HEAVY TRANSPORT AIRCRAFT RELIABILITY STUDY

S.Chiesa, Full Professor, DIASP- Politecnico di Torino, Italy
P.Gianotti, PhD Student, Politecnico di Torino, Italy
P. Maggiore, Researcher, DIASP - Politecnico di Torino, Italy

Abstract

The analysis of safety and reliability is of primary importance during the design of a modern. large complex aircraft. In other hands, the intrinsic complexity of large, multiple-redundant systems usually impose severe limitations on both the depth and the extension of this analysis. In this work. a computer program for the reliability analysis of a generic system is presented. underlining the advantages of a computer-based approach to the problem. The philosophy which stands behind such an approach consists in a tailorization of well-known Failure Modes and Effect Analysis and Fault Tree Analysis techniques. The results of the automatic analysis include symbolic evaluation of fault and functional trees, minimal-paths and minimal cut sets determination, sensitivity analysis.

Introduction

In this paper KBFTA (Knowledge-Based Fault Tree Analysis), a computer tool for Fault Tree automatic synthesis and analysis, is presented. Fault Tree Analysis (FTA) is a top-down technique that, starting from the definition of an undesired top event, leads to the combination of basic events that are both necessary and sufficient to cause it. FTA is nowadays a key tool for the evaluation of safety and reliability characteristics of a complex system such an aircraft.

To perform a FTA, four elements are required:

1) A model of the system (system logical description).
2) A tree construction algorithm (in our case, an Expert System).
3) An algorithm for symbolic/numeric tree evaluation.
4) A set of tools for the analysis of the results.

The main features of the analysis tool are shown in Figure 1.

The System Logical Description

The system under analysis can be described defining its components and the links between them.

Each component is modeled by a set of *logical rules* representing any possible output state as a boolean function of both the component state and inputs. The component is then modeled with a set of *mini-trees*, each referring to a particular condition (see Figure 2). The model of the component is thus *local* and *context-independent*, as may result from a Failure Modes and Effects Analysis (FMEA).

To represent the system model, the rules and the connections, a simple *description language* has been developed. This language, which is the bridge between the human operator and the computer, has been designed to assure a good degree of readability, even if its goal is

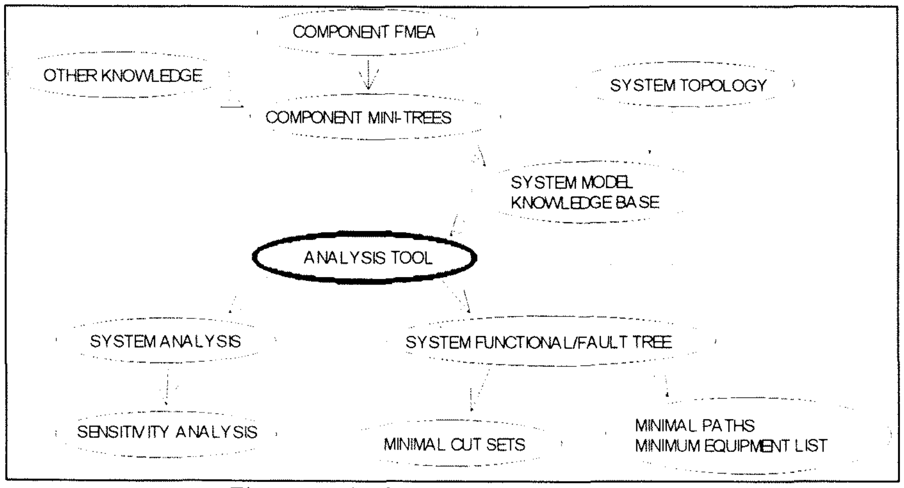
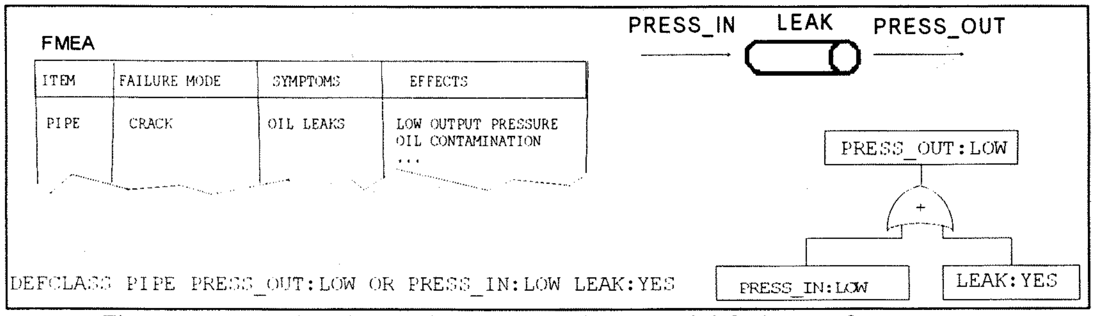


Figure 1. The features of the analysis tool.



ITEM	FAILURE MODE	SYMPTOMS	EFFECTS
PIPE	CRACK	OIL LEAKS	LOW OUTPUT PRESSURE OIL CONTAMINATION ...

```
DEFCLASS PIPE PRESS_OUT:LOW OR PRESS_IN:LOW LEAK:YES
```

Figure 2. FMEA, functional scheme, description and mini-fault-tree of a component

automatic processing. A few glimpses of the language will follow.

