KfK 4527
Juli 1989

Algorithmen und Programme zur Auswertung von Fehlerbäumen mit Multi-state-Komponenten

A. Wickenhäuser
Institut für Reaktorentwicklung

Kernforschungszentrum Karlsruhe

Kernforschungszentrum Karlsruhe
Institut für Reaktorentwicklung

KfK 4527

Algorithmen und Programme zur Auswertung von Fehlerbäumen mit Multi-state-Komponenten

A. Wickenhäuser

Kernforschungszentrum Karlsruhe GmbH, Karlsruhe

Kernforschungszentrum Karlsruhe GmbH
Postfach 3640, 7500 Karlsruhe 1

ISSN 0303-4003

Zusammenfassung

Der Bericht gibt in Teil 1 und 2 einen zusammenfassenden Überblick über Methoden und Algorithmen zur Lösung von Problemen der Fehlerbaumanalyse. Im Einzelnen sind dies:

- Behandlung von Fehlerbaumkomponenten mit mehr als zwei Zuständen,
- Beschleunigung der Lösungsalgorithmen,
- Entmaschung und Modularisierung umfangreicher Systeme,
- Berechnung der vollständigen Strukturfunktion und der exakten Eintrittswahrscheinlichkeit,
- Behandlung statistischer Abhängigkeiten.

Ein flexibles Hilfsmittel zur Lösung dieser Probleme ist die Methode der Bildung von Boole'schen Variablen mit Einschränkungen (BVME). Mit Hilfe dieser Methode können Komponenten mit mehr als zwei Zuständen behandelt, die Möglichkeit der Modulbildung erweitert, und statistische Abhängigkeiten modelliert werden.

Teil 3 dieses Berichts enthält die Beschreibungen der auf diesen Methoden basierenden Computerprogramme MUSTAFA, MUSTAMO, PASPI und SIMUST.

Das Programm MUSTAFA wurde zu einem Instrument zur möglichst vielseitigen und vollständigen Analyse von Fehlerbäumen entwickelt; es enthält die Möglichkeiten:

- auf verschiedene Weise statistische Abhängigkeiten zu modellieren,
- eine inkohärente Boole'sche Funktion zu erkennen und gegebenenfalls auf eine kürzere Basis zu reduzieren,
- eine Strukturfunktion zu einander ausschließenden Termen umzuformen, um den exakten Zahlenwert der Wahrscheinlichkeit zu berechnen,
- Übergangsraten zwischen verschiedenen Zuständen eines mit komplementären Fehlerbäumen dargestellten Systems zu berechnen.

Umfangreiche und stark vermaschte Fehlerbäume, an deren Größe die Anwendung analytischer Fehlerbaumprogramme scheitert, können mit Hilfe des Programms MUSTAMO in kleinere Einheiten zerlegt werden. Es enthält die Möglichkeit, Module und Superkomponenten horizontal abzutrennen, und den Fehlerbaum vertikal in mehrere einfachere Bäume aufzuspalten.
Die Ergebnisse des Programms MUSTAMO lassen sich in übersichtlichen Blockdiagrammen darstellen. Die Zeilen eines Blockdiagramms sind Konjunktionen logisch unabhängiger Module.

Eine Hilfe zur Entmaschung und zur optimalen Anwendung der in MUSTAMO gebotenen Optionen bietet das Programm PASPI, das in mehreren Durchläufen die Struktur eines Fehlerbaums untersucht und alle kritischen Komponenten sowie die zur Abtrennung von Modulen und Superkomponenten geeigneten Knoten auflistet.

Das Programm SIMUST berechnet mit der Monte-Carlo-Methode die Vertrauensgrenzen der Eintrittswahrscheinlichkeit des Top-Ereignisses eines Fehlerbaums auf Grund der eingegebenen Streufaktoren der Wahrscheinlichkeitsdaten der Komponenten.

Der Anhang enthält Beispielrechnungen mit den Rechenprogrammen.

Algorithms and Programs for Evaluating Fault Trees with Multi-state Components

Abstract

Part 1 and 2 of the report contain a summary overview of methods and algorithms for the solution of fault tree analysis problems. The following points are treated in detail:

- Treatment of fault tree components with more than two states.
- Acceleration of the solution algorithms.
- Decomposition and modularization of extensive systems.
- Calculation of the complete structural function and the exact occurrence probability.
- Treatment of statistical dependencies.

A flexible tool to be employed in solving these problems is the method of forming Boolean variables with restrictions. In this way, components with more than two states can be treated, the possibilities of forming modules expanded, and statistical dependencies treated.

Part 3 of the report contains descriptions of the MUSTAFA, MUSTAMO, PASPI, and SIMUST computer programs based on these methods.

The MUSTAFA program has been developed into a tool for the most flexible and complete analysis of fault trees; it incorporates the following capabilities:

- Model statistical dependencies in various ways.
- Recognize an incoherent Boolean function and reduce it to a shorter base, if applicable.
- Rewrite a structural function into mutually exclusive terms in order to calculate the exact numerical value of the probability.
- Calculate transition rates between various states of a system represented by complementary fault trees.

Large, highly interconnected fault trees which, because of their size, make it impossible to apply analytical programs, can be broken down into smaller units by means of the MUSTAMO program. The program contains the possibility for horizontal separation of modules and supercomponents and for splitting the fault tree vertically into several simpler trees. The results of the MUSTAMO program can be represented in clear block diagrams. The lines of a block diagram are conjunction of logically independent modules.

Assistance in decomposition and in making optimum use of the options in MUSTAMO is provided by the PASPI program, which examines the structure of a fault tree in several runs and lists all critical components and the nodes suitable for separating modules and supercomponents.

The SIMUST program uses the Monte Carlo method to calculate the confidence bounds of the occurence probability of the top event of a fault tree in the light of the error factors entered for the probability data of the components.

The appendix contains examples of calculation performed with the computer programs.

Inhaltsverzeichnis

Einleitung

Der Bericht gibt in Teil 1 und 2 einen zusammenfassenden Überblick über die in /1 bis 7/ veröffentlichten Methoden und Algorithmen zur Lösung von Problemen der Fehlerbaumanalyse. Im Einzelnen sind dies:

- Behandlung von Fehlerbaumkomponenten mit mehr als zwei Zuständen,
- Beschleunigung der Lösungsalgorithmen,
- Entmaschung und Modularisierung umfangreicher Systeme,
- Berechnung der vollständigen Strukturfunktion und der exakten Eintrittswahrscheinlichkeit,
- Behandlung statistischer Abhängigkeiten.

Ein flexibles Hilfsmittel zur Lösung dieser Probleme ist die Methode der Bildung von Boole'schen Variablen mit Einschränkungen (BVME). Diese Methode ist mit der von Caldarola BAWRV (Boolean Algebra With Restricted Variables) /1/ genannten Methode identisch. Zur sprachlichen Präzisierung wurde die Nomenklatur geändert und zum besseren Verständnis der Methode wurde versucht, die Bildung der BVME auf einem einfacheren Weg zu herzuleiten.
In Abschnitt 1.2.4 ist der Suche nach der kleinsten irredundanten Basis einer inkohärenten Funktion gewidmet. Die Durchführung dieses in /6/ erwähnten aber nicht veröffentlichten Algorithmus wird hier im Detail vorgestellt.

Teil 3 dieses Berichts enthält die Beschreibungen der auf diesen Methoden basierenden Computerprogramme:

MUSTAFA (MUlti- STAte FAult- tree Analysis)
MUSTAMO (MUlti- STAte MOdularisation)
PASPI (Perform And Search for Points of Intersection)
SIMUST (SImulation with MUlti-STate-components)

Das Programm MUSTAFA wurde zu einem Instrument zur möglichst vielseitigen und vollständigen Analyse von Fehlerbäumen entwickelt; es enthält die Möglichkeiten:

- Multi-State-Komponenten zu behandeln,
- auf verschiedene Weise statistische Abhängigkeiten zu modellieren,
- eine inkohärente Boole'sche Funktion zu erkennen und gegebenenfalls auf eine kürzere Basis zu reduzieren,
- eine Strukturfunktion zu einander ausschließenden Termen umzuformen, um den exakten Zahlenwert der Wahrscheinlichkeit zu berechnen,
- Übergangsraten zwischen verschiedenen Zuständen eines mit komplementären Fehlerbäumen dargestellten Systems zu berechnen.

Umfangreiche und stark vermaschte Fehlerbäume, an deren Größe die Anwendung analytischer Fehlerbaumprogramme scheitert, können mit Hilfe des Programms MUSTAMO in kleinere Einheiten zerlegt werden. Es enthält die Möglichkeit:

- Module und Superkomponenten horizontal abzutrennen,
- den Fehlerbaum vertikal in mehrere einfachere Bäume aufzuspalten.

Die letztgenannte Option wird vor allem dann angewandt, wenn dadurch die Möglichkeit der horizontalen Abtrennung erleichtert wird. Die Ergebnisse des Programms MUSTAMO lassen sich in übersichtlichen Blockdiagrammen darstellen. Die Zeilen eines Blockdiagramms sind Konjunktionen logisch unabhängiger Module.

Eine Hilfe zur Entmaschung und zur optimalen Anwendung der in MUSTAMO gebotenen Optionen bietet das Programm PASPI, das in mehreren Durchläufen die Struktur eines Fehlerbaums untersucht und alle kritischen Komponenten sowie die zur Abtrennung von Modulen und Superkomponenten geeigneten Knoten auflistet.

Das Programm SIMUST berechnet mit der Monte-Carlo-Methode die Verteilung und Streubreite der Eintrittswahrscheinlichkeit des Top-Ereignisses eines Fehlerbaums auf Grund der eingegebenen Streubreiten der Wahrscheinlichkeitsdaten der Komponenten. Der Fehlerbaum muß in der Form logisch unabhängiger Module eingegeben werden.

Dieser Bericht ersetzt und erweitert die 1985 veröffentlichte Beschreibung der Programme MUSTAFA und MUSTAMO /15/.

Verwendete Symbole und Rechenregeln der Boole'schen Algebra

A,a,x	binäre Boole'sche Variable
$\bar{a}$	nicht a, Negation oder Komplement der Variablen a.
A0, B1	Kombination von Großbuchstaben und Ziffern, Variable mit Einschränkungen. (In Computerlisten ohne Unterschied für alle Boole'schen Variablen verwendet)
$\wedge$	Konjunktion, Und-Verknüpfung
$\vee$	Disjunktion, Oder-Verknüpfung [1]
$\overline{a \vee b} = \bar{a} \wedge \bar{b}$	Negation einer Disjunktion ist gleich der Konjunktion der negierten Variablen.
$\overline{a \wedge b} = \bar{a} \vee \bar{b}$	Negation einer Konjunktion ist gleich der Disjunktion der negierten Variablen.
$a \wedge a = a$	Idempotenz
$a \vee (a \wedge b) = a$	Absorption, der Ausdruck $a \wedge b$ wird von a absorbiert
$a \wedge \bar{a} = 0$	Exklusivität, a und das Komplement a schließen sich aus
Term, Monom	Boolescher Ausdruck bestehend aus mehreren mit $\wedge$ verknüpften Variablen, (Boole'sches Produkt)
Minterme	Terme, in denen alle in einer Boole'schen Funktion enthaltenen Variablen oder deren Negation enthalten sind. Minterme haben die Eigenschaft der Exklusivität, da sie sich durch die Negation mindestens einer Variablen unterscheiden.
Komponenten	Basiseinheiten eines Systems, deren Zustände in Strukturfunktionen mit Boole'schen Variablen indiziert werden.

Boole'sche Funktionen können in Tableaus angeschrieben werden.

$a\ b$
$a\ c$
$b\ c$

Die Variablen eines Tableaus sind innerhalb einer Zeile mit $\wedge$ (und), die Zeilen untereinander mit $\vee$ (oder) verknüpft. Das Tableau enthält die Funktion $F = (a \wedge b) \vee (a \wedge c) \vee (b \wedge c)$

---

[1] Disjunktion kann auch als Boole'sche Addition und Konjunktion als Boole'sche Multiplikation bezeichnet werden (+ •). Entsprechend kann der sprachlichen Vereinfachung wegen von Summen und Produkten Boole'scher Variabler gesprochen werden.

