FFORT: A benchmark suite for fault tree analysis

Enno Ruijters[1], Carlos E. Budde[1], Muhammad Chenariyan Nakhaee[2], Mariëlle Stoelinga[1,3], Doina Bucur[2], Djoerd Hiemstra[2], and Stefano Schivo[4]

[1]*Formal Methods and Tools, University of Twente, The Netherlands.*
E-mail: {e.j.j.ruijters, c.e.budde, m.i.a.stoelinga}@utwente.nl
[2]*Data Science, University of Twente, The Netherlands. E-mail: {m.cnakhaee, d.bucur, d.hiemstra}@utwente.nl*
[3]*Software Science, Radboud University Nijmegen, The Netherlands.*
[4]*Faculty of Management, Science & Technology, Open University, The Netherlands.*
E-mail: stefano.schivo@ou.nl

This paper presents FFORT (the Fault tree FOResT): A large, diverse, extendable, and open benchmark suite consisting of fault tree models, together with relevant metadata.

Fault trees are a common formalism in reliability engineering, and the FFORT benchmark brings together a large and representative suite of fault tree models. The benchmark provides each fault tree model in standard Galileo format, together with references to its origin, and a textual and/or graphical description of the tree. This includes quantitative information such as failure rates, and the results of quantitative analyses of standard reliability metrics, such as the system reliability, availability and mean time to failure. Thus, the FFORT benchmark provides: (1) Examples of how fault trees are used in various domains; (2) A large class of tree models to evaluate fault tree methods and tools; (3) Results of analyses to compare newly developed methods with the benchmark results.

Currently, the benchmark suite contains 202 fault tree models of great diversity in terms of size, type, and application domain. The benchmark offers statistics on several relevant model features, indicating e.g. how often such features occur in the benchmark, as well as search facilities for fault tree models with the desired features. In addition to the trees already collected, the website provides a user-friendly submission page, allowing the general public to contribute with more fault trees and/or analysis results with new methods. Thereby, we aim to provide an open-access, representative collection of fault trees at the state of the art in modeling and analysis.

Keywords: Fault Tree Analysis, Reliability Engineering, Case Studies, Quantitative Risk Analysis

1. Introduction

Fault trees (FTs) are a widely-used formalism for safety and reliability analysis (Ericson (1999); Stamatelatos et al. (2002); Ruijters and Stoelinga (2015)). An FT is a graphical representation of the possible *failure modes* of a system, i.e. the distinct processes by which determined system functionality failures can be observed (Rausand and Høyland (2004)), broken down into intermediate failures and their interactions. From quantitative information about elementary failure behavior (like component failure rates), fault tree analysis provides quantitative results on metrics such as time-dependent system failure probability and average system downtime.

Fault trees are popular as a clear graphical formalism to analyze RAMS—reliability, availability, maintainability, and safety—metrics of complex systems. Many extensions and analysis methods and tools have been developed from the original FT concept (Ruijters and Stoelinga (2015)). However, a systematic way of comparing all these methods is lacking: Many published papers use their own examples and case studies to evaluate the merits of their new techniques. From a methodological point of view this practice has significant disadvantages: (1) It is possible to present case studies that are biased in favor of the newly introduced methods and tools; (2) When papers use their own examples, an objective comparison of different methods becomes difficult; (3) Due to lack of sources, the number of examples and case studies used in publications is often relatively small.

This paper makes an important step towards a more systematic comparison in fault tree analysis research, namely by providing a large, open*, quantitative, searchable, and extensible benchmark suite for fault tree models. In this sense, a major feature of FFORT is the diversity of its content, where FTs have different size (number of basic events and gates), type (static vs. dynamic,

Proceedings of the 29th European Safety and Reliability Conference.
Edited by Michael Beer and Enrico Zio
 Published by Research Publishing, Singapore
ISBN: 981-973-0000-00-0 :: doi: 10.3850/981-973-0000-00-0_main

repairable or not), failure behavior (diverse gates and probability distributions in basic events), and RAMS metrics computed. Table 1 offers an overview of this.