All the inputs, outputs and component states are considered as *events*. For computer processing purposes, all the events are referred to using the following syntax:

```
CLASS:SLOT:VALUE:INSTANCE
```

where:

- CLASS is the component type name.
- SLOT is the name of a given component variable.
- VALUE is the value of the slot.
- INSTANCE is the name of the individual component.

Notice that in this way it is possible to describe both binary-valued (i.e. fail / success, like in Reliability Block Diagrams) and multi-valued components.

For example, the fact (event) that a pipe P1 is leaking may be coded as:

```
PIPE:LEAK:YES:P1
```

While the fact that the input pressure of P1 is low may be referred to as:

```
PIPE:PRESS_IN:LOW:P1
```

Finally, the fact that a low output pressure may derive from a low input pressure or a leakage can be defined as follows:

```
DEF PIPE:PRESS_OUT:LOW:P1 OR
PIPE:PRESS_IN:LOW:P1 PIPE:LEAK:YES:P1
```

In the last expression, 4 elements can be identified:

1) DEF is the *command* that tells the computer that an event definition will follow.
2) PIPE:PRESS_OUT:LOW:P1 is the *event* being defined.
3) OR is the logical *operator* (gate),
4) PIPE:PRESS_IN:LOW:P1 and PIPE:LEAK:YES:P1 are the *operands* (i.e. the gate's inputs).

In order to describe the causal logic of an event, all the well known boolean operators are available: AND, OR, NOT, XOR. In addition, binomial parallels can be described using the M/N operator. Finally, the PRIMEV operator is used to define *primary* events, that don't need any further definition. So, going on with the example, since a rupture in the pipe can be considered a primary event (with a given probability, e.g. P=1E-5), it will be defined as follows:

```
DEF PIPE:LEAK:YES:P1 PRIMEV 0.00001
```

To model the behavior of a given class of components, the reference to a particular instance is not necessary, and the DEFCLASS command can be used, thus providing a much greater generality. In this way, for any pipe the following expression will hold:

```
DEFCLASS PIPE PRESS_OUT:HIGH AND
PRESS_IN:HIGH LEAK:NO
```

All the instances (i.e. all the individual components) belonging to a given class will share the same general model defined with DEFCLASS. To deal with specific components, the DEF command can be used to override this general model. In other words, DEFCLASS provides the *default*, standard description of a class of components, while DEF describe deviations from such a standard.

To describe the connections between the components (i.e. the topology of the system), the LINK command is available. So, to describe the fact that the 'input' side of the pipe P1 is connected to a pump U1, the following expression is necessary:

```
LINK PIPE:PRESS_INP:P1
PUMP:PRESS_OUT:U1
```

This command equates the values (whatever they are) of the slots PRESS_INP and PRESS_OUT of the two components P1 and U1. So, the pipe input pressure is related to the pump's output pressure being HIGH or LOW. In a similar way, the connection between the pipe and a rotating hydraulic motor E1 is described as:

```
LINK ROT_ENGINE:PRESS_INP:E1
```

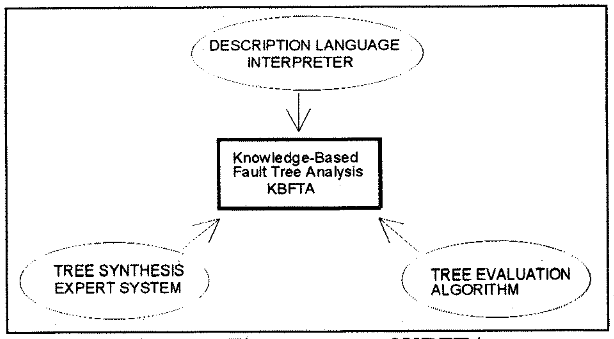
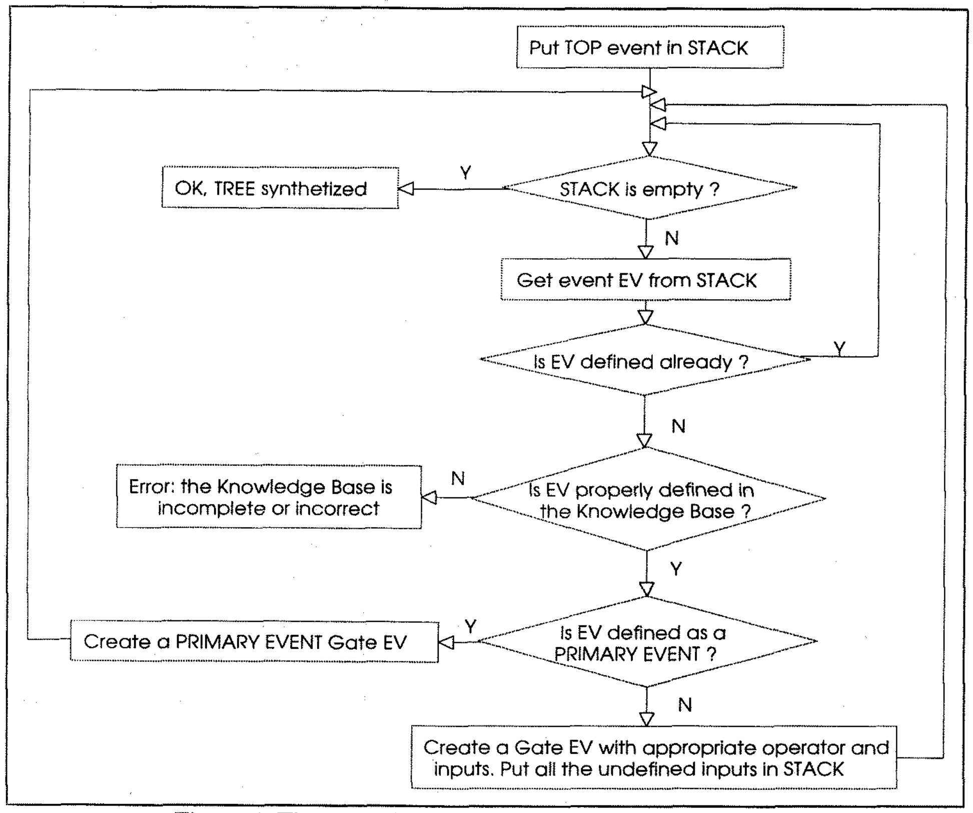