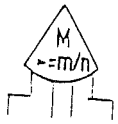
Symbole zur graphischen Darstellung von Fehlerbäumen

C_j = Basisvariable, Primärvariable, bezeichnet den j-ten Zustand einer Komponente C

D_j = Basisvariable der abhängigen Komponente D

Und - Gatter, GA = $a \wedge b$, Konjunktion der Vorgänger

Oder - Gatter, GV = $a \vee b$ Disjunktion

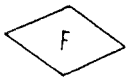


m aus n - Systeme

$\geq m/n$ mindestens m aus n Vorgängern

$= m/n$ exakt m aus n Vorgängern

$\leq m/n$ höchstens m aus n Vorgängern



unaufgelöster Unterbaum

Anschlußstellen, wenn Kante nicht durchgezogen.

1. Methoden und Algorithmen der angewandten Boole'schen Algebra

1.1 Boole'sche Variable mit Einschränkungen (BVME)

Eine Komponente in einem technischen System kann komplexer Natur sein, so daß das Verhalten der Komponente nicht mit einer binären Boole'schen Variablen indiziert werden kann. Würde man zur Beschreibung des Verhaltens mehrere Variable definieren, so könnten die Beziehungen zwischen den einzelnen Zuständen nicht exakt wiedergegeben werden, wenn man lediglich unabhängige Variable benutzen würde. Die von Caldarola gefundene Lösung des Problems geht davon aus, daß man aus mehreren unabhängigen zweiwertigen Variablen eine Unterfunktion (UF) bildet. Die Minterme einer UF enthalten alle Verknüpfungen der Konjunktion und Disjunktion, die zwischen den Variablen möglich sind. Es existieren dabei auch Minterme, die Verknüpfungen beinhalten, die in der Realität nicht auftreten. Es können nun einzelne Minterme oder die Summe mehrerer Minterme von Variablen mit Einschränkungen (BVME) symbolisiert werden. Unter der Voraussetzung, daß in den BVME alle Minterme einer UF enthalten sind, kann innerhalb der übergreifenden Strukturfunktion mit BVME gerechnet werden. Für einer Unterfunktion mit n BVME gilt:

Einschränkung 1	$\overset{n-1}{\underset{j=0}{\vee}} C_j = 1$
Einschränkung 2	$C_j \wedge C_k = 0 \quad j \neq k$
Komplementbildung	$\overline{C_j} = \overset{n-1}{\underset{k=0}{\vee}} C_{k,\, k \neq j}$

Die Einschränkung 1 verlangt, daß auf die n BVME alle 2^n Minterme der UF aufgeteilt sind. Die Einschränkung 2 bedeutet Exklusivität zwischen den BVME der selben UF. Die Regel zur Komplementbildung ist aus den Einschränkungen abgeleitet.

Diese Regeln sind bei der Durchführung von Rechenoperationen zu beachten. In der Regel ist es von Vorteil, eine negierte BVME so bald als möglich durch die nicht negierten BVME ihres Komplements zu ersetzen. Einer der Gründe dafür ist eine Besonderheit gegenüber dem Rechnen mit gewöhnlichen Boole'schen Variablen. Nach der Regel ist $C_j \wedge C_k = 0$, aber möglich ist $\overline{C_j} \wedge \overline{C_k} \neq 0$, wenn die UF mehr als 2 BVME enthält. Allgemein bedeutet dies, daß die Konjunktion negierter BVME ungleich 0 ist, wenn die Zahl der in einem Term verknüpfter negierter Variabler ein- und derselben UF um 1 kleiner ist als die Zahl der BVME, mit der die UF definiert ist.

Zur Unterscheidung der BVME von gewöhnlichen Boole'schen Variablen werden in diesem Bericht BVME mit Großbuchstaben und einer angehängten Ziffer angeschrieben, so daß die UF durch einen Buchstaben bzw. eine Buchstabenkombination und die definierte Aufteilung der Minterme durch Ziffern identifiziert werden kann.

In den folgenden drei Beispielen werden verschiedene Möglichkeiten der Definition von BVME aus den Mintermen einer UF gezeigt.

Funktion A

Die UF wird aus den zwei binären Variablen a,b für zwei Komponenten eines Systems gebildet. Die BVME repräsentieren je einen Minterm der UF.

Minterme	BVME	Ereignis bzw. Zustand
$\bar{a}\,\bar{b}$	A0	beide Komponenten intakt
$a\,\bar{b}$	A1	a ausgefallen
$\bar{a}\,b$	A2	b ausgefallen
$a\,b$	A3	beide ausgefallen

Die Bildung einer Unterfunktion aus mehreren binären Komponenten hat den Vorteil, daß für das von der UF repräsentierte Untersystem eine exakte Zustandsanalyse durchgeführt werden kann, deren Ergebnisse den BVME zugeordnet werden. Im obigen Beispiel kann in einer Zustandsanalyse die exakte Eintrittswahrscheinlichkeit von A3 berechnet und in die Wahrscheinlichkeitsrechnung der Strukturfunktion des Gesamtsystems eingesetzt werden, auch wenn wegen eines Common-Mode-Ausfalls der direkte Übergang von A0 zu A3 berücksichtigt werden muß, was nicht möglich wäre, wenn die Komponenten und ihr gemeinsamer Ausfall nur mit den unabhängigen Variablen a und b angezeigt wäre.

Funktion B

Die UF wird aus drei Komponenten a,b,c gebildet, die zusammen ein unabhängiges Untersystem bilden. Einzelne BVME sind Symbole einer Summe von Mintermen.

Minterme	BVME	Ereignis bzw. Zustand
$\bar{a}\,\bar{b}\,\bar{c}$	B0	Alle Komponenten des Untersystems intakt
$a\,\bar{b}\,\bar{c}$ $\bar{a}\,b\,\bar{c}$ $\bar{a}\,\bar{b}\,c$	B1	genau eine von drei Komponenten des Untersystems ausgefallen
$a\,b\,\bar{c}$ $a\,\bar{b}\,c$ $\bar{a}\,b\,c$	B2	genau zwei von drei Komponenten des Untersystems ausgefallen
$a\,b\,c$	B3	alle Komponenten des Untersystems ausgefallen

Die 2^n Minterme einer UF können in eine kleinere Anzahl von BVME aufgeteilt werden, wenn die Unterfunktion vom Gesamtsystem logisch unabhängig ist. Die Zahl der durch BVME indizierten Zustände ist abhängig davon, welchen Grad der Auflösung des Untersystems die Logik des Gesamtsystem verlangt. Die Zusammenfassung mehrerer Minterme unter je einer BVME vereinfacht die Darstellung der Funktion und verringert den Rechenaufwand.

Funktion S Die UF definiert die Zustände einer komplexen Komponente, deren Ausfallzustände unterschieden werden müssen.

Minterme	BVME	Ereignis bzw. Zustand
$s_1\ \bar{s}_2$	S1	Ausfallzustand 1
$\bar{s}_1\ s_2$	S2	Ausfallzustand 2
$\bar{s}_1\ \bar{s}_2$ $s_1\ s_2$	S0	Negation der Ausfallzustände,Komponente intakt

Caldarola /1/ hat aus diesem typischen Beispiel für eine Multistate-Komponente die BAWRV hergeleitet. Es ist die Modellierung der Logik eines automatischen Schalters, der ausgefallen in geöffnetem Zustand bei Anforderung nicht schließt und bei Ausfall in geschlossenem Zustand bei Anforderung nicht öffnet. Die unterschiedlichen Folgen eines falsch geschlossenen oder falsch geöffneten Stromkreises für das System, dem der Schalter als Komponente angehört, machen eine Unterscheidung in zwei Ausfallzustände notwendig. Von diesem Beispiel ausgehend kann die Bildung von Funktionen mit BVME für Multistate-Komponenten in folgender Weise dargelegt werden:

Eine Multistate-Komponente eines Fehlerbaums mit n Zuständen kann sich in $n - 1$ Ausfallzuständen oder im Intaktzustand befinden.

Werden die $n - 1$ Ausfallzustände mit gewöhnlichen binären Variablen bezeichnet, so entsteht eine Funktion mit 2^{n-1} Mintermen.

Es existieren $n - 1$ Minterme, die exklusiv einen Ausfallzustand zusammen mit der Negation aller anderen Ausfallzustände enthalten; für jeden dieser Minterme wird eine BVME definiert.

Alle übrigen Minterme bilden das Komplement zu den obengenannten $n - 1$ Mintermen und werden in der n-ten BVME zusammengefaßt.

Innerhalb der n-ten BVME existiert ein Minterm, der die Negation aller Ausfallzustände enthält, dies entspricht dem Intaktzustand. Alle anderen im Komplement enthaltenen Minterme bezeichnen physikalisch nicht mögliche Zustände.

Da sich ein- und dieselbe Komponente immer nur in einem der angegebenen Zustände befinden kann, können die Ausfallzustände nur durch durch Minterme vom oben beschriebenen Typ definiert werden. Die formale Logik verlangt die Vervollständigung der Funktion durch die in der n-ten BVME zusammengefaßten komplementären Summe aller übrigen Minterme. Die Zusammenfassung mehrerer Minterme unter einer BVME erlaubt es, die Erfordernisse der formalen Logik zu erfüllen.

Die Unterscheidung zwischen physikalisch möglichen und unmöglichen Zuständen wird in der Zustandsanalyse der Komponente getroffen. Es können nur Übergänge von den Ausfallzuständen zum Intaktzustand verlaufen, Übergänge zu den physikalisch nicht möglichen Zuständen existieren nicht, daher ist deren Eintrittswahrscheinlichkeit = 0.

Es ist nicht notwendig, die unmöglichen Zustände mit einer Boole'schen Operation zu eliminieren, sie treten in der Boole'schen Rechnung nicht explizit, sondern immer nur als Glieder einer BVME

auf, sie können auch das Ergebnis der angeschlossenen Wahrscheinlichkeitsrechnung nicht beeinflussen.

Die hier vorgestellte Methode der Variablen mit Einschränkungen ist mit der von Caldarola als Boole'sche Algebra mit beschränkten Variablen, - BAWRV, Boolean Algebra With Restricted Variables, - beschriebenen Methode der Sache nach identisch /1/. Die dort verwendete eigene Terminologie wurde nicht übernommen, weil die gewählten Begriffe mißverstanden werden und den Gedanken nahelegen könnten, die BAWRV sei eine Erweiterung der Boole'schen Algebra. Dies ist jedoch nicht der Fall. Das Rechnen mit BVME kann als Algorithmus innerhalb der Boole'schen Algebra betrachtet werden, der es erlaubt, Boole'sche Operationen auf der Ebene der Minterme von UF durchzuführen, ohne auf die unterste Ebene der binären Boole'schen Variablen absteigen zu müssen. Das Rechnen mit UF erlaubt es, Ergebnisse von Zustandsanalysen komplexer Komponenten und Untersysteme in eine einfache und für die Berechnungen in und mit der übergeordneten Strukturfunktion brauchbare, handliche Form zu bringen. Caldarola hat zunächst die BAWRV zur Modellierung der Multistate-Komponenten entwickelt und die Minterme der physikalisch unmöglichen Zustände mit der Definition einer Filterfunktion eliminiert. Im Laufe der weiteren Entwicklung, als die BAWRV zur Modellierung statistischer Abhängigkeiten und auf Module von Unterbäumen angewandt wurde, hat sich ergeben, daß diese Methode ein flexibles Instrument darstellt, mit dem unter einer BVME Minterme in beliebiger Zahl und Konstellation zusammengefaßt werden können. Daher können die Minterme unmöglicher Zustände ohne Anwendung einer Filterfunktion der n-ten BVME zugeordnet werden. Sie vervollständigen als Dummy-Elemente die logische Einheit der formalen Logik. Inhaltlich haben sie weder Einfluß auf die Boole'schen Operationen in der übergreifenden Funktion noch auf die nachfolgende Wahrscheinlichkeitsrechnung.

In den Computerprogrammen mit Anwendung der BVME werden aus Gründen einheitlicher Bearbeitung auch binäre Variable als UF mit zwei BVME aufgefaßt. Dies ist erlaubt, da die zwei Werte einer binären Variablen als die zwei Minterme der kleinstmöglichen UF aufgefaßt werden können. Existieren in einer UF nur zwei BVME, dann fallen die Rechenregeln für die BVME mit den allgemeinen Regeln der binären Boole'schen Algebra zusammen.

Die BVME sind die Basisvariablen der Fehlerbäume und Strukturfunktionen, die in den Rechenprogrammen bearbeitet werden. Alle Rechnungen können durchgeführt werden, ohne daß die BVME aufgelöst und und durch die binären Variablen, aus denen die UF gebildet wurde, dargestellt werden müßten.

Terme von Funktionen, die ausschließlich unaufgelöste BVME enthalten, können im übertragenen Sinne als Minimalschnitte, Minterme und Primimplikanten bezeichnet werden.