Furthermore, reproducibility has gained renewed interest in computer science, particularly in research on formal methods (Schlick et al. (2018)). According to the ACM (2018), such reproducibility should be tested by a different team using a different experimental setup from the original research. So far, there has been no systematic effort to reproduce published results in fault tree analysis. Thus and in addition to the contributions mentioned above, we reproduce parts of previously published papers by analyzing their fault trees in a systematic way, thereby validating the published results.

To these aims, the benchmark suite provides for each FT it contains:

- A complete textual representation in the standard Galileo format, following the syntactic guidelines detailed in Sect. 4.
- A summarized description and pictorial illustration (taken from the authors when available) to facilitate understanding.
- Quantitative information such as rates and failure probabilities of basic events.
- Values of RAMS metrics, namely previously-published values if available, plus newly computed values for reference purposes.

The FTs—and their metadata—that FFORT thus provides come from an unbiased collection of case studies gathered from scientific literature, ranging from the classic NASA fault tree handbook (Stamatelatos et al. (2002)) to modern papers on advanced analysis techniques. These include several industrial cases modelling software or physical assets of companies, such as PCBAs, vehicle guidance, railways, ship mooring, and tank storage. Search and filter facilities are provided to select FTs with specific characteristics; an essential feature in the benchmark as per the great diversity of FTs offered.

Thus, FFORT makes significant steps towards:

(1) A systematic and comprehensive way of *comparing the capabilities and performance* of FT analysis tools.
(2) *Validation* of new analysis techniques by comparing outcomes to reference results already published.
(3) Confirmation of the *reproducibility* of published results. Indeed, we have already identified one unreproducible result.
(4) *History-tracking of case studies*, as variations of FTs are created in different publications.
(5) Examination of how often *more complex modeling features* are used, e.g. dynamic gates and maintenance representation.

In addition to the trees already collected, we provide a user-friendly submission page, allowing modelers to upload more examples of FTs or analysis results with new methods. Thereby, we aim to continue to provide an open-access, representative collection of fault trees at the state of the art in modeling and analysis.

The structure of this paper is as follows: Sect. 2 provides a brief overview of fault trees. Sect. 3 describes the data and metadata that is stored in FFORT. Sect. 4 explains the methodology used to collect the FTs currently in FFORT, while Sect. 5 provides some statistics about these FTs. Sect. 6 shows the user interface to access FFORT, before ending with a conclusion and discussion in Sect. 7.

2. Fault Trees

Fault tree analysis is an industry-standard, widely used formalism to graphically model systems and analyse them for reliability and safety (IEC61025 (2006)). Entities like NASA, NRC, ProRail, Boeing, etc. use fault tree analysis to ensure compliance to both national and international safety regulations. FTs model the interactions of component failures that may lead to (sub-) system failures, thereby supporting the analysis of a wide range of qualitative and quantitative dependability analyses.

A fault tree is a directed acyclic graph, in which the leaves are called *basic events* and the remaining nodes are called *gates*. Basic events specify elementary failure causes (e.g., failures of individual components, external causes), while the gates specify how these failures combine to cause system level failure. The root of the FT is called the *top level event* and denotes system failure.

Fig. 1 shows an example of a fault tree. The top level event is called "System." Its symbol ( ) denotes an OR-gate, representing that a failure of either child causes a system failure. Its children are the basic event "HCR" and the AND-Gate "Batteries." The latter denotes that the battery subsystem requires both batteries B_1 and B_2 to fail for the subsystem (and thus the entire system) to fail.

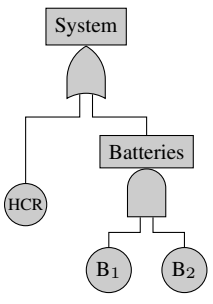


Figure 1. Simple fault tree

Standard (or *static*) FTs support only boolean gates (i.e., AND, OR, and k-out-of-N). Various extensions have been developed supporting more complex combinations. The most prominent is the *dynamic fault tree* (DFT) by Dugan et al. (1990) which adds PAND (Priority-AND) gates imposing temporal requirements, SPARE gates used to denote spare parts, FDEP (function dependency) gates denoting subtrees that cause other subtrees to fail, and SEQ (sequence-enforcer) gates that denote that certain failures can only occur in a particular order. A more recent extension by Ruijters et al. (2016) is the *fault maintenance tree*, which adds *inspection* and *repair modules* to specify complex maintenance and repair policies.