DESCRIPTION LANGUAGE INTERPRETER
Knowledge-Based Fault Tree Analysis KBFTA
TREE SYNTHESIS EXPERT SYSTEM
TREE EVALUATION ALGORITHM

Figure 3. The structure of KBFTA.



Put TOP event in STACK
STACK is empty ?
Y
OK, TREE synthetized
N
Get event EV from STACK
Is EV defined already ?
Y
N
Is EV properly defined in the Knowledge Base ?
N
Error: the Knowledge Base is incomplete or incorrect
Y
Is EV defined as a PRIMARY EVENT ?
Y
Create a PRIMARY EVENT Gate EV
N
Create a Gate EV with appropriate operator and inputs. Put all the undefined inputs in STACK

Figure 4. The core of the expert system's inference engine.

```
PIPE:PRESS_OUT:P1
```

To model complex connections (e.g. junctions, T-branches, etc.) the LINK command is often inadequate, and a logical description of the connection has to be performed via the DEF command.
The whole description of a system (called the Knowledge Base) is stored as a plain ASCII file, that can be fully edited, extended and updated on-line.

The Tree Synthesis Expert System

Once a system description is available, and a top event is given, a tree synthesis expert system is needed to *build* the tree. The core of the expert system is an *inference engine* that scans the knowledge base searching for the rules that could explain a given event (see Figure 3). Obviously, the expert system uses a *top-down* approach (FTA has a top-down approach itself !). In order to analyze the knowledge base, an appropriate *language interpreter* has been developed. The inference engine uses a *backward-tracking, deductive* technique: it starts from the effects and looks for the causes. When a cause is found, it is assumed as a new effect, and the process goes on until possible (i.e. until primary events are reached, or a lack of knowledge or an error is found). This is exactly what a human would do while constructing a fault tree. In the case of automatic construction, however, any risk of subjectivity is avoided, and it is evident the advantage in terms of speed and correctness. The risk of subjectivity obviously remains in the system description: this is accomplished solely on the basis of the operator's knowledge and insight. The expert system only checks the knowledge base for *formal correctness* and *completeness*: no check is possible about the *inherent correctness* of the models.
The algorithm is shown in Figure 4. Details about the implementation of the algorithm are beyond the scope of this paper, and just a few hints are given:

- Object Oriented Programming is used, providing a great flexibility.
- The tree is stored in memory as a dynamic structure, made of dynamically connected gates, corresponding to the usual tree gates.
- It is possible to display the tree or a part of it in a semi-graphic manner, for manual analysis purposes.

Only fault trees have been mentioned so far, but given the nature of the expert system, any top event can be considered and expanded, not only failures! This is a powerful tool, and thus other kinds of analysis can be performed: for example, the determination of the Minimum Equipment List of a system. In this case, the knowledge base must comprehend also the description of "normal", nominal events, not failures alone, and the result is a *functional tree*.

The Tree Analysis Algorithm

In FTA, the construction (synthesis) of the tree is just the first part of the problem, the last being the evaluation (analysis) of the tree.
In literature, an *exact evaluation* of the tree is usually avoided, and many algorithms have been presented to find the *Minimal Cut Sets* (MCS), i.e. the sets of events that are both necessary and sufficient to cause the top

event. In addition, many methods are available to treat the MCS to obtain approximate quantitative results. There are mainly three reasons for this:

1. MCS are self-explaining, and provide a very good qualitative insight of system's weaknesses and critical items.
2. A conservative, approximate solution is sufficient in many cases.
3. An exact solution is often unpractical, since it requires complex algorithms, long computational times and very large data storage capabilities.

In KBFTA, an exact boolean evaluation technique has been implemented, but approximate evaluation methods have been left as an alternative.

The exact solution is achieved via a symbolic boolean evaluation algorithm. This algorithm treat the tree as a boolean algebra expression, and expand it to an AND-OR sequence. In other words, the final expression is an OR gate containing only AND gates as inputs. In this way, in general the solution is found *without* searching for the MCS.

In boolean algebra, the main obstacle consist in the evaluation of the OR gates, since the total probability is in general *not equal* to the sum of the probabilities of the events in OR. Three methods are available in KBFTA to evaluate the probability of an OR gate:

1) P(A+B)= P(A)+ P(B)
2) P(A+B)= P(A)+ P(B) - P(A*B)
3) P(A+B)= P(A)+ P(B*NOT(A))

The first is very simple and fast, but is exact only if the events in OR are mutually exclusive. In this way, Minimal Cut Sets are found, and the probability of the top event is made equal to the sum of their probabilities. This is always conservative, but leads to an overestimation of the risk.

The second is exact, but its extension to multiple-inputs cases is expensive.

The third is exact too, but has proved to be very efficient. This is the default solution method in KBFTA.