Für Minterme einer mit BVME dargestellten Funktion gilt:

- Die Zahl der BVME innerhalb eines Minterms entspricht der Zahl der in der Funktion enthaltenen UF.

- Die Anzahl der Minterme ist das Produkt aller der Zahlen, die für jede UF angeben, mit wievielen BVME sie repräsentiert wird.

Für eine aus den drei Unterfunktionen A, B, C bestehenden Funktion müssen die Zahlen n_A, n_B, n_C bekannt sein, die angeben, in wieviele eingeschränkte Variable die Minterme der UF aufgeteilt sind. Da nach der Einschränkung 2 alle Konjunktionen von Variablen ein- und derselben

Funktion 0 ergeben, bleiben in den Mintermen nur je eine BVME Variable einer UF - in diesem Falle drei - stehen.

Die Anzahl der aus BVME zusammengesetzten Minterme beträgt $n_A \cdot n_B \cdot n_C$, da BVME verschiedener UF sich zueinander wie gewöhnliche Boole'sche Variable verhalten.

Caldarola hat in /6/ darauf hingewiesen, daß sich Funktionen mit BVME von gewöhnliche Boole'schen Funktionen auch dadurch unterscheiden, daß sie hinsichtlich der Kohärenz nicht in allen Fällen symmetrisch sind. Eine kohärente Funktion mit BVME kann eine inkohärente Komplementfunktion besitzen.

1.2 Algorithmen mit Boole'schen Strukturfunktionen

Um die im Folgenden beschriebenen Algorithmen übersichtlich darzustellen und verständlich zu machen, ist ihre Durchführung allein mit binären Variablen dargestellt. Die Durchführung aller hier vorgestellten Algorithmen ist mit BVME möglich und in der Praxis erprobt.

1.2.1 Die Bildung einer Komplementfunktion

Gewöhnlich wird eine Boole'sche Funktion in der disjunktiven Normalform durch disjunkte Terme (Minimalschnitte, Primimplikanten) dargestellt. Wird diese Funktion negiert, so ist eine Umformung notwendig, wenn die negierte Funktion ebenfalls in der Normalform dargestellt werden soll.

Die Durchführung könnte auf dem Weg über die Minterme der Funktion erfolgen, indem aus der Liste aller Minterme diejenigen gestrichen werden, die einen Term der Funktion F überdecken. Die Liste der stehengebliebenen Minterme könnte dann nach der Regel $(a \wedge b) \vee (a \wedge \overline{b}) = a$ zu einer Liste von Minimalschnitten bzw. Primimplikanten reduziert werden. Eine zweite weniger aufwendige Methode wird meist bei Handrechnungen verwendet. Dabei wird aus den negierten Primimplikanten die jeweils am häufigsten auftretende Variable ausgeklammert. Die Anwendung dieser Methode wurde in /9/ mit Testbeispielen und der benötigten Rechenzeit veröffentlicht. Die angegebenen Zeiten aber betragen trotz Berücksichtigung der unterschiedlichen Geschwindigkeit der Computer ein Vielfaches von dem, was die im Programm MUSTAFA implementierte Methode der schrittweisen Multiplikation mit gezielter Absorptionsabfrage benötigt.

In der schrittweisen Multiplikation bedeutet der i-te Schritt die Konjunktion des nach dem Schritt i-1 vorliegenden Ergebnisses mit dem negierten i-ten Primimplikanten.

Die Aufgabe lautet:

$$F = \bigvee_{i=1}^{n} p_i \; ; \quad \overline{F} = \bigwedge_{i=1}^{n} \overline{p_i} = \bigvee_{k=1}^{m} q_k$$

Die Ausführung geschieht mit:

$$f_0 = 1 ; \quad f_i = f_{i-1} \wedge \overline{p_i} = \bigvee_{k=1}^{nx} q_{i,k} ; \quad f_n = \overline{F}$$

Die Zahl der im i-ten Schritt entstehenden Terme nx ist vorher unbekannt. Während der Rechnung kann diese Zahl bei einzelnen Schritte erheblich größer sein als die am Ende errechnete Zahl der m Primplikanten q der Komplementfunktion, es ist daher notwendig, ein genügend großes Arbeitsfeld bereitzustellen. Ohne Berücksichtigung von Idempotenz und Absorption wäre die Zahl der Terme nach dem n-ten Schritt kv^n, wenn man die durchschnittliche Anzahl von Variablen in den Termen mit kv angibt. Daher müssen die notwendigen Abfragen nach Idempotenz und Redundanz während der Durchführung der einzelnen Schritte erfolgen. Mit den folgenden Anweisungen konnte die Zahl der Abfragen minimiert werden.

- Jeder Term, der nach dem Schritt i-1 vorliegenden Produktsumme wird zu Beginn des i-ten Schritts mit den negierten Variablen des Primimplikanten p_i verglichen. Bei Feststellung einer Identität wird der Term wegen Idempotenz ungeändert in eine Liste U eingespeichert.

- Liegt keine Idempotenz vor, wird der Term mit den Variablen aus p_i, multipliziert. Die entstehenden Terme sind, wenn gegenseitiger Ausschluß besteht $= 0$, oder sie enthalten eine Variable mehr als die Terme der Liste U und werden in eine Liste V eingespeichert.

- Zur Suche nach möglicher Absorption müssen am Ende lediglich die Terme der Liste V mit den Termen der Liste U verglichen und gegebenenfalls wegen Absorption gestrichen werden.

Die Zahl der Vergleichsoperationen ist durch diese Maßnahmen auf ein Minimum reduziert. Die Suche nach möglicher Absorption entfällt ganz, wenn entweder alle während des i-ten Schritts erzeugten Terme in die Liste U oder alle in die Liste V eingespeichert wurden.

Der Algorithmus zur Komplementbildung einer Funktion wird benötigt, um durch zweimalige Negation einer Funktion die vollständige Liste F_{tot} aller Primplikanten der Funktion F zu erhalten. Die Berechnung der vollständigen Basis einer inkohärenten Funktion durch zweimalige Negation ist unter dem Namen *Nelson-Algorithmus* bekannt /6/.

$$G = \overline{F}; \quad F_{tot} = \overline{G}$$

Da der Algorithmus der schrittweisen Multiplikation nur wenig Aufwand an Rechenzeit benötigt, konnte er als Hilfsmittel zur Durchführung komplizierterer Algorithmen verwendet werden vor allem zur Feststellung ob eine Produktsumme 0 ergibt oder nicht.

1.2.2 Der Algorithmus zur Feststellung obligatorischer Primimplikanten

In der Fehlerbaumanalyse wird in der Regel die Zuverlässigkeit bzw. das Ausfallverhalten eines technischen Systems untersucht. Die durch Auflösung der Fehlerbäume erhaltenen Funktionen sind fast ausnahmslos kohärent. Inkohärente Funktionen, die man unter Funktionen findet, deren Terme aus negierten und nicht negierten Variablen zusammengesetzt sind, würden in der Anwendung auf die Zuverlässigkeit implizieren, daß auch der Übergang einer Komponente von Ausfallzustand in den intakten Zustand zum Ausfall des Systems beitragen würde. Inkohärente Funktionen treten daher bei Zuverlässigkeitsuntersuchungen sehr selten auf. Die Einführung der Variablen mit Einschränkungen haben Fragen nach Kohärenz und Inkohärenz aufgeworfen, so daß Caldarola Untersuchungen in dieser Richtung durchführte /6/, /4/.

Inkohärente Funktionen besitzen, wie bereits erwähnt, Primimplikanten, die nicht negierte und negierte Variable enthalten können. Sie haben die Eigenschaft, daß sie sich durch mehrere äquivalente Basisfunktionen darstellen lassen. Für jede inkohärente Funktion existiert eine vollständige Basis, die aus der Summe aller Primimplikanten besteht. Aus der Gesamtzahl aller Primimplikanten einer vollständigen Basis können mehrere äquivalente Basisfunktionen mit einer kleineren Anzahl an Primimplikanten gebildet werden. Vergleicht man die äquivalente Basisfunktionen, so findet man, daß die Primimplikanten eingeteilt werden können in:

- obligatorische Primimplikanten, die in jeder Basisfunktion enthalten sein müssen,
- nicht-obligatorische, die untereinander, einzeln oder kleinen Teilmengen, austauschbar sind,
- redundante Primimplikanten, die in allen Basisfunktionen fehlen können.

Der Algorithmus zur Feststellung, ob ein Primimplikant obligatorisch ist oder nicht, prüft, ob die um den zu untersuchenden Primimplikanten verminderte Basis mit der Basis äquivalent ist.

B = Basisfunktion,
Bk = die um den Primimplikanten p_k verminderte Basis.
B = $Bk \vee p_k$
B $\neq$ Bk , wenn Primimplikant k obligatorisch ist.
B = Bk, wenn Primimplikant k nicht obligatorisch ist.

Die Feststellung der Eigenschaft *obligatorisch* wird getroffen nach der Operation:

$$G = \overline{Bk} \wedge p_k ; \quad \overline{Bk} = \bigwedge_{j=1, j\neq j}^{n} \overline{p}_j$$

Jeder Primimplikant schließt sich mit der negierten Basis aus, daher wird G bei nicht obligatorischen Primimplikanten 0, da die verminderte Basis Bk in diesem Falle gleich der Basis B ist, im gegenteiligen Falle ungleich 0. Die Durchführung dieses Algorithmus kann erheblich verkürzt werden,

- da bei Idempotenz einer Variablen in p_k *und* $\overline{p}_j$ der j-te Schritt keine Änderung der Funktion G ergibt und deshalb übersprungen werden kann,
- da in vielen Fällen das Ergebnis G = 0 schon vor Erreichen des letzten Schritts festgestellt wird.

Dieser Algorithmus kann in zweifacher Weise angewandt werden zur:

- Feststellung des Anteils obligatorischer Primimplikanten in einer vollständigen Basisfunktion,
- Herstellung einer weniger umfangreichen Basisfunktion.

Im ersten Falle wird zur Berechnung der Testfunktion G in jedem Fall die vollständige Basis herangezogen.
Im zweiten Falle wird jeder gefundene nicht-obligatorische Primimplikant sofort aus der Basis gestrichen. Im weiteren Verlauf der Rechnung nehmen andere nicht-obligatorische Primimplikanten, die mit den gestrichenen austauschbar sind, die Eigenschaft von obligatorischen an.

1.2.3 Die Berechnung von Übergangsfunktionen

Unter Übergangsfunktionen werden hier nur die in nachstehender Formel definierten Funktionen verstanden, sie werden benötigt um die Übergangsraten und Übergangswahrscheinlichkeiten von und zu Systemzuständen, die mit Boole'schen Strukturfunktionen beschrieben sind, zu berechnen. Zu diesem Zwecke wurde für die Fehlerbaumprogramme ein entsprechendes Unterprogramm entwickelt, das den Boole'schen Teil der Berechnung der Übergangsfunktionen enthält. Caldarola fand dann aber heraus, daß Übergangsfunktionen eine entscheidende Bedeutung bei der Suche nach der kleinsten irredundanten Basisfunktion zukommt.

Eine Übergangsfunktion TF für den Übergang eines Systems vom einen Zustand in einen anderen setzt sich zusammen aus den Termen tf der einzelnen Komponenten. Die tf einer in F enthaltenen Variablen x zu ihrem in F enthaltenen Komplement $\bar{x}$ ist definiert als

$$tf(x \rightarrow \bar{x}) = F \wedge \overline{F} \mid_{\bar{x}=1}$$

und wird errechnet aus der Konjunktion der Funktion mit ihrer Komplementfunktion, unter der Voraussetzung, daß das in der Funktion $\overline{F}$ enthaltene Komplement $\bar{x}$ einer in F enthaltenen Variablen x gleich 1 gesetzt wurde. Der Übergang von $x \rightarrow \bar{x}$ kann innerhalb der Operation als sicheres Ereignis bezeichnet werden Die vollständige TF einer Funktion setzt sich zusammen aus den einzelnen tf_i aller der in der Funktion F enthaltenen nc Variablen.

$$TF(F \rightarrow \overline{F}) = \bigvee_{i=1}^{nc} tf(x_i \rightarrow \bar{x}_i)$$

Die Durchführung der notwendigen Rechenoperationen kann auf eine einmalige Und-Verknüpfung der Funktion F mit ihrer Komplementfunktion reduziert werden nach den folgenden Anweisungen:

- Bilde die Konjunktion jedes Primimplikanten der Funktion F mit jedem Primimplikanten der Komplementfunktion $\overline{F}$.