Fault trees can be analyzed to obtain various metrics relevant to reliability engineering. Qualitative analysis provides information such as *cut sets*: Sets of component that, if failed, cause the system to fail. If basic events are decorated with failures probabilities or rates, quantitative metrics can be calculated. Which properties are applicable depends on the provided information: Given failure probabilities, the system *reliability* (probability of no system failure) can be computed. If failure rates over time are available, one can compute *timed reliability* (probability of no system failure occurring before a given mission time) or *mean time to failure* (MTTF). When repair information is available as well, the system *availability* (average proportion of time that the system is not failed when operating under standard conditions) can also be computed.

3. The FFORT Benchmark Suite

In addition to storing the trees themselves, FFORT also stores metadata to explain each FT and its origins. Since we collect FTs from published literature, FFORT does not need to store the full details of each tree, rather referring to the original publication for full details. Nonetheless, we try to provide sufficient context for each tree to be reasonably understandable without reading the entire paper.

FFORT stores all FTs as *variants* of a particular *family*, with each variant possibly having some associated set of *results*. We have identified two patterns to the variants: (1) Different FTs modeling the same system, e.g. to suit different analysis tools; (2) FTs modeling similar systems, e.g. to explore the effects of adding certain redundancies. Many families contain only a single variant, as that is the only published FT.

The FTs themselves (i.e. the *data*) are stored in the Galileo format (as described by Sullivan and Dugan (1998); Sullivan et al. (1999)), while the associated *metadata* is stored in JSON. We store the following metadata—required fields indicated in bold—for each family:

- **Name** and **short description** of the FT.
- **Reference** to the publication first describing the FT (title, author names, publication year, and DOI or website link if available online).
- **Name** and **e-mail address** of the person who submitted the FT to FFORT.
- **Date** in which the FT was added to FFORT.
- Additional *references* providing further information.

In turn, for each variant in a family we store the following metadata:

- **Name** and **description** if the family has multiple variants.
- *Image* of the FT, when practical due to size, etc.
- **Reference** introducing the variant, if different from the family reference.
- *Results* if available, each one including:
  - **Metric** calculated, e.g. reliability for some time horizon t, availability, etc.
  - *Reference* to the publication containing the result (omitted only for reference results computed for FFORT).
  - **Value** of the calculated result.
  - **Tool name** used to calculate the result, or "manual computation" if the result comes from analytical calculations.

In addition, we automatically extract quantitative information about the fault trees data, for statistical and search/filtering purposes. In particular, we calculate for each FT:

- The number of BEs and their attributes.
- The number of gates of each type.
- Whether the FT supports any type of repairs.

Furthermore, we provide reference results of standard RAMS metrics for each FT in the collection. We apply two tools to every FT for this: *Storm-DFT* by Volk et al. (2018) and *DFTCalc* by Arnold et al. (2013), the latter with its 'exact' backend if possible, otherwise with its IMCA backend. For non-repairable FTs we calculate mean time to failure and reliability; for repairable FTs we calculate availability in addition. In the case of reliability, if a particular time point was used in previous results, we calculate reliability for that time as well; otherwise we compute reliability for time $t = 1$. Some FTs could not be analyzed using one or both of these tools due to restrictions on the supported FT features—e.g., Storm-DFT cannot process repairable FTs—or computational resource limits.