The symbolic evaluation algorithm is depicted in Figure 5.

If the first evaluation method has been chosen (i.e. the user wants to find the MCS, not the exact solution), a super-set elimination procedure, that eliminates non-minimal cut sets, has to be executed.

A Cut-off procedure, that excludes events whose probability is lower than a given cut-off level, is available too.

Analysis of the results

Once the symbolic solution (exact or approximate) has been found, it can be evaluated numerically substituting to each event its probability. In addition, for each primary event E, a *sensitivity analysis* can be performed:

- It is possible to determine the weight W of E in the solution P (W is defined as the sum of the events containing E in the solution P).
- It is possible to calculate the new value P0 of the solution in the case the probability of E was forced to zero (if E is a failure, then P-P0 is the potential *risk reduction*).
- It is possible to calculate the new value P1 of the solution in the case the probability of E was forced to unity (if E is a failure, then P1-P is the potential *risk increase*).

So, when the reliability and safety characteristics have been found (i.e. the probabilities of the top events), it is

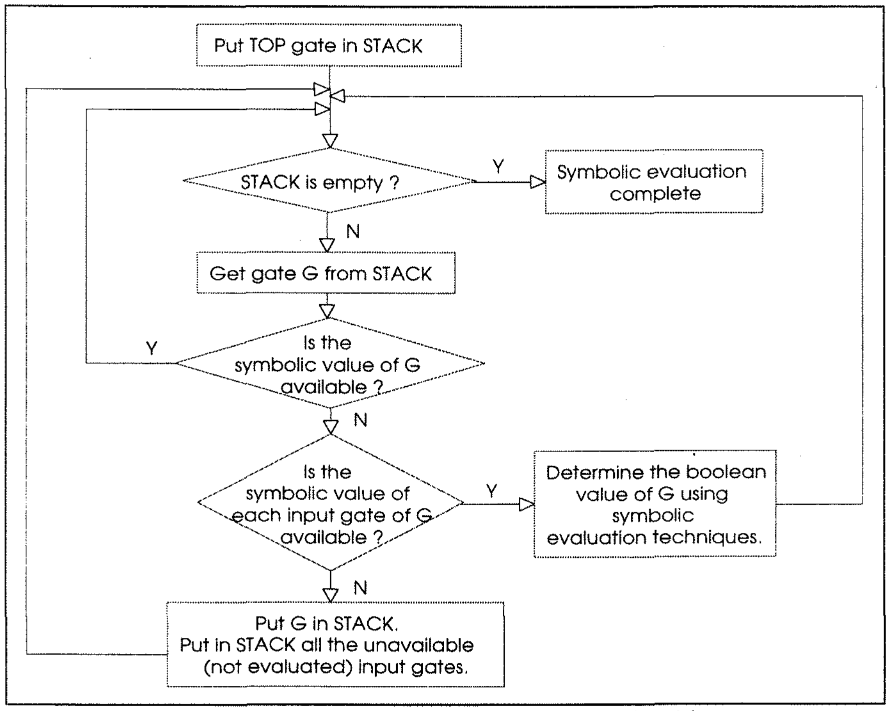

Put TOP gate in STACK
STACK is empty ?
Y
Symbolic evaluation complete
N
Get gate G from STACK
Is the symbolic value of G available ?
Y
N
Is the symbolic value of each input gate of G available ?
Y
Determine the boolean value of G using symbolic evaluation techniques.
N
Put G in STACK. Put in STACK all the unavailable (not evaluated) input gates.

Figure 5. The symbolic boolean evaluation algorithm.

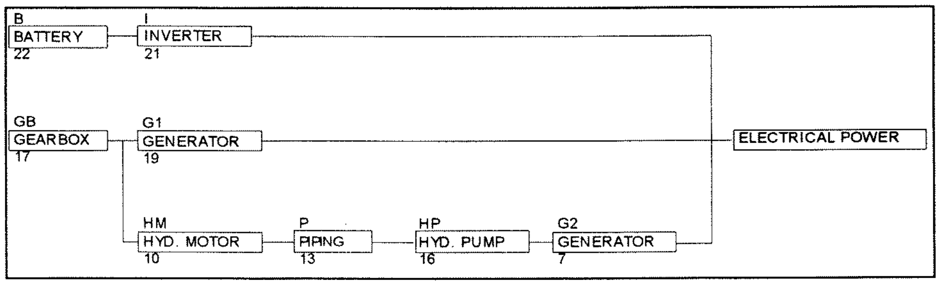


Figure 6. The electrical system.

possible to identify *where* a design modification is significant and effective.