- Bewahre alle diejenigen Terme auf, die den Übergang nur einer einzigen Variablen x_i enthalten und entferne das Komplement $\bar{x}_i$, das nach obiger Formel $=1$ gesetzt wurde. (Alle diejenigen Terme, die mehr als einen Übergang enthalten, werden $= 0$).

- Ordne die aufbewahrten Terme nach Übergängen von Variablen, $TF(x_1 \rightarrow \bar{x}_1, \ldots, x_{nc} \rightarrow \bar{x}_{nc}) = tf(x_1), \ldots, tf(x_{nc})$.

Die Absorption der im Laufe der Rechnung entstandenen redundanten Terme wird je nach Einsatz der TF verschieden gehandhabt.

- Bei Einsatz der TF zur Berechnung der Wahrscheinlichkeitsdaten von Übergängen wird Absorption redundanter Terme nur innerhalb der $tf(x_i)$ der einzelnen Variablen durchgeführt. Redundanz besteht nicht zwischen den Termen der Übergänge verschiedener Variablen.

- Bei Einsatz der TF zur Berechnung der kleinsten irredundanten Basis kann Absorption redundanter Terme innerhalb der gesamten TF durchgeführt werden.

Die Berechnung einer Übergangsfunktion wird im folgenden Abschnitt als Teil der Durchführung des Algorithmus zur Berechnung der kleinsten irredundanten Basis gezeigt.

1.2.4 *Der Algorithmus zur Suche nach der kleinsten irredundanten Basis*

Nach Auflösung eines Fehlerbaumsliegt die Funktion F_a in Form von Minimalschnitten vor ($MS_j, j = 1,m$). In kohärenten Systemen existiert nur eine Basis, die MS sind mit den Primimplikanten dieser Basis identisch. In nicht kohärenten Systemen kann die gesuchte kleinste irredundante Basis erheblich weniger Primimplikanten umfassen als die vollständige Basis und auch weniger als die Ausgangsfunktion. Zur Vorbereitung des Algorithmus zur Suche nach der kleinsten Basis müssen die Terme q_i der Komplementfunktion K, die Primiplikanten p_i der vollständige Basis F_{tot} und aus den p und q die Terme t_j der Übergangsfunktion TF berechnet werden.

$$F_a = \bigvee_{i=1}^{m} MS_i; \quad K = \overline{F}_a = \bigvee_{i=1}^{nk} q_i; \quad F_{tot} = \overline{K} = \bigvee_{i=1}^{n} p_i$$

$$TF = \bigvee_{j=1}^{nt} t_j$$

Die Anzahl der Terme, die nach der Komplementbildung, nach der Bildung der Übergangsfunktion und nach ähnlichen Operationen innerhalb des Algorithmus entstehen, ist unbekannt. Die in den Formeln hierfür eingesetzten Variablen (nk, n, nt, kx) sind bei ihrem ersten Auftreten Ergebnisse der jeweiligen Rechnung. Die folgenden Formeln (1) bis (3) beschreiben die Durchführung des Algorithmus. Die darin verwendeten Alpha-Symbole a_i bezeichnen die durch den Index i identifizierten Primimplikanten.

(1) $$t_j \wedge x_{j,i} = (t_j \wedge \overline{p_i}) \vee (t_j \wedge a_i)$$

(2) $$Y_j = \bigwedge_{i=1}^{n} x_{j,i} = \bigvee_{k=1}^{m} y_{j,k}$$

$$y_{j,k} = y(v)_{j,k} \wedge y(a)_{j,k}; \quad Y(a)_j = \bigvee_{k=1}^{m} y(a)_{j,k}; \quad Y_{tot} = \bigvee_{j=1}^{nt} Y(a)_j$$

(3) $$B = \overline{Y}_{tot} = \bigvee_{k=1}^{k=kx} b_k$$

Zu (1) Zu berechnen ist ein Ausdruck $x_{j,i}$ Dieser Ausdruck wird entweder 1 oder er besteht aus einer Summe von Variablen und einem Alpha-Symbol.

$x_{j,i} = 1$, wenn Idempotenz einer Variablen in t_j und $\overline{p_i}$ festgestellt wird. In diesem Fall schließen t_j und a_i einander aus, und die übrigen Produkte aus der Konjunktion $t_j \wedge \overline{p_i}$ werden absorbiert. Im nachfolgenden Demonstrationsbeispiel sind, wie aus den Ordnungszahlen ersehen werden kann, alle $x_{j,i} = 1$ weggelassen.

Es existieren in $\overline{p_i}$ Variable, die sich mit t_j ausschließen, so daß ein Ausdruck $x_{j,i} = v_i \vee a_i$ entsteht. Mit v_i sind die in $\overline{p_i}$ noch verbliebenen Variablen bezeichnet.

Der Term t_j wird $=1$ gesetzt, so daß der folgende Schritt nur mit den $x_{j,i}$ durchgeführt wird.

Zu (2) Durch Konjunktion der Ausdrücke $x_{j,i}$ entstehen die Terme $y_{j,k}$. Diese bestehen entweder aus Alpha-Symbolen oder allein aus Variablen oder es sind gemischte Terme aus Variablen und Alpha-Symbolen. Die Unterscheidung zwischen Alpha-Symbolen und Variablen innerhalb eines Terms ist in der obigen Formel mit $y(v)_{j,k}$ *und* $y(a)_{j,k}$ angegeben. Für die weitere Rechnung werden aus den $y_{j,k}$ alle Variablen gestrichen, so daß die Funktionen $Y(a)_j$ und ihre Summe Y_{tot} allein noch Alpha-Symbole enthalten. Innerhalb dieser Funktionen kann auf die $Y(a)_j$ das Absorptionsgesetz angewandt werden, auch wenn die einzelnen $y(a)_{j,k}$ aus den oben erwähnten gemischten Ausdrücken $y_{j,k}$ stammen.

Zu (3) Die Bildung des vollständigen Komplements zu Y_{tot} ergibt eine unbekannte, u.U. sehr große Anzahl von Termen b_k. Da aber nur der kleinste, die geringste Zahl an Alpha-Symbolen enthaltende Term gesucht wird, kann die Komplementbildung mit schrittweiser Multiplikation so durchgeführt werden, daß zur Ausführung des folgenden Schritts, die im vorangegangenen Schritt entstandenen umfangreicheren Terme vernachlässigt werden und nur die kleinsten zur Produktbildung herangezogen werden. Eine zweite Möglichkeit auf schnellem Wege den kleinsten Term zu finden besteht darin, daß die Alpha-Symbole mit der größten Häufigkeit ausgeklammert werden.

Die Terme b_k enthalten Konjunktionen negierter Alpha-Symbole, Das bedeutet Konjunktion negierter Primimplikanten. Die Ausführung dieser Konjunktionen ergibt in jedem Falle $b_k = \overline{F}$ die Komplementfunktion. Somit ist $\overline{b_k} = F$, das bedeutet, daß die Summe der von den in b_k symbolisierten Primimplikanten eine Basis der Funktion F darstellt.

Eine vollständige Beweisführung für diesen Algorithmus liegt nicht vor, daher mußte bei der Implementierung im Programm MUSTAFA eine Proberechnung angefügt werden. Die Proberechnung enthält im ersten Teil die Überprüfung, ob die durch Negation der kleinsten Basis errechnete Komplementfunktion identisch ist mit der durch Negation der Ausgangsfunktion errechneten Komplementfunktion. Im zweiten Teil der Proberechnung wird untersucht, ob alle in der gefundenen kleinsten Basis enthaltenen Primimplikanten innerhalb dieser Basis obligatorisch sind. Ein Versagen des Algorithmus wurde in keinem der gerechneten Fälle gemeldet.

Die folgenden Tabellen enthalten die Durchführung eines Testbeispiels mit 5 binären Variablen.

Eingangsfunktion		Komplementfunkt.		vollständige Basis
MS	k	q	i	p
$a\ b\ c$	1	$\bar{b}\ \bar{d}$	1	$a\ b\ c$
$a\ d\ \bar{e}$	2	$\bar{c}\ \bar{d}$	2	$a\ b\ d$
$b\ c\ \bar{d}$	3	$\bar{a}\ \bar{b}\ \bar{e}$	3	$a\ \bar{c}\ d$
$\bar{c}\ d\ e$	4	$\bar{a}\ \bar{c}\ \bar{e}$	4	$a\ d\ \bar{e}$
$\bar{a}\ d\ e$	5	$\bar{a}\ d\ \bar{e}$	5	$\bar{a}\ d\ e$
	6	$a\ \bar{b}\ c\ e$	6	$b\ c\ \bar{d}$
			7	$b\ c\ e$
			8	$b\ d\ e$
			9	$\bar{c}\ d\ e$

Tabelle der vollständigen Übergangsfunktion

j	p(i),q(k) Übergang	t_j (absorbiert von)
1	(4) (5) $a\ (\rightarrow \bar{a})$	$a\ d\ \bar{e}$
2	(5) (6) $\bar{a}\ (\rightarrow a)$	$\bar{a}\ \bar{b}\ c\ d\ e$ (12)
3	(1) (6) $b\ (\rightarrow \bar{b})$	$b\ a\ c\ e$
4	(6) (1) $b\ (\rightarrow \bar{b})$	$b\ c\ \bar{d}$
5	(6) (2) $c\ (\rightarrow \bar{c})$	$c\ b\ \bar{d}$ (4)
6	(3) (6) $\bar{c}\ (\rightarrow c)$	$\bar{c}\ a\ \bar{b}\ d\ e$ (9)
7	(4) (1) $d\ (\rightarrow \bar{d})$	$d\ a\ \bar{b}\ \bar{e}$ (1)
8	(4) (2) $d\ (\rightarrow \bar{d})$	$d\ a\ \bar{c}\ \bar{e}$
9	(9) (2) $d\ (\rightarrow \bar{d})$	$d\ \bar{c}\ e$
10	(5) (1) $d\ (\rightarrow \bar{d})$	$d\ \bar{a}\ \bar{b}\ e$ (12)
11	(6) (5) $\bar{d}\ (\rightarrow d)$	$\bar{d}\ \bar{a}\ b\ c\ \bar{e}$ (4)
12	(5) (5) $e\ (\rightarrow \bar{e})$	$e\ \bar{a}\ d$
13	(4) (6) $\bar{e}\ (\rightarrow e)$	$\bar{e}\ a\ \bar{b}\ c\ d$ (7)

Schritt 1 und 2 mit der vollständigen Übergangsfunktion

j	i	$x_{j,i}$	j	Y_j
1	1	$\bar{b} \vee \bar{c} \vee a_1$	1	a_4
	2	$\bar{b} \vee a_2$		
	3	$c \vee a_3$		
	4	a_4		
2	5	a_5	2	a_5
3	1	a_1	3	$a_1 \ a_6 \ a_7$
	2	$\bar{d} \vee a_2$		$a_1 \ a_2 \ a_7 \ a_8$
	6	$d \vee a_6$		
	7	a_7		
	8	$\bar{d} \vee a_8$		
4	1	$\bar{a} \vee a_1$	4	a_6
	6	a_6		
	7	$\bar{e} \vee a_7$		
5	1	$\bar{a} \vee a_1$	5	a_6
	6	a_6		
	7	$\bar{e} \vee a_7$		
6	3	a_3	6	$a_3 \ a_9$
	9	a_9		
7	3	$c \vee a_3$	7	a_4
	4	a_4		
8	2	$\bar{b} \vee a_2$	8	$a_3 \ a_4$
	3	a_3		
	4	a_4		
9	2	$\bar{a} \vee \bar{b} \vee a_2$	9	$a_5 \ a_9$
	3	$\bar{a} \vee a_3$		$a_3 \ a_9$
	5	$a \vee a_5$		
	8	$\bar{b} \vee a_8$		
	9	a_9		
10	5	a_5	10	a_5
	9	$c \vee a_9$		
11	6	a_6	11	a_6
12	5	a_5	12	a_5
	7	$\bar{b} \vee \bar{c} \vee a_7$		
	8	$\bar{b} \vee a_8$		
	9	$c \vee a_9$		
13	4	a_4	13	a_4

Durchführung des 3. Schritts

Funktion Y_{tot} nach Absorption aller redundanten Terme:

$$(a_3\ a_9) + (a_6) + (a_1\ a_2\ a_7\ a_8) + (a_5) + (a_4)$$

Vollständige Liste aller b_k

$$b_1 = \overline{a_3}\ \overline{a_6}\ \overline{a_1}\ \overline{a_5}\ \overline{a_4}$$

$$b_2 = \overline{a_3}\ \overline{a_6}\ \overline{a_2}\ \overline{a_5}\ \overline{a_4}$$

$$b_3 = \overline{a_3}\ \overline{a_6}\ \overline{a_7}\ \overline{a_5}\ \overline{a_4}$$

$$b_4 = \overline{a_3}\ \overline{a_6}\ \overline{a_8}\ \overline{a_5}\ \overline{a_4}$$

$$b_5 = \overline{a_9}\ \overline{a_6}\ \overline{a_1}\ \overline{a_5}\ \overline{a_4}$$

$$b_6 = \overline{a_9}\ \overline{a_6}\ \overline{a_2}\ \overline{a_5}\ \overline{a_4}$$

$$b_7 = \overline{a_9}\ \overline{a_6}\ \overline{a_7}\ \overline{a_5}\ \overline{a_4}$$

$$b_8 = \overline{a_9}\ \overline{a_6}\ \overline{a_8}\ \overline{a_5}\ \overline{a_4}$$

Im Anhang befinden sich Computerlisten einer Durchführung des Algorithmus mit BVME und einer Beispielrechnung aus /9/.