Fig. 2 shows part of the front page of the FFORT website with the 'Cardiac Assist System' and the results of one of its variants open. The first line of the table shows the combined metadata about the variants (number of BEs, types of gates etc.). Below this is the overall description and reference, where an image is also displayed if one

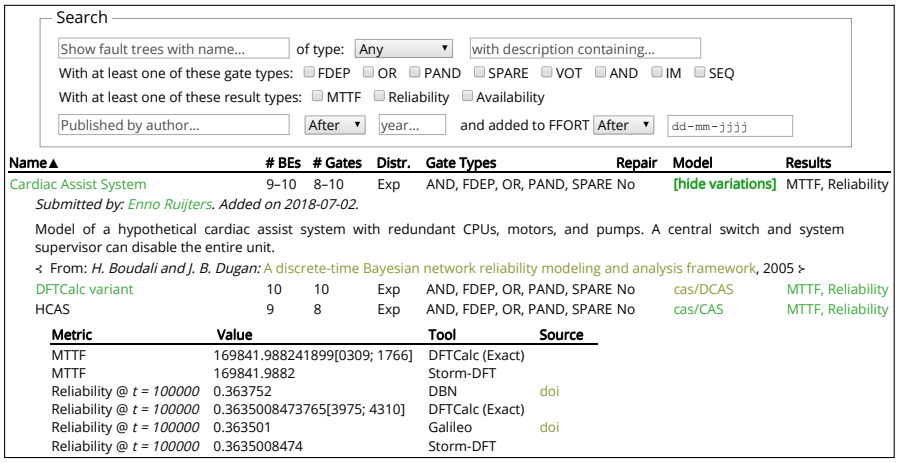


Figure 2. Excerpt from the web interface of FFORT

is available. The lines for the variants show metadata about the individual FTs as well as the link to the model file. The results show that this variant has two published results (with DOIs linked in the 'Source' column) and four unpublished reference results which closely match the published values.

4. Collection Methodology

To populate FFORT, we have applied specific criteria in our literary survey, to ensure a consistent and homogeneous representation of the data in line with the required features described in Sect. 3.

4.1. *Validation rules*

When considering fault tree $\mathcal{FT}$ (family or variant) for inclusion in the benchmark, the following conditions were checked:

(1) $\mathcal{FT}$ must have been introduced in a *referenceable publication*, namely a scientific article published in a journal or conference or workshop proceedings, or a published book.
(2) *Nontrivial size*; specifically $\mathcal{FT}$ must have at east 10 nodes, unless it is part of a family where there are other trees that satisfy this condition.
(3) *Structural unambiguity*, i.e. there must be either a complete graphical representation or a clear description (or a combination of these) describing $\mathcal{FT}$ entirely.
(4) *Analyzability by public software*, i.e. there must exist some publicly available tool that can compute the metrics for $\mathcal{FT}$ that appear in the publication it was taken from[†]. Notice this does not necessarily rule out trees appearing in publications where results were computed analytically ("by hand"), as long as the corresponding Galileo encoding of $\mathcal{FT}$ can theoretically be analyzed by existing tools.
(5) $\mathcal{FT}$ must contain *quantitative information* that admits the computation of some standard RAMS metric, e.g. availability, reliability, (untimed) failure probability, etc.

Item 5 rules out fault trees which, due to the available published information, are susceptible to qualitative analyses alone, for instance those studied by Zhang et al. (2018). This is motivated by the focus of FFORT on *quantitative fault tree analysis*, oriented to the development and improvement of software tools that target this goal.

4.2. *Naming and structural conventions*

The data content of the fault trees in FFORT is stored as plain text files, written in the standard Galileo format[‡]. The file extension is **`dft`** and, to facilitate parsing by software tools, we use the following conventions for the nodes of the tree:

- Names, be these of basic events or gates, are enclosed in double quotes "like this."
- The top level event (i.e. the root node of the tree) is named "`System`".
- The names of all other nodes follow the publication from which the tree was extracted:
  - if the node name in the publication is an abbreviation (e.g. `HCR_2`), the string is used verbatim enclosed in double quotes as per the first item (e.g. "`HCR_2`");
  - if a long name is used instead, possibly including spaces (e.g. `higher cabin`

[†] We target theoretical analyzability, disregarding practical hardships like tree size or computation time.