An example

As an example, a simple system (the alternate current generation system of an aircraft) can be considered. Since the scope is demonstrating the capabilities of KBFTA, and not an accurate analysis, the system has been drastically simplified (e.g. the control logic and the switching circuitry has been removed from the model) and the probabilities given to the primary events are only indicative. The system is represented in Figure 6 (see Figure 7 for system's logical description) .
As a top event, the loss of electrical power istaken:

```
DEFCLASS BATTERY VOLT_OUT:LOW PRIMEV 0.00005
DEFCLASS INVERTER VOLT_OUT:LOW OR VOLT_IN:LOW UNIT:FAILED
DEFCLASS INVERTER UNIT:FAILED PRIMEV 0.000111
DEFCLASS GEARBOX POWER_OUT:LOW PRIMEV 0.00005
DEFCLASS GENERATOR VOLT_OUT:LOW OR POWER_IN:LOW UNIT:FAILED
DEFCLASS GENERATOR UNIT:FAILED PRIMEV 0.00005
DEFCLASS PIPE PRESS_OUT:LOW OR PRESS_IN:LOW LEAK:YES
DEFCLASS PIPE LEAK:YES PRIMEV 0.00002
DEFCLASS HYD_PUMP PRESS_OUT:LOW OR POWER_IN:LOW UNIT:FAILED
DEFCLASS HYD_PUMP UNIT:FAILED PRIMEV 0.000091
DEFCLASS HYD_MOTOR POWER_OUT:LOW OR PRESS_IN:LOW UNIT:FAILED
DEFCLASS HYD_MOTOR UNIT:FAILED PRIMEV 0.000077
LINK INVERTER:VOLT_IN:I BATTERY:VOLT_OUT:B
LINK GENERATOR:POWER_IN:G1 GEARBOX:POWER_OUT:GB
LINK GENERATOR:POWER_IN:G2 HYD_MOTOR:POWER_OUT:HM
LINK HYD_MOTOR:PRESS_IN:HM PIPE:PRESS_OUT:P
LINK PIPE:PRESS_IN:P HYD_PUMP:PRESS_OUT:HP
LINK HYD_PUMP:POWER_IN:HP GEARBOX:POWER_OUT:GB
DEF AIRCRAFT:EL_POWER:LOW:A AND
          INVERTER:VOLT_OUT:LOW:I
          GENERATOR:VOLT_OUT:LOW:G1
          GENERATOR:VOLT_OUT:LOW:G2
```

Figure 7. The system's logical description.

AIRCRAFT:EL_POWER:LOW:A

The resulting tree is shown in Figure 8, whose gates and primary events are given in Figure 9.
The exact solution can be seen in Figure 10, while in Figure 11 an approximate solution, containing only Minimal Cut Sets is shown. Both the solutions lay in the high reliability area (quite obviously: the loss of electrical power is a *safety-critical* failure !), and so the difference between them is not significant.
Finally, in figure 12 a sensitivity analysis is shown. It is evident that the gearbox unit has the highest relative weight, followed by the battery and the inverter. The improvement of the reliability characteristics of these items has the greatest influence on the system safety.

Conclusions

Started as an experiment and a training tool to evaluate the problems and the capabilities of automatic tree synthesis and analysis, KBFTA has developed to a full-scale design tool. Anyhow, it has maintained its original simplicity and flexibility. KBFTA's code is less than 200 Kbytes, and so it can be run on available PC's at a very high speed (the code is entirely stored in the CPU's high performance cache memory). Memory requirements depend on the size of the system under analysis, but for simple cases KBFTA can be run even on 640 Kbytes machines.
KBFTA can be used to evaluate the safety level of complex systems. As a matter of fact, the decision to develop KBFTA has been taken during the design program of the C1350 "Gulliver" aircraft. This design program [1] has grown up as a joint effort of students

```
-[AND]  005  (AIRCRAFT:EL_POWER:LOW:A)
 +----[OR ]  002  (INVERTER:VOLT_OUT:LOW:I)
 :    +----[AND]  020  (INVERTER:VOLT_IN:LOW:I)
 :    :    +----[PRIMEV]  022  (BATTERY:VOLT_OUT:LOW:B)
 :    +----[PRIMEV]  021  (INVERTER:UNIT:FAILED:I)
 +----[OR ]  003  (GENERATOR:VOLT_OUT:LOW:G1)
 ª    +----[AND]  018  (GENERATOR:POWER_IN:LOW:G1)
 ª    ª    +----[PRIMEV]  017  (GEARBOX:POWER_OUT:LOW:GB)
 ª    +----[PRIMEV]  019  (GENERATOR:UNIT:FAILED:G1)
 +----[OR ]  004  (GENERATOR:VOLT_OUT:LOW:G2)
      +----[AND]  006  (GENERATOR:POWER_IN:LOW:G2)
      ª    +----[OR ]  008  (HYD_MOTOR:POWER_OUT:LOW:HM)
      ª         +----[AND]  009  (HYD_MOTOR:PRESS_IN:LOW:HM)
      ª         ª    +----[OR ]  011  (PIPE:PRESS_OUT:LOW:P)
      ª         ª         +----[AND]  012  (PIPE:PRESS_IN:LOW:P)
      ª         ª         ª    +----[OR ]  014  (HYD_PUMP:PRESS_OUT:LOW:HP)
      ª         ª         ª         +----[AND]  015  (HYD_PUMP:POWER_IN:LOW:HP)
      ª         ª         ª         ª    +----[PRIMEV]  017  (GEARBOX:POWER_OUT:LOW:GB)
      ª         ª         ª         +----[PRIMEV]  016  (HYD_PUMP:UNIT:FAILED:HP)
      ª         ª         +----[PRIMEV]  013  (PIPE:LEAK:YES:P)
      ª         +----[PRIMEV]  010  (HYD_MOTOR:UNIT:FAILED:HM)
      +----[PRIMEV]  007  (GENERATOR:UNIT:FAILED:G2)
```