1.2.5 Die Umformung einer Boole'schen Funktion zu einer äquivalenten Funktion mit einander ausschließenden Termen

Die Wahrscheinlichkeit eines Ereignisses, dessen logische Struktur in Form von Minimalschnitten einer Boole'schen Funktion vorliegt, kann in den meisten Fällen als Summe der Wahrscheinlichkeiten der Minimalschnitte berechnet werden. Dies ist aber nicht der exakte Wert, denn die Berechnung Wahrscheinlichkeit zweier Ereignisse a und b muss mit allen Mintermen der von a und b gebildeten Funktion errechnet werden:
$p\{a \vee b\} = p\{a \wedge \bar{b}\} + \{\bar{a} \wedge b\} + p\{a\} \cdot p\{b\}$

Da aber $p\{\bar{a}\} = 1 - p\{a\}$, *kann mit* $p\{a \vee b\} = p\{a\} + p\{b\} - p\{a\} \cdot p\{b\}$
vereinfacht werden.
Die obige Gleichung angewandt auf n Minimalschnitte einer Funktion ergibt eine Reihe von Termen, die nach einem bestimmten Prinzip geordnet die Wahrscheinlichkeit als Summe der Terme 1. Ordnung minus Summe der Terme 2. Ordnung plus Summe der Terme 3.Ordnung usw. ergeben. Terme k-ter Ordnung sind Produkte aus allen Kombinationen von k aus n Mintermen.

Es ist möglich die Minimalschnitte (= Terme 1. Ordnung) so zu erweitern, daß sie alle einander ausschließen. Damit werden die Terme 2. und höherer Ordnung zu 0, da sie Produkte der Terme 1. Ordnung sind. Die Erweiterung geschieht mit der Beziehung $a = (a \wedge b) \vee (a \wedge \bar{b})$; denn es ist $b \vee \bar{b} = 1$ *und* $p\{b\} + p\{\bar{b}\} = 1$

Die Erweiterung der Terme der Funktion muß durch Vergleiche der Terme untereinander fortgesetzt werden, bis alle Terme einander ausschließen. Nach dem Verfasser /11/ wird diese Methode *Bennett-Algorithmus* genannt.

Einen effektiveren Algorithmus hat Nakazawa /10/ veröffentlicht. Er wird hier unter der Bezeichnung *Keystone-Algorithmus* in der von Caldarola modifizierten Form vorgestellt. Er besteht darin, daß die Funktion zu gegenseitig sich ausschließenden Schlüsselmonomen K, die mit sogenannten einfachen Funktionen *sf* (simple functions) verknüpft sind, umgeformt wird.

$$F = \bigvee_{k=1}^{kx} K_k \wedge sf_k$$

Als einfache Funktion wird eine Funktion bezeichnet, deren Terme untereinander logisch unabhängig sind. Dies ist der Fall, wenn jede in einer *sf* enthaltene Variable mit der Häufigkeit 1 auftritt. Die exakte Wahrscheinlichkeit einer einfachen Funktion mit n Termen t kann mit einer Rekursionsformel berechnet werden.

Zuerst wird $P_1 = p\{t_1\}$ definiert, sodann erhält man mit der Formel $P_i = P_{i-1} + p\{t_i\} \cdot (1 - P_{i-1})$ mit P_n die exakte Eintrittswahrscheinlichkeit der *sf*.

Das erste Schlüsselmonom K_1 besteht zunächst aus der in in F am häufigsten auftretenden Variablen x. Die Konjunktion von x mit der Funktion F ergibt nach Streichung etwaiger 0-Produkte und absorbierter Terme die Funktion G_x.

$$x \wedge F = x \wedge G_x$$

Enthält G_x Variable mit der Häufigkeit > 1, so wird das zu bildende Schlüsselmonom um eine weitere Variable y erweitert und die Funktion
G_{xy} errechnet.

$$(x \wedge y) \wedge G_x = G_{xy}$$

Die Erweiterung des Schlüsselmonoms wird solange fortgesetzt bis G eine einfache Funktion ist.

$$(x \wedge y) \wedge G_{xy} = G_{xyz} \; ; \quad G_{xyz} = {}^{f}1 \; ; \quad k1 = (x \wedge y \wedge z)$$

Das zweite und die nachfolgenden Schlüsselmonome werden dadurch gebildet, daß die jeweils letzte Variable durch ihr Komplement ersetzt wird.

$$\textit{Wenn } K_1 = (x \wedge y \wedge z) \; , \quad \textit{dann ist } \quad K_2 = (x \wedge y \wedge \bar{z})$$

Das neugebildete Schlüsselmonom wird ebenso zuerst mit F multipliziert und gegebenenfalls erweitert bis eine einfache Funktion entstanden ist.
Wenn die letzte Variable und ihr Komplement bereits in die vorangegangenen Schlüsselmonomen eingesetzt waren, so wird sie gestrichen und das Komplement der vorletzten Variablen eingesetzt. Die Einsetzung der Variablen und ihrer Negationen wird durch Zähler kontrolliert. Bei Erweiterung eines Schlüsselmonoms wird für die eingesetzte Variable ein Zähler gesetzt, der bei Einsetzung des Komplements um 1 verringert wird. Die Abfrage der Zähler beginnt bei der letzten Variablen und wird nur solange fortgeführt, bis ein Zähler > 0 gefunden wird. Die dahinter stehenden Variablen mit dem Zähler 0 werden gestrichen.

Am Ende der Prozedur stehen die Zähler aller im letzten Schlüsselmonom enthaltenen Variablen auf 0.

Funktion F	Schlüssel-monom K	$F \wedge K$	einfache Funktion
a b *a c* *b c* *b d* *c e* *d e*	c	*a b* a b *b d* e *d e*	a b e
	$\bar{c}$	*a b* *b d* *d e*	
	$\bar{c}$ b	a d *d e*	a d
	$\bar{c}\ \bar{b}$	*d e*	*d e*

Durchführung des Keystone-Algorithmus

1.3 Entmaschung und Modulbildung

1.3.1 Die Vermaschung, Feststellung und Ursache

Die Auflösung eines Fehlerbaums bis zu dem Stande, wo die Strukturfunktion Primärvariable enthält, ohne Untersuchung und Durchführung möglicher Absorptionen, ergäbe die mögliche Gesamtmenge von Termen. Je nach Grad der Vermaschung enthielte diese Menge einen mehr oder weniger hohen Anteil an redundanten Termen, der bei starker Vermaschung um einige 10er-Potenzen höher sein kann als die Zahl der Primimplikanten bzw. der Minimalschnitte der Strukturfunktion. Es gibt ein einfaches Verfahren um die Zahl der Auflösungswege L_j an jedem Knoten K_j eines Fehlerbaums zu ermitteln. Hierzu wird jedem Basisknoten die Zahl L = 1 zugeteilt. Von den Basisknoten aufsteigend wird an jedem Knoten berechnet

$$L_j = L_i \cdot L_k \quad \text{, wenn} \quad K_j = K_i \wedge K_k$$
$$L_j = L_i + L_k \quad \text{, wenn} \quad K_j = K_i \vee K_k$$

Die an der Spitze eines Fehlerbaums errechnete Zahl L(TOP) kann darüber Aufschluß geben, ob ein umfangreicher und vermaschter Fehlerbaum innerhalb der von Speicherplatz und Rechenzeit gegebenen Begrenzung mit einem analytischen Rechenprogramm in einem Durchgang bearbeitet werden kann, oder ob Unterteilung in Module erforderlich ist.

Vermaschung wird allein von den Knoten (Basisknoten oder Gattern) verursacht, von denen mehr als eine Kante in Richtung zur Spitze des Baumes verlaufen. Dies hat zur Folge, daß die Funktionen mehrerer Knoten ein- und dieselbe Basisvariable enthalten und daher untereinander logisch abhängig sind. Logische Abhängigkeit von Funktionen hat zur Folge, daß bei ihrer Verknüpfung Terme entstehen, auf die das Idempotenzgesetz oder das Absorptionsgesetz oder das Gesetz des gegenseitigen Ausschlusses angewandt werden muß.

Zur Erläuterung der Prüfung von Boole'schen Funktionen wird die folgende Namensgebung verwendet:

Knoten mit mehr als zwei Ausgängen in Richtung zur Spitze des Baumes werden *kritische Knoten* genannt. *Kritische Komponenten* werden Komponenten genannt, wenn von einer ihrer Variablen mehr als zwei Wege ausgehen oder wenn mehr als ein Zustand dieser Variable als Basisvariable im Fehlerbaum vorhanden sind. Eine ununterbrochene Folge von Kanten, wobei sich jeweils eine Kante zum nächstfolgenden in Richtung zur Spitze liegenden Knoten anschließt, wird *Weg* genannt. Anfang und Ende eines Weges wird von Knoten definiert.

1.3.2 Behandlung kritischer Knoten bei Auflösung eines Fehlerbaumes

Die Auflösung eines Fehlerbaums von der Spitze her (Top-down) hat sich als brauchbar erwiesen. Sie wurde von J.B.Fussel als Grundlage des Computerprogramms MOCUS in übersichtlicher Tableau-Form vorgestellt /8/. In stark vermaschten Systemen kann aber auch dieser Algorithmus wegen der allzu großen Zahl von Auflösungswegen durch Überschreitung der Rechenzeit oder des zur Verfügung stehenden Speicherplatzes versagen. Mit einfachen Maßnahmen kann dieser Algorithmus erheblich beschleunigt werden:

Während der Bearbeitung eines Terms, der unaufgelöste Knoten enthält, hat die Auflösung nicht kritischer Knoten Priorität.

Diese Regel wird in den Programmen MUSTAFA und MUSTAMO angewandt, außerdem können im Programm MUSTAMO durch Eingabe einige Knoten zu so genannten Stop-gates deklariert werden. Die Auflösung wird dann in Stufen durchgeführt, zunächst bis zu den als Primärvariable behandelten Stop-gates, die erst in der folgenden Stufe aufgelöst werden. Erfahrungen mit vermaschten Systemen haben gezeigt, daß diese Maßnahme die Rechenzeit erheblich vermindern kann.
Wenn die Zahl der Auflösungswege des Fehlerbaums sehr viel größer ist als die seiner Negation, kann zunächst durch Auflösung des negierten Fehlerbaums die Komplementfunktion gefunden werden, deren Negation durchgeführt mit dem Algorithmus der schrittweisen Multiplikation die gesuchte Strukturfunktion des Fehlerbaums darstellt.

Wenn das Verhältnis der Auflösungswege eine bestimmte Schranke übersteigt, führt das Programm MUSTAFA diese Maßnahme automatisch durch.

