$ : Trip circuit CPU time

<i>Steps</i>	<b>CPU Time</b> $\mathcal{ETMA}$ ( <i>Seconds</i> )	<b>CPU Time</b> Isograph ( <i>Seconds</i> )	<i>Steps</i>	<b>CPU Time</b> $\mathcal{ETMA}$ ( <i>Seconds</i> )	<b>CPU Time</b> Isograph ( <i>Seconds</i> )
<i>Step 1</i>	0.291600	NA	<i>Step 3</i>	0.000631	NA
<i>Step 2</i>	0.000162	NA	<i>Step 4</i>	0.004319	2.752

## 6 Conclusions

In this paper, we proposed a new event trees modeling and analysis tool, called  $\mathcal{ETMA}$ , using *list* data-structure in Python.  $\mathcal{ETMA}$  provides several features to model any generic, complete and sequential ET diagram consisting of a large number of system components. Also,  $\mathcal{ETMA}$  provides deleting/reducing features to remove irrelevant specific nodes or paths from a complete ET diagram to model the exact logical behavior of the given system. Moreover,  $\mathcal{ETMA}$  provides partitioning of ETs paths and probabilistic analysis of the occurrence of a certain event. For illustration purposes, we conducted the ET modeling and analysis of the trip circuit in the power grid transmission lines. The results of  $\mathcal{ETMA}$  were used for making redundancy decisions about the critical components in the trip circuit. We also compared the results obtained in  $\mathcal{ETMA}$  with those from the Isograph tool, which is commonly used for ET analysis. We plan to extend our  $\mathcal{ETMA}$  tool with additional features for safety assessment [17], reliability analysis [18] and machine learning [19].

## References

- [1] R. E. Mackiewicz, “Overview of IEC 61850 and Benefits,” in *Power Syst. Conf. and Expo.*, Montreal, Canada, 2006, pp. 623–630.
- [2] R. Palin, D. Ward, I. Habli, and R. Rivett, “ISO 26262 Safety Cases: Compliance and Assurance,” in *IET Conf. on Syst. Safety*, Birmingham, UK, 2011, pp. 1–6.
- [3] R. Ferdous, F. Khan, R. Sadiq, P. Amyotte, and B. Veitch, “Handling Data Uncertainties in Event Tree Analysis,” *Process Safety and Environmental Protection*, vol. 87, no. 5, pp. 283–292, 2009.
- [4] Y. S. Hu and M. Modarres, “Evaluating System Behavior Through Dynamic Master Logic Diagram (DMLD) Modeling,” *Rel. Eng. & Syst. Safety*, vol. 64, no. 2, pp. 241–269, 1999.

- [5] ITEM Software, Available: <https://itemsoft.com/eventtree.html>, Accessed on: September 26, 2019.
- [6] Isograph Software, Available: <https://www.isograph.com>, Accessed on: September 26, 2019.
- [7] D. K. Sen, J. C. Banks, G. Maggio, and J. Railsback, “Rapid Development of an Event Tree Modeling Tool Using COTS Software,” in *Aerospace Conf.*, Big Sky, USA, 2006, pp. 1–8.
- [8] I. A. Papazoglou, “Mathematical Foundations of Event Trees,” *Rel. Eng. & Syst. Safety*, vol. 61, no. 3, pp. 169–183, 1998.
- [9] G. Van Rossum and F. L. Drake, *The Python Language Reference Manual*. Network Theory Ltd., 2011.
- [10] J. J. Grainger and W. D. Stevenson, *Power System Analysis*. McGraw-Hill, 2003.
- [11] I. A. Papazoglou, “Functional Block Diagrams and Automated Construction of Event Trees,” *Rel. Eng. & Syst. Safety*, vol. 61, no. 3, pp. 185–214, 1998.
- [12] PyGraphviz Python package, Available: <https://pygraphviz.github.io/>, Accessed on: September 26, 2019.
- [13] X. Niuniu and L. Yuxun, “Review of Decision Trees,” in *Int. Conf. on Comput. Sci. and Inform. Technology*, Chengdu, China, 2010, pp. 105–109.
- [14] *ETMA* Tool, Available: <https://github.com/hvg-concordia/ETMA>, 2019.
- [15] E. C. Portante, S. F. Folga, J. A. Kavicky, and L. T. Malone, “Simulation of The September 8, 2011, San Diego Blackout,” in *Winter Simulation Conf.*, Savannah, USA, 2014, pp. 1527–1538.
- [16] R. N. Allan, *Reliability Evaluation of Power Systems*. Springer Sci. & Bus. Media, 2013.
- [17] L. Qiang and L. Xiaoli, “An Approach to the Safety and Cost Control of Electrical Grid in Power Market,” in *Int. Conf. on Electricity Distribution*, Guangzhou, China, 2008, pp. 1–5.
- [18] P. J. Tavner, J. Xiang, and F. Spinato, “Reliability Analysis for Wind Turbines,” *Int. Journal for Progress and Applicat. in Wind Power Conversion Technology*, vol. 10, no. 1, pp. 1–18, 2007.
- [19] S. Raschka and V. Mirjalili, *Python machine learning*. Packt Publishing Ltd, 2017.