Figure 8. The fault tree.

```
                        List of events ( sorted by code )
+----------------------------------------------------------------------------+
|CODE| EVENT DESCRIPTION               | TYPE | PARAMETERS                    |
+----+---------------------------------+------+-------------------------------|
|   1|                             TRUE |PRIMEV|             P = 1.0000000000 |
+----+---------------------------------+------+-------------------------------|
|   2|         INVERTER:VOLT_OUT:LOW:I |  OR  |                              |
|    |                                 |      |       INVERTER:VOLT_IN:LOW:I |
|    |                                 |      |       INVERTER:UNIT:FAILED:I |
+----+---------------------------------+------+-------------------------------|
|   3|      GENERATOR:VOLT_OUT:LOW:G1  |  OR  |                              |
|    |                                 |      |   GENERATOR:POWER_IN:LOW:G1 |
|    |                                 |      |   GENERATOR:UNIT:FAILED:G1  |
+----+---------------------------------+------+-------------------------------|
|   4|      GENERATOR:VOLT_OUT:LOW:G2  |  OR  |                              |
|    |                                 |      |   GENERATOR:POWER_IN:LOW:G2 |
|    |                                 |      |   GENERATOR:UNIT:FAILED:G2  |
+----+---------------------------------+------+-------------------------------|
|   5|        AIRCRAFT:EL_POWER:LOW:A  |  AND |                              |
|    |                                 |      |      INVERTER:VOLT_OUT:LOW:I |
|    |                                 |      |    GENERATOR:VOLT_OUT:LOW:G1 |
|    |                                 |      |    GENERATOR:VOLT_OUT:LOW:G2 |
+----+---------------------------------+------+-------------------------------|
|   6|      GENERATOR:POWER_IN:LOW:G2  |  AND |                              |
|    |                                 |      |   HYD_MOTOR:POWER_OUT:LOW:HM |
+----+---------------------------------+------+-------------------------------|
|   7|       GENERATOR:UNIT:FAILED:G2  |PRIMEV|             P = 0.0000500000 |
+----+---------------------------------+------+-------------------------------|
|   8|     HYD_MOTOR:POWER_OUT:LOW:HM  |  OR  |                              |
|    |                                 |      |    HYD_MOTOR:PRESS_IN:LOW:HM |
|    |                                 |      |    HYD_MOTOR:UNIT:FAILED:HM  |
+----+---------------------------------+------+-------------------------------|
|   9|      HYD_MOTOR:PRESS_IN:LOW:HM  |  AND |                              |
|    |                                 |      |         PIPE:PRESS_OUT:LOW:P |
+----+---------------------------------+------+-------------------------------|
|  10|      HYD_MOTOR:UNIT:FAILED:HM   |PRIMEV|             P = 0.0000770000 |
+----+---------------------------------+------+-------------------------------|
|  11|            PIPE:PRESS_OUT:LOW:P |  OR  |                              |
|    |                                 |      |          PIPE:PRESS_IN:LOW:P |
|    |                                 |      |              PIPE:LEAK:YES:P |
+----+---------------------------------+------+-------------------------------|
|  12|             PIPE:PRESS_IN:LOW:P |  AND |                              |
|    |                                 |      |   HYD_PUMP:PRESS_OUT:LOW:HP  |
+----+---------------------------------+------+-------------------------------|
|  13|                 PIPE:LEAK:YES:P |PRIMEV|             P = 0.0000200000 |
+----+---------------------------------+------+-------------------------------|
|  14|      HYD_PUMP:PRESS_OUT:LOW:HP  |  OR  |                              |
|    |                                 |      |     HYD_PUMP:POWER_IN:LOW:HP |
|    |                                 |      |     HYD_PUMP:UNIT:FAILED:HP  |
+----+---------------------------------+------+-------------------------------|
|  15|       HYD_PUMP:POWER_IN:LOW:HP  |  AND |                              |
|    |                                 |      |    GEARBOX:POWER_OUT:LOW:GB  |
+----+---------------------------------+------+-------------------------------|
|  16|        HYD_PUMP:UNIT:FAILED:HP  |PRIMEV|             P = 0.0000910000 |
+----+---------------------------------+------+-------------------------------|
|  17|       GEARBOX:POWER_OUT:LOW:GB  |PRIMEV|             P = 0.0000500000 |
+----+---------------------------------+------+-------------------------------|
|  18|      GENERATOR:POWER_IN:LOW:G1  |  AND |                              |
|    |                                 |      |    GEARBOX:POWER_OUT:LOW:GB  |
+----+---------------------------------+------+-------------------------------|
|  19|       GENERATOR:UNIT:FAILED:G1  |PRIMEV|             P = 0.0000500000 |
+----+---------------------------------+------+-------------------------------|
|  20|          INVERTER:VOLT_IN:LOW:I |  AND |                              |
|    |                                 |      |       BATTERY:VOLT_OUT:LOW:B |
+----+---------------------------------+------+-------------------------------|
|  21|          INVERTER:UNIT:FAILED:I |PRIMEV|             P = 0.0001110000 |
+----+---------------------------------+------+-------------------------------|
|  22|          BATTERY:VOLT_OUT:LOW:B |PRIMEV|             P = 0.0005000000 |
+----------------------------------------------------------------------------+
```

Figure 9. The gates of the tree.