1.3.3 Modulbildung ohne und mit Schlüsselkomponenten

Ist die Boole'sche Funktion eines Knotens K von der Boole'schen Funktion F logisch unabhängig, dann kann diese Funktion innerhalb des Systems als Modul, dargestellt durch eine binäre Variable, behandelt werden. Der Unterbaum wird danach nur noch repräsentiert durch seine Spitze, die als Basisvariable in den übergreifenden Baums des Gesamtsystems eingeht. Die logische Unabhängigkeit eines Knotens steht fest, wenn alle Wege, die von Knoten innerhalb des durch den Knoten K gebildeten Unterbaumes ausgehen durch den Knoten K verlaufen, und wenn in anderen Verzweigungen des Baumes keine zum Komplement einer Basisvariablen des Unterbaums gehörenden Elemente vorhanden sind. Innerhalb des Unterbaumes können kritische Knoten vorhanden sein. Wenn die von ihnen ausgehenden Wege spätestens am Knoten K wieder zusammentreffen, behält die Funktion des Knotens K ihre logische Unabhängigkeit. In Unterbäumen mit logisch abhängigen Funktionen können durch Ausklammern der Variablen kritischer Komponenten logisch unabhängige Module gebildet werden. Diese Maßnahme kann erheblich zur schnelleren und leichteren Auflösung eines vermaschten Fehlerbaums beitragen. Werden Funktionen von Unterbäumen, die auf diese Weise aufgeschlüsselt und vereinfacht wurden, in die Funktion des Systems eingesetzt, dann werden mit Sicherheit alle Idempotenzen kritischer Komponenten korrekt behandelt. Es kann aber hierbei nicht garantiert werden, daß nach Abschluß der Rechnung alle Redundanzen beseitigt sind, da die explizite Zusammensetzung aller Terme durch Modulbildung überdeckt ist. In Fehlerbäumen, deren Funktion den Ausfall eines Systems indiziert, errechnet die angeschlossene Wahrscheinlichkeitsrechnung bei Vorhandensein versteckter Redundanzen einen etwas erhöhten Zahlenwert und bewegt sich daher auf der sicheren Seite. Im Falle versteckter Idempotenzen würde das Ergebnis einen niedrigeren als den exakten Zahlenwert aufweisen.

1.3.4 Modulbildung mit Superkomponenten

Wenn zwei oder mehr Gatter wegen kritischer Komponenten oder kritischer Knoten voneinander abhängig, aber gemeinsam von der Funktion des Gesamtsystems unabhängig sind, kann eine Superkomponente mit mehr als zwei Zuständen gebildet werden, deren Zustände mit BVME indiziert werden. /7/

Aus zwei Knoten eines Fehlerbaums, K1 und K2, die wegen gemeinsamer Komponenten logisch abhängig sind, wird eine Superkomponente mit vier Zuständen gebildet, von denen drei Zustände im Programm errechnet werden:

$$SK1 = K1 \wedge \overline{K2}$$
$$SK2 = \overline{K1} \wedge K2$$
$$SK3 = K1 \wedge K2$$

Die Knoten K1 und K2 werden im Fehlerbaum durch die Knoten K1* und K2* ersetzt, die in folgender Weise definiert sind.

$$K1^* = SK1 \vee SK3 \; ; \quad K2^* = SK2 \vee SK3$$

Eine Superkomponente kann auch aus $k > 2$ Knoten gebildet werden. Da aber die Zahl der Zustände, deren Funktion berechnet werden muß $2^k - 1$ beträgt, wird man wohl nur in seltenen Ausnahmefällen mehr als 4 Knoten zu einer Superkomponente zusammenfassen.

Die Funktionen der einzelnen Zustände einer Superkomponente enthalten Negationen. In kohärenten, nur aus binären Variablen bestehenden Funktionen können die Negationen eliminiert werden. Man erhält auf diese Weise eine Funktion, die Caldarola *assoziierte kohärente Funktion* genannt hat. Der Nachweis, daß die assoziierten kohärenten Funktionen mit den Minimalschnitten identisch sind findet sich in /7/.

1.3.5 Vertikale Aufspaltung durch deterministische Komponenten

Die Methode der vertikalen Aufspaltung hat G. Weber /12/ als *splitting* beschrieben.

Existiert in einem Fehlerbaum eine kritische Komponente, von der Wege durch mehrere Unterbäume verlaufen, so kann diese Komponente zur deterministischen Komponente deklariert werden. Der Fehlerbaum muß dann so oft aufgelöst werden, wie diese Komponente Zustände besitzt. Die Ergebnisse der mehrfachen Auflösung werden addiert. Es sei F(Top) die Funktion des Fehlerbaums, und C die kritische Komponente, binär mit den zwei Zuständen C0 und C1, dann ist

$$F1 = F(Top) \wedge C0\,; \quad F2 = F(Top) \wedge C1\,; \quad F(Top) = F1 \vee F2$$

Zur Durchführung dieser Rechnung wird einmal C0, das andere mal C1 = 1 gesetzt. Wird die Rechnung mit mehreren kritischen Komponenten durchgeführt, wird F(Top) mit allen Mintermen, die sich mit den Variablen der ausgewählten Komponenten bilden lassen multipliziert.[2] Durch die Elimination kritischer Komponenten werden Funktionen von Unterbäumen logisch unabhängig, so daß diese als einfache Module behandelt werden können, denn bei der Auflösung des Baumes werden die Knoten, die eine Variable der deterministischen Komponente als Vorgänger besitzen, nach folgender Vorschrift umgeformt:

Knoten im ursprünglichen Fehlerbaum: $GO = A1 \vee C1$; $GA = B1 \wedge C1$

Wenn C0 = 1, C1 = 0, dann ist $GO = A1 \vee 0 = A1$; $GA = B1 \wedge 0 = 0$

Wenn C0 = 0, C1 = 1, dann ist $GO = A1 \vee 1 = 1$; $GA = B1 \wedge 1 = B1$

Der Vorteil dieser Methode besteht darin, daß geschickte Auswahl einer oder einiger wenigen Komponenten, die von diesen verursachten logischen Abhängigkeiten ausschaltet und die Bildung unabhängiger Module ermöglicht.

---

[2] In Funktionen mit BVME sind dies Minterme gemäß Definition in 1.1 Seite 8.

2. *Die Wahrscheinlichkeitsrechnung in den Fehlerbaumprogrammen*

2.1 Die Eintrittswahrscheinlichkeit der Komponentenzustände

Beide Programme berechnen die Eintrittswahrscheinlichkeit des Top-Ereignisses eines Fehlerbaums aus der mittleren asymptotischen Eintrittswahrscheinlichkeit der Komponentenzustände. Eine Berechnung der Wahrscheinlichkeit zu einem bestimmten Zeitpunkt kann in beiden Programmen nicht durchgeführt werden. Wenn angenommen werden kann, daß sich der Ausfall einer Komponente über die Zeit exponentiell verhält, kann die Berechnung der Wahrscheinlichkeit innerhalb der Programme auf Grund der eingegebenen Raten erfolgen. Hat das Ausfallverhalten eine andere Verteilung, so können die zuvor errechneten Wahrscheinlichkeiten direkt eingegeben werden.

Zur automatischen Berechnung werden in der Dimension $[\frac{1}{t}]$ die Übergangsraten $\lambda_{i\to j}$ vom Zustand i zu Zustand j einer Komponente benötigt. Die Raten sind die reziproken Werte der mittleren Zeiten bis zum Eintritt eines Ereignisses.

MTTF (mean time to failure)	Mittlere Zeit bis zum Übergang vom intakten Zustand zu einem Ausfallzustand, ergibt Ausfallrate.
MTTR (mean time to repair)	Mittlere Dauer einer Reparatur, wenn Fehler sofort entdeckt wird, ergibt Reparaturrate.
TI (inspection time)	Inspektionszeit, ergibt Regenerationsrate einer in regelmäßigen Abständen inspizierten Komponente.

Aus den Raten $\lambda_{j\to i}$ werden die asymptotischen Werte der Eintrittswahrscheinlichkeiten der Zustände 0 bis n-1 nach dem Ansatz der Differentialgleichung

$$\frac{dp_i}{dt} = \sum_{\substack{j=0 \\ j\neq i}}^{n-1} (\lambda_{j\to i} \cdot p_j) - p_i \cdot \sum_{\substack{j=0 \\ j\neq i}}^{n-1} \lambda_{i\to j}$$

Für eine binäre Komponente ergibt sich aus $\lambda_{1\to 0} = \frac{1}{MTTR}$ *und* $\lambda_{0\to 1} = \frac{1}{MTTF}$

$$p_0 = \frac{\lambda_{1\to 0}}{\lambda_{1\to 0} + \lambda_{0\to 1}} \; ; \; p_1 = \frac{\lambda_{0\to 1}}{\lambda_{1\to 0} + \lambda_{0\to 1}}$$

Für Komponenten mit mehr als zwei Zuständen wird ein Gleichungssystem mit n-1 Unbekannten aufgestellt, nachdem zunächst für $p_0 = 1$ eingesetzt wurde. Wenn in die Gleichungen die angegeben Startwerte eingesetzt werden,

$$p_i \cdot \sum_{\substack{j=0 \\ j\neq i}}^{n-1} \lambda_{i\to j} - \sum_{\substack{j=1 \\ j\neq i}}^{n-1} (p_j \cdot \lambda_{j\to i}) = \lambda_{0\to i}; \qquad p_i(start) = \frac{\lambda_{0\to i}}{\sum_{\substack{j=0 \\ j\neq i}}^{n-1} \lambda_{i\to j}}$$

kann das Gleichungssystem iterativ gelöst werden. Die Startwerte sind kleiner als die Lösung und es sind alle $\lambda_{j\to i} \geq 0$ und alle $p_i \geq 0$ So sind bei jedem Iterationsschritt Korrekturen nur in positiver Richtung möglich. Das Verfahren führt daher in jedem Falle zu einer Lösung, auch wenn nicht in allen Gleichungen das Diagonalelement überwiegt, wie es für das allgemeine Iterationsverfahren als Voraussetzung gefordert wird.

Da zunächst $p_0 = 1$ gesetzt wurde, müssen die Wahrscheinlichkeiten durch Multiplikation mit einem Korrekturfaktor FK normiert werden.

$$FK = \frac{1}{\sum_{i=0}^{i=n-1} p_i}$$

2.2 Die Behandlung statistischer Abhängigkeiten

2.2.1 *Modellierung mit Superkomponenten*

Statistische Abhängigkeiten zwischen einigen wenigen Komponenten können mit Bildung einer Superkomponente exakt dargestellt werden, da bei der Berechnung der Eintrittswahrscheinlichkeit der Zustände alle Übergänge berücksichtigt werden können. Ein einfaches Beispiel hierfür bietet das Bestehen eines Common-Mode-Ausfalls zweier Komponenten /7/.

Aus den beiden Komponenten a und b wird eine Superkomponente A mit 4 Zuständen gebildet, wie in Abschn. 1.1 beschrieben. Die Zustandsanalyse der Superkomponente weist einen direkten Übergang von Zustand $A0 = \bar{a} \wedge \bar{b}$ zu Zustand $A3 = a \wedge b$ auf. Die Übergangsrate von A0 zu A3 ist die Ausfallrate des Common-Mode-Ausfalls $\lambda_{0\to 3}$, die berücksichtigt werden kann, wenn die Wahrscheinlichkeiten mit der im vorigen Abschnitt beschriebenen Matrix berechnet werden.

2.2.2 *Die Methode der Inhibitorfunktionen*

Inhibitorfunktionen, auch master-functions /6/ genannt, definieren die Bedingungen, unter denen geänderte Wahrscheinlichkeitsdaten einer Komponente einzusetzen sind: Die Bedingungen lassen sich in Form von Boole'schen Funktionen darstellen, die Basisvariable des Fehlerbaums enthalten. Die Zustände einer Inhibitorfunktion werden in Form von Termen so eingegeben, daß sich die den Zuständen zugeordneten Teilfunktionen einander ausschließen und zu 1 ergänzen. Komponenten mit statischen Abhängigkeiten besitzen für jeden Zustand der ihnen zugeordneten Inhibitorfunktion definierte Wahrscheinlichkeitsdaten, die jeweils für den eingetretenen Zustand einzusetzen sind. Die Prüfung, welcher Zustand eingetreten ist geschieht in folgender Weise:

Es sei MS ein Minimalschnitt, der eine Variable der abhängigen Komponente D enthält. ID0, ID1, ID2 seien drei Zustände der zugeordneten Inhibitorfunktion. Die Zustände der Inhibitorfunktion sind definiert durch ihre Minimalschnitte und erscheinen so in der Eingabe. Das Programm errechnet daraus die negierten Zustands-Funktionen $\overline{ID1}$, $\overline{ID2}$, $\overline{ID0}$. Für die Komponente D sind

drei Sätze von Wahrscheinlichkeitsdaten definiert und daher für jede Variable drei Wahrscheinlichkeiten gespeichert.

Ergibt nun $MS \wedge \overline{ID1} = 0$,

so ist zur Berechnung der Eintrittswahrscheinlichkeit des Minimalschnitts MS die diesem Zustand der Inhibitorfunktion zugeordnete Eintrittswahrscheinlichkeit des Komponentenzustandes einzusetzen.