[‡] Described at `https://dftbenchmarks.utwente.nl/galileo.html`

relay 2), spaces are stripped and the string is written in camel case (e.g. "higherCabinRelay2").

- After the tree root on the first line, each node of the tree appears as a single line in the file, describing it according to the Galileo format.
- The order of declaration of the nodes in the file follows a preorder (i.e. root-first order) of the tree hierarchy; that is, if gates $\mathtt{G}_1, \mathtt{G}_2, \ldots, \mathtt{G}_N$ are children of gate G, then the line declaring G in the file appears before the lines declaring $\{\mathtt{G}_i\}_{i=1}^{N}$.
- Basic events, i.e. the fault tree leaves, are declared in the file after all gates, and in its left-to-right order of definition; e.g. for the (sub-) tree PAND(BE1,BE2) the line declaring the PAND gate appears first in the file, then the line declaring the basic event BE1 is declared on a lower (not necessarily consecutive) line, and immediately below it is the line declaring the basic event BE2.

To provide a full concrete example, the fault tree in Fig. 1 is translated into the content of the tree.dft file shown in Fig. 3, where the failure rates of the basic events (i.e. the values assigned to the lambda constants) are assumed given in the text of the corresponding publication.

```
toplevel "System";
"System" or "HCR" "batteries";
"batteries" and "B1" "B2";
"HCR" lambda=2.8e-5;
"B1" lambda=1.13e-6;
"B2" lambda=1.13e-6;
```

Figure 3. Galileo description of the FT in Fig. 1

5. Statistics

FFORT is a diverse benchmark suite with fault trees that differ in *size* (i.e. the number of nodes in the tree), *type* (static vs. dynamic, repairable or not, with maintenance support or not), *failure behaviour* (diverse failure probability distributions for the basic events, and several gate types), and *metrics computed* (untimed failure probability, reliability for certain time horizon, availability, mean time to failure). Table 1 offers an overview of this diversity.

At the time of publication, FFORT contains 202 FTs from a total of 24 families. There is considerable variation in the number of FTs per family, with the largest family containing 68 FTs, and many 'families' having only one FT. The families containing many variants are mainly those that were used as benchmarks for analysis tools (where variants of different sizes demonstrate scalability of the tool). Some of the main properties of the FTs currently in FFORT are shown in Table 1

Table 1. Properties of some FTs currently in FFORT: Distribution is either 'D' for discrete probabilities or 'E' for failure rates describing exponential distributions; Results are 'P' for published, 'R' for reference, 'PR' for both, 'X' for neither, or blank for not applicable to the model type.

Family acronym[§]	# Variants	Dynamic	Maintenance	Distribution	Reliability	Availability	MTTF
FTPP	16	✓	✓	E	P	R	R
HECS	28	✓	✓	E	PR	P	R
RCabin	9		✓	E	R	PR	R
TFS	1		✓	E	X	X	X
CAS	2	✓		E	PR		R
RCross	68	✓		E	R		R
CSDE	1			D	R		
SMS	12			D	PR		
ST	3			D	P		
WDQ	1			D	X		

5.1. *FT elements*

We observe a balance between discrete-time and continuous-time FTs, with 16 families specifying basic events' failure rates, and the remaining 8 specifying simple failure probabilities. We did not encounter any FTs that mixed these types.

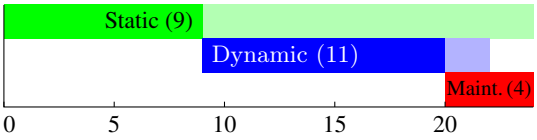
In terms of size, the FTs vary from 6 to 253 basic events (median 22), and 4 to 161 gates (median 14). We note that the largest FTs are contained in families that also have more modest sizes, with the smallest FTs per family ranging from 6 to 54 BEs (median 10) and 4 to 50 gates (median 9).

With respect to the FT types, we have a mix of *static*, *dynamic*, and *maintenance* FTs, as shown in Fig. 4. For the purpose of this classification, we consider each FT is a member of the most restrictive class it fits in. So for instance, a tree containing only static gates is considered a static fault tree, even though it also meets the formal definition of a dynamic fault tree.