getting their final degree in Aeronautical Engineering. The workload has been distributed in order to cover all the fields of design, and management of information sharing is carried on by the design staff. In the general field of technology assessment for the next generation of aircraft, the objective was to investigate the feasibility of a transport aircraft of unusual dimensions and weights (1350 tons). The extremely large dimensions lead to a great complexity of the aircraft systems, even if it was decided to use state-of-the-art technology only. As an example, in Figure 13 the primary pitch control system is shown. The application of FTA to a system of this kind is practically impossible without a tool like KBFTA (just consider the different operation sequences, the possible working states, the different levels of redudancy, the acceptable degraded-output conditions, etc.). So, the need of a tool like KBFTA has become evident soon, given both the intrinsic scientific problem and the consideration that, for an aircraft like C1350, safety becomes an even more critical factor. Thanks to KBFTA's ease of use and power, it was possible to analyze the system shown in Figure 13 and the other systems of the C1350 (this activity is not reported here for space reasons). Concluding, we can say that KBFTA has well reached its target.

```
Solution method :Complete
SYMBOLIC VALUE of gate 5 ( AIRCRAFT:EL_POWER:LOW:A )
 +-----------------------------------------------------------------------
 ¦ i ¦ ± ¦#EV¦ PROBABILITY      ¦ EVENTS IN AND...
 +---+---+---+-----------------+-----------------------------------------
 ¦  1¦  1¦  2¦  2.5000000E-0008¦   17  22
 ¦  2¦  1¦  3¦  5.5472250E-0009¦   17  21 -22
 ¦  3¦  1¦  4¦  2.2748863E-0012¦   16 -17  19  22
 ¦  4¦  1¦  5¦  5.0477224E-0013¦   16 -17  19  21 -22
 ¦  5¦  1¦  5¦  4.9992950E-0013¦   13 -16 -17  19  22
 ¦  6¦  1¦  6¦  1.1092886E-0013¦   13 -16 -17  19  21 -22
 ¦  7¦  1¦  6¦  1.9246901E-0012¦   10 -13 -16 -17  19  22
 ¦  8¦  1¦  7¦  4.2706756E-0013¦   10 -13 -16 -17  19  21 -22
 ¦  9¦  1¦  7¦  1.2497025E-0012¦    7 -10 -13 -16 -17  19  22
 ¦ 10¦  1¦  8¦  2.7729524E-0013¦    7 -10 -13 -16 -17  19  21 -22
 +-----------------------------------------------------------------------
Number of events in OR : 10
Numerical value of gate #  5 ( AIRCRAFT:EL_POWER:LOW:A ) :  3.0554494E-0008
The event AIRCRAFT:EL_POWER:LOW:A occurs every  3.2728409E+0007 missions
```

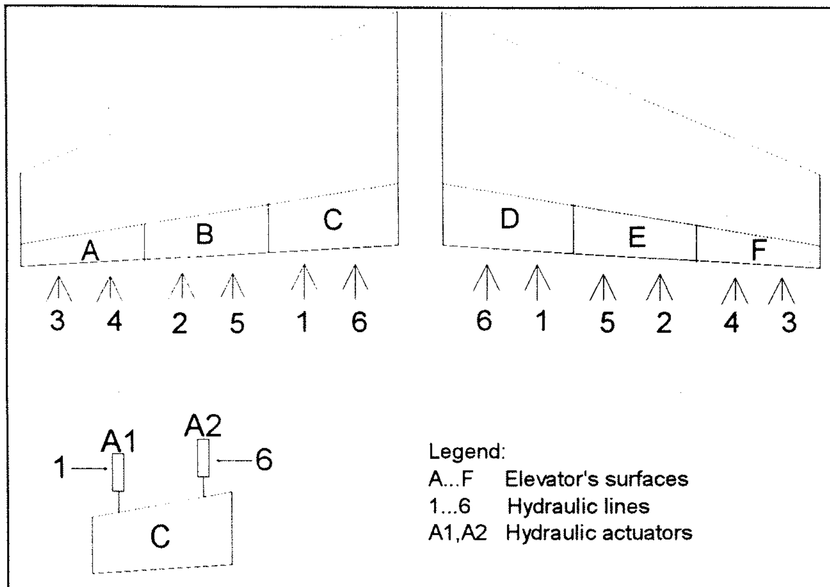
Figure 10. The exact solution.

```
Solution method :Minimal
SYMBOLIC VALUE of gate 5 ( AIRCRAFT:EL_POWER:LOW:A )
 +-----------------------------------------------------------------------
 ¦ i ¦ ± ¦#EV¦ PROBABILITY      ¦ EVENTS IN AND...
 +---+---+---+-----------------+-----------------------------------------
 ¦  1¦  1¦  2¦  2.5000000E-0008¦   17  22
 ¦  2¦  1¦  2¦  5.5500000E-0009¦   17  21
 ¦  3¦  1¦  3¦  2.2750000E-0012¦   16  19  22
 ¦  4¦  1¦  3¦  5.0505000E-0013¦   16  19  21
 ¦  5¦  1¦  3¦  5.0000000E-0013¦   13  19  22
 ¦  6¦  1¦  3¦  1.1100000E-0013¦   13  19  21
 ¦  7¦  1¦  3¦  1.9250000E-0012¦   10  19  22
 ¦  8¦  1¦  3¦  4.2735000E-0013¦   10  19  21
 ¦  9¦  1¦  3¦  1.2500000E-0012¦    7  19  22
 ¦ 10¦  1¦  3¦  2.7750000E-0013¦    7  19  21
 +-----------------------------------------------------------------------
Number of events in OR : 10
Numerical value of gate #  5 ( AIRCRAFT:EL_POWER:LOW:A ) :  3.0557271E-0008
The event AIRCRAFT:EL_POWER:LOW:A occurs every  3.2725436E+0007 missions
```