2.2.3 *Die Anwendung einer Teilungskomponente*

Auf gleiche Weise wie eine Inhibitorfunktion wird auch eine Teilungskomponente (partition function) gebildet. Auch diese besteht aus Teilfunktionen, die sich zu 1 ergänzen. Mit den n Zuständen der Teilungskomponente PT wird die in Minimalschnitten vorliegende Funktion eines Systems verknüpft.

$$F_{Top} = \bigvee_{j=0}^{j=n-1} (F_{Top} \wedge PT_j)$$

Jedem Zustand der Teilungskomponente wird ein dimensionsloser Faktor s_j zugeordnet, so daß das Ergebnis der Wahrscheinlichkeitsrechnung ergibt:

$$p\{F_{Top}\} = \sum_{j=0}^{j=n-1} p\{F_{Top} \wedge PT_j\} \cdot s_j$$

Dieses Verfahren wurde einmal durchgeführt, wobei das Verhalten eines Untersystems untersucht wurde, in dem bei einer bestimmten Konstellation ein Ausfall erst bei Inspektion einer bestimmten Komponente oder des ganzen Untersystems entdeckt werden konnte. Die Faktoren s_j wurden in einem speziellen Analyseprogramm errechnet. Sie ergaben sich aus dem Verhältnis der Unverfügbarkeit im optimalen Zustand (sofortige Reparatur aller auftretenden Fehler) und zeitabhängiger Unverfügbarkeit bei unterschiedlichen Inspektionszeiten. /13/,/14/.

2.3 Übergänge zwischen Systemzuständen

Systemzustände werden durch Fehlerbäume eines Systems, deren Boole'sche Funktionen einander ausschließen und in ihrer Gesamtheit sich zu 1 ergänzen, dargestellt.

Das Programm MUSTAFA enthält die Option, Übergangsraten zwischen Systemzuständen zu berechnen. Zur Erläuterung der Formel werden die folgenden Größen definiert:

Top , $\overline{Top}$ = Fehlerbaumfunktion und ihre Negation
$U_1 = p\{Top\}$ = Eintrittswahrscheinlichkeit des Top-Ereignisses des Fehlerbaums
$tt(x_i \rightarrow \bar{x}_i)$ = Term der Übergangsfunktion einer in *Top* enthaltenen Komponente x
$\lambda(x_i \rightarrow \bar{x}_i)$ = Übergangsrate der Komponente x zu ihrem Komplement
nx = Anzahl der in *Top* enthaltenen, Zustände von Komponenten indizierenden Variablen

ni = Anzahl der Terme der Übergangsfunktion einer Variablen

Die Bildung der Übergangsfunktion als Summe der Übergangsfunktionen der einzelnen Variablen wurde in Abschnitt 1.2 beschrieben.

Die Übergangsrate $TR(Top \rightarrow \overline{Top})$ wird berechnet mit:

$$TR(Top \rightarrow \overline{Top}) = \frac{1}{U_1}\left[\sum_{1}^{nx} \lambda(x_i \rightarrow \bar{x}_i) \cdot \sum_{1}^{ni} tt(x_i \rightarrow \bar{x}_i)_j\right]$$

Die Berechnung der Übergangsrate wurde hier zunächst mit binären Variablen dargestellt. Sie kann auch mit Funktionen durchgeführt werden, die BVME enthalten. Für eine Komponente C mit n-1 Zuständen ergeben sich von Zustand C_k so viele Übergänge mit ihren Funktionen, wie die Funktion $\overline{Top}$ komplementäre Variable $C_m, m \neq k$ besitzt, zu denen eine Übergangsrate > 0 vorhanden ist.

3. Beschreibungen der Computerprogramme

3.1 Das Programm MUSTAFA

3.1.1 Die Eingabe

Die Eingabedaten sind gegliedert in:

A Eingabe zur Programmsteuerung,
B Eingabe der Fehlerbaumdaten,
C Eingabe der Inhibitorfunktionen, wenn abhängige Komponenten vorhanden,
D Eingabe der Teilungskomponente, wenn Option 12 gewählt wurde.

A1) **(NUN(J),J+1,8)** **Format(8I2)**

NUN(J) Nummern externer Speichereinheiten. NUN(1) bis NUN(4) muß in jedem Fall mit einer Zahl > 0 angegeben werden, entsprechend 4 temporären Datensätzen, die in Jobs auf IBM-Maschinen mit JCL angelegt werden müssen. Für NUN(6) bis NUN (8) gilt das gleiche, wenn mit NTR > 0 die Berechnung von Übergangsraten verlangt wurde, Erläuterungen hierzu im folgenden Abschnitt *Anwendungsmöglichkeiten*.[3]

A2) **NOPT,NTR,NSE,NPRP,KPR1,KPR2,KPR3,KPR4,EPS,ALPH,BET,GAM**
Format(8I2,4x,4E10.4)

NOPT,NTR verlangte Option, Erläuterung im folgenden Abschnitt

NSE Wird nicht mehr benötigt

NPRP Steuerung des Aufrufs des Unterprogramms PREPAR zur Feststellung der Kohärenz. Aufruf erfolgt nur, wenn NPRP$\neq$ 0.

KPR1 bis KPR4 Steuerung der Druckausgabe, Erläuterungen im folgender Abschnitt.

EPS Schranke bei Vernachlässigung von Minimalschnitten (ms) mit geringen Gewicht. $ms_j < \epsilon \cdot \sum_{i=1}^{j-1} ms_i$

ALPH,BET,GAM Steuerungsgrößen bei Reduzierung der vollständigen Basisfunktion. Default-Werte $\alpha = 0.166$; $\beta = 0.75$; $\gamma = 0.125$ Mit diesen Werten wird

---

[3] In der bei Control Data installierten Version des Programms entfällt die Eingabe von A1). Die Speichereinheiten 1 bis 4 sind fest einprogrammiert. Die Angabe, welche von den Speichereinheiten (5) bis (8) zu Übergangsrechnungen benutzt werden sollen, werden in B1) eingegeben. Die Eingabe B1) lautet dort: (NUN(J),J=5,8), IDCASE ; Format(4I2,A64) Für die benötigten NUN(J) können nur die Zahlen 11, 12, 13, 14 verwendet werden.

der Algorithmus zur Berechnung der kleinsten irredundanten Basis in den seltensten Fällen erreicht. Mit der Eingabe ALPH $<<$ 1, BET = 1, GAM $<<$ 1 kann man nach Eingabe des Fehlerbaums eines inkohärenten Systems den Aufruf des Algorithmus zur Suche der kleinsten Basis erzwingen.

B1) **IDCASE** **Format(8x,A64)**

IDCASE Kommentar von maximal 64 Zeichen Länge zur Fallidentifikation[3]

B2) **NI,N1,N2,N3,N4** **Format(A4,2I4,I3,I1)**

NI maximal 4 Zeichen umfassende Identifikation einer Komponente

N1 Zahl der Zustände der Komponente, $2 \leq N1 \leq 8$. Die Zustände werden vom Programm automatisch numeriert von 0 bis N1-1.

N2 Zahl der in B3) einzugebenden Wahrscheinlichkeitsdaten,
$N2 = N1$, wenn Eintrittswahrscheinlichkeiten,
$N2 \geq N1$, wenn Übergangsraten eingegeben werden.

N3 wird nur dann $\neq$ blanc eingegeben, wenn die Abhängigkeit einer Komponente nur bei Berechnung von Übergangsraten berücksichtigt werden soll.

N4 Zahl der Zustände der Inhibitorfunktion abhängiger Komponenten.

B3) **(IDS(1,J),IDS(2,J),B(J),J=1,N2)** **Format(7(2I2,E10.4))**

IDS(1,J),IDS(2,J) Zustand (1) von dem und Zustand (2) zu dem Übergangsrate verläuft, oder blanc.

B(J) Wenn IDS(1,J) und IDS(2,J) $\neq$ blanc, B(J) = Übergangsrate $[1/t]$, wenn IDS(1,J) und IDS(2,J) = blanc, B(J) = Eintrittswahrscheinlichkeit des Zustandes J-1.

Für jede Komponente wird B2) und (wenn N2 > 0) B3) zusammen eingegeben. Für abhängige Komponenten muß B3) N4 mal eingegeben werden.

Nach der Eingabe der letzten Komponente wird B2) mit NI = 'END ' als Endmarkierung eingegeben.

B4) **NI,NTP,M,N,(INP(J),IST(J),NG(J),J=1,N)** **Format(2A4,2I1,8(A4,I2,A1))**

NI maximal 4 Zeichen umfassende Identifikation eines Gatters

NTP 3 Zeichen umfassende Bezeichnung des Typs des Gatters:

'AND ' Und-Verknüpfung der Vorgänger, $2 \geq N \geq 8$,

'OR ' Oder-Verknüpfung der Vorgänger, $2 \geq N \geq 8$,

'NOT ' Negation eines Vorgängers, N = 1,

'MAJ ' mindestens M aus N Vorgängern, $3 \geq N \geq 8$; $m < N$,

'EXC ' exakt M aus N Vorgängern, $3 \geq N \geq 8$; $m < N$,

'NMR ' nicht mehr als M aus N Vorgänger, $3 \geq N \geq 8$; $m < N$,

M Angabe für Systeme M aus N, (M < N)

N Zahl der unmittelbaren Vorgänger des Gatters,

INP(J) Identifikation des j-ten Vorgängers,

IST(J) Identifikation des Zustandes, wenn INP(J) die Identifikation einer Komponente, sonst blanc,

NG(J) blanc, im Programm MUSTAFA ohne Bedeutung.

B4) wird für jedes Gatter eines Fehlerbaums in beliebiger Reihenfolge eingegeben.

Nach Eingabe des letzten Gatters muß B4) mit NI = 'END ' als Endmarkierung eingegeben werden.

C1) **NST,NCMP** **Format(I4,4x,A4)**

NST Ordnungszahl des Zustandes der Inhibitorfunktion, (beginnend mit 1)

NCMP Identifikation der abhängigen Komponente, der die Inhibitorfunktion zugeordnet ist,

C2) **LG,(NO(L),NS(L),L=1,LG)** **Format(I4,4x,8(A4,I2,2x))**

LG Zahl der Variablen im eingegebenen Term,

NO(L),NS(L) Definition der L-ten Variablen im Term mit Identifikation einer Komponente und des Zustandes,

Nach dem letzten Term des mit NST und NCMP identifizierten Zustandes einer Inhibitorfunktion wird das Ende mit C2) LG = 0; NO(1) ='ENDF ' markiert.

D) **TRENN,(LISTC(J),J=1,8)** **Format(A8,8A4)**

TRENN ='- - - - - - - -' Markierung als Trennkarte (Abschluß vor er Eingabeende oder Eingabe des folgenden Fehlerbaums).

LISTC(J) Identifikation von Komponenten, die bei Auflistung der Funktion als Schlüsselkomponenten ausgeklammert werden sollen.

3.1.2 Anwendungsmöglichkeiten

Die verlangte Option wird mit den Parametern NOPT und NTR angegeben. Jede der folgenden Optionen bewirkt einen Durchlauf durch das Programm. Für einen weiteren Durchlauf muß die Eingabe von A1) bis D) wiederholt werden.

NOPT = 0, NTR = 0

Die Minimalschnitte als Ergebnis der Auflösung eines Fehlerbaums werden berechnet.

Wenn alle Komponentendaten vorhanden sind (Daten auf B3)), wird Eintrittswahrscheinlichkeit des Top als Summe der Wahrscheinlichkeiten der Minimalschnitte berechnet.

NOPT = 1, NTR = 0

Prüfung der Fehlerbaumfunktion auf Kohärenz, Berechnung der vollständigen Basisfunktion, gegebenenfalls der kleinsten irredundanten Basis, sonst wie NOPT = 0.

NOPT = 2, NTR = 0, Komponentendaten vollständig mit B3)

Berechnung wie NOPT = 1. Vor Berechnung der Eintrittswahrscheinlichkeit Top wird zur exakten Berechnung die Funktion erweitert mit dem Keystone-Algorithmus.

NOPT = 3, NTR = 0, Komponentendaten vollständig mit B3)

Berechnung wie NOPT = 1. Vor Berechnung der Eintrittswahrscheinlichkeit Top wird zur exakten Berechnung die Funktion erweitert mit dem Bennett-Algorithmus.