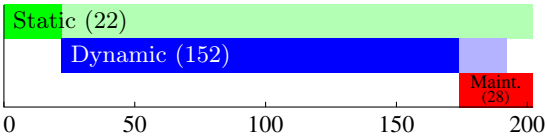
The gate types used are shown in Fig. 5. As one would expect, the AND- and OR-gates are by far the most common, with relatively even numbers for the more complex types. The sequence-enforcer gate and inspection module are the least used, which is understandable given their poor support by many analysis tools.

The average number of each type of gate per FT can be found in Fig. 6. The OR-gate is by far the most common, which was expected as this

[§]Capitals of the tree's **Name**, see Fig. 2



(a) Per family: 4 maintenance, 11 dynamic (+2 also maintenance gates), 9 static (+11 also dynamic, +4 also maintenance).



(b) Per variant: 28 maintenance, 152 dynamic (+18 also maintenance gates), 22 static (+152 also dynamic, +28 also maint.).

Figure 4. Trees containing gates of type: static (green, top), dynamic (blue, middle), and maintenance (red, bottom). Lighter color indicate families that also contain gates of more extended types.

is what usually connects different failure modes (e.g. a general system failure may take place if functionality A *or* B are lost, which can be caused by failure modes F_A *or* F_B respectively). Next are the AND-, SPARE-, and PAND-gates, often connecting failure modes related via redundancy. The FDEP, voting (VOT), and sequence enforcer (SEQ) gates occurs more rarely, as many systems do not include these features. Inspection modules (IM) occur very infrequently, as most FTs that include maintenance specify only one policy represented by an IM.

5.2. *Quantitative results*

Of the 24 families in FFORT, 16 include at least one quantitative result. As described in Sect. 3, we provide *published* results that are taken from scientific literature (generally the same paper that describes the FT itself) and *reference* results computed ourselves for the sake of the benchmark. Of the 16 FTs with results, 11 contain published results, while the remaining five have only reference results.

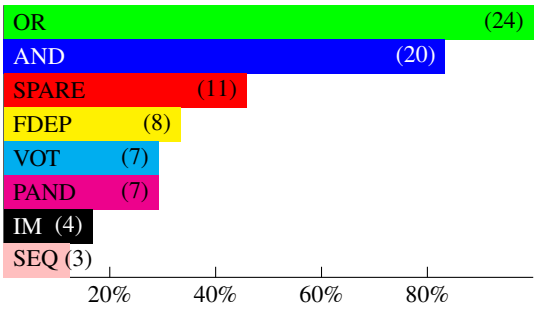


Figure 5. Number of families containing the gate types

Of three FTs that do not have results, one contains a feature not supported by any of the analysis tools available to us (the *Restoration factor*), and the other two are too large for the tools to analyze them in two hours on our computers[¶].

Fig. 7 shows the percentage of FT (families) for which a particular type of result is available. An interesting remark is that the published results contain only metrics on reliability and availability. The mean time to failure is included as a reference result for many FTs, but is apparently not published in general. We did not compute reference results for availability, as the only tool available to us to compute the availability for repairable FTs (DFTCalc) is the same tool that generated the published results.

In computing reference results, we have already identified one published result (by Arnold et al. (2013)) that did not match the reference result. In collaboration with the original authors, we identified that the published value was erroneous, caused by a typographical error in the program invocation.

6. User Interface

The FFORT website[‖] consists of three web pages: The *main page* where (filtered subsets of) FTs can be viewed and downloaded, the *statistics page* where various statistics of the collection can be found in real-time, and the *submission page* where new FTs can be submitted.

In addition to the website, a git repository[††] can be downloaded containing all the models with their JSON metadata (as well as the source code of the website). This feature allows e.g. tool authors to automatically execute their tool on all available FTs.