Figure 11. The approximate solution and the MCS.

```
Sensitivity Analysis on event: AIRCRAFT:EL_POWER:LOW:A
+-------------------------------------------------------+
¦EV ¦ EVENT DESCRIPTION             ¦     W      ¦  W/P   ¦
+---+-------------------------------+------------+--------¦
¦  7¦        GENERATOR:UNIT:FAILED:G2¦  1.5E-0012 ¦     0.0 ¦
¦ 10¦        HYD_MOTOR:UNIT:FAILED:HM¦  2.4E-0012 ¦     0.0 ¦
¦ 13¦                 PIPE:LEAK:YES:P¦  6.1E-0013 ¦     0.0 ¦
¦ 16¦         HYD_PUMP:UNIT:FAILED:HP¦  2.8E-0012 ¦     0.0 ¦
¦ 17¦        GEARBOX:POWER_OUT:LOW:GB¦  3.1E-0008 ¦   100.0 ¦
¦ 19¦        GENERATOR:UNIT:FAILED:G1¦  7.3E-0012 ¦     0.0 ¦
¦ 21¦           INVERTER:UNIT:FAILED:I¦  5.6E-0009 ¦    18.2 ¦
¦ 22¦           BATTERY:VOLT_OUT:LOW:B¦  2.5E-0008 ¦    81.8 ¦
+-------------------------------------------------------+
```

Figure 12. The sensitivity analysis.



Legend:

A...F	Elevator's surfaces
1...6	Hydraulic lines
A1,A2	Hydraulic actuators

Any of the six surfaces can be:
1) working normally
2) slowed down
3) stuck in any position
4) free and streamlined

An actuator can be:
1) working normally
2) slowed down
3) stuck in any position
4) free and damped

An hydraulic line can be
1) working normally
2) supplying low pressure
3) supplying no pressure

Problems:
Since each surface has 4 possible states, and the elevator has 6 independent surfaces, there are 4^6=4096 possible elevator states!
Any hydraulic line supplies both sides of the elevator, so a hydraulic failure determines a performance degradation of two different surfaces! So, for example, the state of surface "A" is not statistically-independent from the state of surface "F", thanks to the common hydraulic supply.

Figure 13. C1350's primary pitch control system.

References

1. S.CHIESA et alii, "Velivolo cargo di grandissime dimensioni: sviluppo del progetto e sperimentazione didattica", XIII Congresso Nazionale AIDAA, Roma, 11-15 settembre 1995.
2. E.PARZEN, "La moderna teoria delle probabilità e le sue applicazioni", Franco Angeli Editore, Milano 1976
3. S.CHIESA. "Affidabilità, sicurezza e manutenzione nel progetto dei sistemi", CLUT, Torino 1988.
4. S.RAI, K.K.AGGARWAL, "An efficient method for reliability eveluation of a general network", IEEE Transactions on Reliability, August 1978.
5. J.M. KONTOLEON, "A general approach for determining measures of repairable systems", Reliability Engineering, vol.2 no.1, 1981.
6. W.P.DOTSON, J.O.GOBIEN, "A new analysis technique for probabilistic graphs", IEEE Transactions on Reliability, October 1979.
7. K.K.AGGARWAL, S.RAI, "Symbolic reliability evaluation using logical signal relations", IEEE Transactions on Reliability, August 1978.
8. Y.B.YOO, N.DEO, "A comparison of algorithms for terminal-pair reliability", IEEE Transactions on Reliability, June 1988.
9. F.A.PATTERSON-HINE, B.V.KOEN, "Direct evaluation of fault trees using object-oriented programming techniques", IEEE Transactions on Reliability, June 1989.
10. A.POUCET, P.DE MEESTER, "Modular fault tree synthesis - A new method for computer-aided fault tree construction", Reliability Engineering, vol.2 no.1, 1981.
11. Y.LUNG CHEN, H.WEI, J.YUAN, "On establishment of I/O tables in automation of a fault tree synthesis", Reliability Engineering and System Safety, vol.40, 1993.
12. J.R.TAYLOR, "An algorithm for fault-tree construction", IEEE Transactions on Reliability, vol.R31 no.2, June 1982.
13. S.A.LAPP, G.J.POWERS, "Computer-aided synthesis of fault-trees", IEEE Transactions on Reliability, April 1977.
14. A.CARPIGNANO, A.POUCET, "Computer assisted fault tree construction: a review of methods and

concerns", Reliability Engineering and System Safety , 1994.

15. J.VATN, "Finding minimal cut sets in a fault tree", Reliability Engineering and System Safety, vol.36. 1992.
16. T.KOHDA, E.J.HENLEY, K.INOUE, "Finding modules in fault trees", IEEE Transactions on Reliability, vol.38 no.2, June 1989.
17. G.ZIPF, "Computation of minimal cut sets of fault trees: experiences with three different methods", Reliability Engineering, vol.7, 1984.
18. F.E.HASKIN, G.E.RADKE Jr., J EVANOFF, "Algorithms for analyzing the probability of M-out-of-N events", Reliability Engineering and System Safety, vol.47, 1995.
19. S.CONTINI, "A new hybrid method for fault tree analysis", Reliability Engineering and System Safety, vol.49, 1995.
20. I.NIEMELA, "On simplification of large fault trees", Reliability Engineering and System Safety, vol.44, 1994.
21. K.D.RUSSELL, D.M.RASMUSON, "Fault tree reduction and quantification- an overview of IRRAS algorithms", Reliability Engineering and System Safety, vol.40, 1993.
22. N.G.LEVESON, "Safeware: system safety and computers", Addison-Wesley Publishing Company, 1995.
23. A.RAUZY, "New algorithms for fault tree analysis", Reliability Engineering and System Safety, vol.40, 1993.