6.1. *Main page*

Fig. 2 shows part of the front page of FFORT. At the bottom of the image, the list of FTs can be seen. Information about each model can be seen by clicking its name, which expands the entry to shows its description, reference, and image if available (an example description is shown at the top of Fig. 8). For FT families with a single variant, the Model column contains links to the Galileo files. For families with multiple variants, the clickable text [show variations] expands a list

[¶]Our computers run Linux kernel 4.4.0-138, on Intel® Xeon® E5520 CPUs with 24 GB of DDR3 RAM @1066MHz.

[‖]The website can be found at `https://dftbenchmarks.utwente.nl/`.

[††]The repository can be found at `https://dftbenchmarks.utwente.nl/ffort.git`.

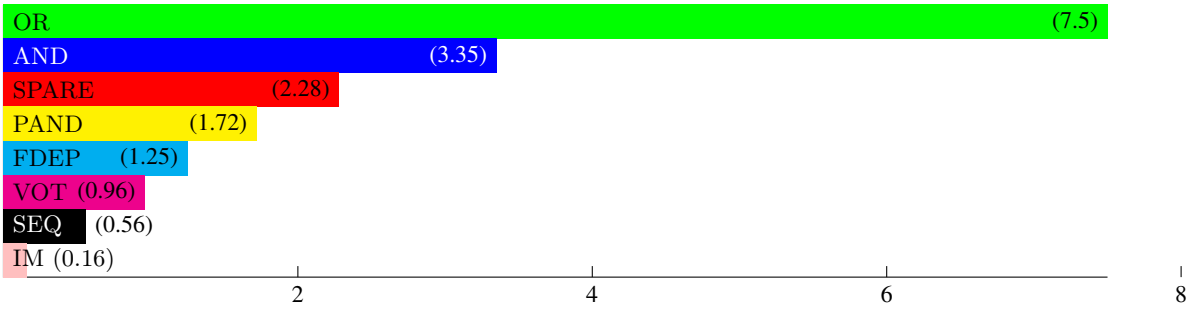


Figure 6. Number of each element type per FT (averaged per family).

showing the variants with links to their Galileo files.

At the top of the page is the search box. Here, the user can search for FTs by name, description, or publication author or year. In addition, the user can choose to show only FTs of a particular type (static, dynamic, or maintenance), containing particular gates, and/or for which a particular type of result is available. Furthermore, the FTs can be restricted by the date they were added to FFORT. In this way, a tool author can, for example, try their tools on all FTs with certain properties, and later easily check whether any new FTs were added with those properties.

6.2. *Statistics page*

The statistics page of FFORT shows various statistics similar to those reported in Sect. 5, computed in real-time over the models present in FFORT. In addition, the statistics can be calculated over subsets of the submitted trees using all the filtering capabilities of the main page as described in the previous section.

6.3. *Submission page*

Fig. 8 shows a partly filled out submission page. At the top of the page, the user can see how the submitted information would appear on the main page (except for the information that is automatically calculated from the tree).

The submission page allows multiple variants to be submitted by adding additional models, and one or more results can be submitted for each variant. New variants can be submitted by naming the model identical to an already-included one, and new results can be submitted by using an identical name and omitting the model file.

For expert users, the JSON-formatted metadata can be edited manually to perform tasks that would otherwise be cumbersome (e.g., adding large numbers of variants by copying and pasting one variant and making minor adjustments) or unsupported (e.g., specifying a different submission date for re-submissions with corrected models).

Completed forms can be submitted automatically to the maintainers of FFORT. After submission, the maintainers verify that the submitted tree and metadata are valid (i.e., that the submitted Galileo file has the correct syntax, DOIs refer to the correct papers, etc.), generate the model-derived metadata (i.e., number of gates, etc.), and add the submission to the main page.

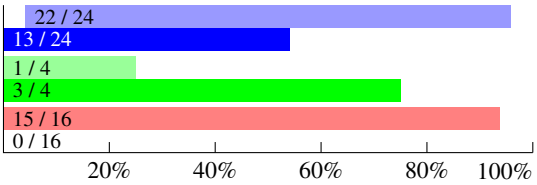


Figure 7. Percentage of families with given results types (out of families for which the type is applicable). Reference results in lighter color at the top of each pair, published results below it. Reliability at the top (blue), availability in the middle (green), and MTTF at the bottom (red)

7. Conclusion

This paper has presented FFORT, a compilation of diverse fault trees for benchmark purposes. We have collected FTs from the scientific literature, described them in a uniform input language, and we make them publicly available together with metadata about the FTs. We provide the metadata both on a user-friendly website and in machine-readable form. We further hope to expand the FFORT both by collecting further FTs ourselves and by soliciting contributions from other researchers on fault tree analysis.

Discussion One of the goals of FFORT is to provide validation for analysis techniques. Already during the construction of FFORT, the calculation of reference results identified software bugs in both tools used for the task (DFTCalc and Storm-DFT). Furthermore, we identified a case where

Figure 8. Excerpt from the submission page of FFORT

published results do not match our reference result, which was traced to a typographical error in the tool invocation used for the published result. We hope that by collecting published results, FFORT will help authors and developers to improve their own software tools, and promote reproducibility of published results.

Acknowledgments

Thanks to Matthias Volk for submitting several FTs to FFORT. This work was partially funded by STW project SEQUOIA (15474), KIA KIEM and BetterBe B.V. grant StepUp (628.010.006), and EU project SUCCESS (102112).

References

ACM (2018). Artifact review and badging. `https://www.acm.org/publications/policies/artifact-review-badging`.

Arnold, F., A. Belinfante, F. van der Berg, D. Guck, and M. Stoelinga (2013). DFTCalc: A tool for efficient fault tree analysis. In *Proc. 32nd Int. Symp. on Comput. Safety, Reliability, and Security (SAFECOMP)*, Volume 8153 of *LNCS*, pp. 293–301.

Dugan, J. B., S. J. Bavuso, and M. A. Boyd (1990). Fault trees and sequence dependencies. In *Proc. IEEE Annu. Rel. and Maintainability Symp.*, pp. 286–293.

Ericson, C. (1999). Fault Tree Analysis – a history. *Proc. 17th Int. System Safety Conf.*, 87–96.

IEC61025 (2006). IEC 61025: Fault tree analysis.

Rausand, M. and A. Høyland (2004). *System reliability theory: models, statistical methods, and applications* (2 ed.). John Wiley & Sons.

Ruijters, E., D. Guck, P. Drolenga, and M. Stoelinga (2016, January). Fault maintenance trees: reliability contered maintenance via statistical model checking. In *Proc. IEEE Rel. and Maintainability Symp. (RAMS)*.

Ruijters, E. and M. Stoelinga (2015). Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools. *Comput. Sci. Review 15–16*, 29–62.

Schlick, R., M. Felderer, I. Majzik, R. Nardone, A. Raschke, C. Snook, and V. Vittorini (2018). A proposal of an example and experiments repository to foster industrial adoption of formal methods. In *Int. Symp. Leveraging Applications of Formal Methods, Verification and Validation (ISoLA)*, pp. 249–272.

Stamatelatos, M., W. Vesely, J. B. Dugan, J. Fragola, J. Minarick, and J. Railsback (2002). *Fault Tree Handbook with Aerospace Applications*. Office of safety & mission assurance, NASA HQ.

Sullivan, K. J. and J. B. Dugan (1998). Galileo user's manual & design overview. `www.cse.msu.edu/~cse870/Materials/FaultTolerant/manual-galileo.htm`. v2.1-alpha.

Sullivan, K. J., J. B. Dugan, and D. Coppit (1999). The Galileo fault tree analysis tool. In *Proc. 29th Int. Symp. on Fault-Tolerant Computing (FTCS)*, pp. 232–235.

Volk, M., S. Junges, and J.-P. Katoen (2018). Fast dynamic fault tree analysis by model checking techniques. *IEEE Trans. Industrial Informatics 14*(1), 370–379.

Zhang, Z., X. Liu, and Z. Bian (2018). Analysis of restricted-speed accidents using Fault Tree Analysis. In *Proc. 2018 ASME/IEEE Joint Rail Conf.*