Die folgenden Optionen gelten für Fehlerbäume, die zu ein- und demselben System gehören und einen Systemzustand beschreiben. Gesucht werden die Übergangsraten. Wenn die Berechnung der Übergangsraten im selben Job erfolgt, können NUN(5) bis NUN(8) als temporäre Datensätze deklariert werden, andernfalls müssen eigene Datensätze benutzt, gegebenenfalls angelegt werden.

NOPT = 2, NTR = 1, NUN(5) > 0 Berechnung der Funktion eines Systemzustandes, Speicherung der Funktion auf die unter NUN(5) angegebene Nummer einer externen Speichereinheit.

NOPT = 2, NTR = 2, NUN(5) > 0, NUN(6) > 0, evtll. NUN(7),NUN(8) > 0 Berechnung der Funktion eines Systemzustandes. Speicherung der Funktion auf die unter NUN(5) angegebene Nummer einer externen Speichereinheit.

Berechnung von Übergangsraten zu den Systemzuständen, deren Funktion in einer mit NUN(J), J=6,8, NUN(J) > 0, angebenen Speichereinheit als Ergebnis eines vorangegangenen Programmdurchlaufs vorhanden ist. (Die betreffenden Nummern mußten in einem der vorangegangenen Durchläufe unter NUN(5) angegeben sein.)

NOPT = 2, NTR = 3, NUN(5) > 0, NUN(6) > 0 Berechnung der Übergangsrate zwischen den zwei Zuständen eines binären Systems. Hierbei muß der Fehlerbaum nur ein einziges Mal eingegeben werden, denn sein Komplement wurde bei Durchführung des Nelson-Algorithmus berechnet und wird bei dieser Option automatisch auf NUN(6) abgespeichert.

3.1.3 *Die Druckausgabe*

Die Druckausgabe des Programms MUSTAFA enthält Fehlermeldungen, Listen und das Ergebnis der Wahrscheinlichkeitsrechnung.

Vom Programm ausgegebene Fehlermeldungen beziehen sich auf die Eingabe und enthalten die Nachricht 'ERROR i'.

i	Beschreibung des Fehlers.
0	Karte mit IDCASE nicht vorhanden,
1	Zu viele Kommentarkarten,
2	Karte mit IDCASE nicht vorhanden,
4	Mehrfach verwendete Bezeichnung eines Gatters,
5	Fehler in einem Record B4), Typ eines Gatters, Parameter M, N
7,8	nicht identifizierte Komponente in einer Inhibitorfunktion,
9,10	nicht identifizierte Komponente oder deren Zustand im Laufe der Rechnung.
11	Fehler bei Eingabe der Übergangsraten oder der Eintrittswahrscheinlichkeit von Komponentenzuständen.

Wenn nicht durch KPR(J) besonders gesteuert, wird an Listen je nach Option ausgedruckt:

Liste	Inhalt
1	Wiedergabe der Eingaberecords,
2a,2b	Liste der Komponenten und Gatter des Fehlerbaums: 2a) mit Angabe der Vorgänger und Nachfolger, 2b) zusätzliche Liste zum Testen des Programms mit errechneten und eingesetzten Größen nach Aufspaltung der Mehrfachgatter zu Gattern mit nur je zwei Vorgängern.
3	Liste der Komponenten mit den berechneten Wahrscheinlichkeiten der Zustände,
4	Liste der Primimplikanten der vollständigen Basis,
5a,5b	a) Liste der Minimalschnitte, b) mit Erweiterung, wenn Bennett-Algorithmus verlangt wurde.

Im Anschluß an die Listen wird die Eintrittswahrscheinlichkeit des Top ausgedruckt.

Mit den Parametern KPR(1) bis KPR(4) können Änderungen des Ausdrucks von Listen vorgenommen werden:

KPR1	wenn KPR1 *lt.* 0 wird Liste 2a nicht ausgedruckt, wenn $KPR1 \neq 0$ wird Liste 2b nicht ausgedruckt

KPR2 KPR2 < 0, Listen 4a bzw. 4b, Listen 5a bzw. 5b,werden nicht ausgedruckt,

KPR3 0, Liste 3 wird nicht ausgedruckt,

KPR4 1, Liste der Komponenten mit Angabe der Nummern der Minimalschnitte, in denen sie enthalten sind, wird zusätzlich ausgedruckt,
2, Ausdruck der Übergangsfunktionen wird verlangt.

3.2 Das Programm MUSTAMO

3.2.1 Die Eingabe

Die Eingabe in das Programme MUSTAMO enthält die Fehlerbaumdaten (Eingabeblock B), daran anschließend Programmsteuerkarten (Eingabeblock E).

Die Eingabe der Fehlerbaumdaten ist mit der im Programm MUSTAFA compatibel, wenn die Fehlerbaumdaten keine als abhängig deklarierte Komponenten und keine Gatter vom Typ 'NOT ', 'EXC ' und 'NMR ' enthalten.

B1) IU1,IU2,IU3,IU4,IDCASE Format(4I2,A64)

IU1,IU2 Nummern von Speichereinheiten, die benötigt werden, wenn in den folgenden Programmsteuerkarten in mindestens einem Falle NOPT > 10 verlangt wird.

IU3 Nummer einer Speichereinheit, 0 < IU3 < 50 bewirkt Lesen der Fehlerbaumdaten von UNIT IU3 für einen Restart-Job, in dem die Eingabe der Fehlerbaumdaten entfällt, *IU3* ≥ 50 muß eingegeben werden, wenn in der folgenden Eingabe eine oder mehrere Komponenten durch Vorschaltung von NAMI = '*DET' als deterministisch deklariert sind.

IU4 Nummer einer Speichereinheit zur Datenspeicherung für späteren Restart.[4]

Zu beachten ist, daß IU3 > 0 und IU4 > 0 auch die Funktion von Steuerparametern übernehmen und anzeigen, ob Daten für einen Restart geschrieben werden sollen, ob gespeicherte Daten bei Restart gelesen werden sollen, oder ob. eine oder mehrere Komponenten im Eingabeblock zu deterministischen erklärt wurden.

IDCASE Kommentar von maximal 64 Zeichen Länge zur Fallidentifikation

---

[4] In der bei Control Data installierten Version entfällt IU4. Das Programm liest die Daten einer Vorrechnung, wenn IU1 > 0, und schreibt Daten für Restart, wenn IU2 > 0 eingegeben wird. IU3 > 0 dient als Steuerung einer Rechnung mit deterministischen Komponenten (wie oben mit IU3 > 50).

B2) **NI,N1,N2,N3,N4** **Format(A4,2I4,I3,I1)**

NI maximal 4 Zeichen umfassende Identifikation einer Komponente

N1 Zahl der Zustände der Komponente, $2 \leq N1 \leq 16$ (in MUSTAFA maximal 8) Die Zustände werden vom Programm automatisch numeriert von 0 bis N1-1.

N2 Zahl der in B3) einzugebenden Wahrscheinlichkeitsdaten,
$N2 = N1$, wenn Eintrittswahrscheinlichkeiten,
$N2 \geq N1$, wenn Übergangsraten eingegeben werden.

N3 ohne Bedeutung.

N4 Identifikation des Zustandes der Komponente, der im aktuellen Programmdurchlauf deterministisch = 1 gesetzt werden soll, sonst 0.

B3) **(IDS(1,J),IDS(2,J),B(J),J=1,N2)** **Format(7(2I2,E10.4))**

IDS(1,J),IDS(2,J) Zustand (1) von dem und Zustand (2) zu dem Übergangsrate verläuft, oder blanc.

B(J) Wenn IDS(1,J) und IDS(2,J) $\neq$ blanc, B(J) = Übergangsrate $[1/t]$, wenn IDS(1,J) und IDS(2,J) = blanc; B(J) = Eintrittswahrscheinlichkeit des Zustandes J-1.

B2) und **B3)** werden für jede folgende Komponente wiederholt,

Nach der letzten Komponente wird **B2)** mit NI = 'END ' als Endmarkierung eingegeben.

B4) **NI,NTP,M,N,(INP(J),IST(J),NG(J),J=1,N)**
Format(2A4,2I1,8(A4,I2,A1)

NI maximal 4 Zeichen umfassende Identifikation eines Gatters

NTP 3 Zeichen umfassende Bezeichnung des Typs des Gatters:

'AND ' Und-Verknüpfung der Vorgänger, $2 \geq N \geq 8$,

'OR ' Oder-Verknüpfung der Vorgänger, $2 \geq N \geq 8$,

'MAJ ' mindestens M aus N Vorgängern, $3 \geq N \geq 8$; $m < N$,

M Angabe für Systeme M aus N, (M < N)

N Zahl der unmittelbaren Vorgänger des Gatters,

INP(J) Identifikation des j-ten Vorgängers,

IST(J) Identifikation des Zustandes, wenn INP(J) die Identifikation einer Komponente, sonst blanc,

NG(J) Markierung der Negation des j-ten Vorgängers, wenn NG(J) $\neq$ blanc.

B4) wird für jedes Gatter eines Fehlerbaums in beliebiger Reihenfolge eingegeben.

Nach Eingabe des letzten Gatters muß B4) mit NI = 'END ' als Endmarkierung eingegeben werden. In diesem Record kann *NTP ≠ blanc* zur Outputsteuerung verwendet werden (siehe Abschnitt 3.2.3)

E1) **NPRF,NPRT** **Format(2I4)**

wenn NPRF > 0

E2) **NSCM,(NO(J),J+1,NSCM)** **Format(I4,6x,12(A4,1x))**

NPRF Zahl der folgenden Records E2),

NPRT Outputsteuerung (siehe Abschnitt 3.2.3)

NO(J) Identifikation von NSCM Gattern, ($1 \leq NSCM \leq 8$) deren (gemeinsame) logische Unabhängigkeit geprüft werden soll.

Diese Rechnung kann zur ersten Überprüfung umfangreicher Fehlerbaumdaten verwendet und bei weiteren Läufen übersprungen werden.

E3) **NOPT,NGT,NKE,NHALT,KF,(IGT(J),J=1,NGT)** **Format(5I2,4(A4,1x))**

NOPT Option, Erläuterungen siehe im folgenden Abschnitt,

NGT Zahl der Gatter IGT(J)

NKE Zahl der Schlüsselkomponenten in Record E4)

NHALT Zahl der Stop-Gatter in Record E5)

KF Steuerparameter zur Modulbildung bei vorläufigen Superkomponenten, siehe unter NOPT=4 in folgenden Abschnitt,

IGT(J) Identifikationen von Gattern an denen die in NOPT verlangte Operation durchgeführt werden soll.

E4) **(KEY(J),J=1,NKEY)** **Format(14(A4,1x))**

E5) **(NH(J),J=1,NHALT)** **Format(14(A4,1x))**

KEY(J) Identifikationen von Schlüsselkomponenten,

NH(J) Identifikationen von Stop-Gattern zur Beschleunigung der Auflösung des Baumes mit der Spitze IGT(J).

E4) und E5) entfallen, wenn NKEY bzw. NHALT = 0.

Die Eingabe E3) mit E4) und E5) kann solange für jeweils näher zur Spitze liegende Gatter wiederholt werden, bis mit NGT=1 und NGT(1) das Gatter an der Spitze des zu rechnenden Baumes eingegeben wird.

Es folgt entweder das Eingabeende oder ein Record E3) mit NOPT = 0 als Trennkarte zum folgenden zu rechnenden Fall.

3.2.2 *Arbeitsweise und Anwendungsmöglichkeiten*

Das Programm MUSTAMO errechnet an den in Record E3) angegebenen Knoten die Boole'sche Funktion, die je nach verlangter Option zu einem Modul zusammengefaßt oder in mehrere Module eingeteilt wird. Wird diese Maßnahme zuerst an Knoten in der Nähe der Basis des Baumes durchgeführt, so reduziert sich der Aufwand zur Auflösung von übergeordneten Knoten, die näher zur Spitze liegen bis hin zur Funktion der Spitze des Fehlerbaums. Aus diesem Grunde wird vom Programm eine in E3) verlangte Maßnahme zuerst ausgeführt und die Ergebnisse werden ausgegeben, ehe der folgende Record E3) eingelesen wird. Die Anordnung der Programmsteuerkarten E3) ff. muß so erfolgen, daß sukzessiv von unten nach oben Unterbäume abgeschnitten und durch Modulbezeichnungen ersetzt werden. Innerhalb der Unterbäume wird die Funktion mit dem Algorithmus von oben her errechnet. Das Programm MUSTAMO stellt somit eine Kombination der beiden Auflösungsalgorithmen (von unten her und von oben her) dar.

Zur Erläuterung der Bezeichnungen *Stop-Gatter* und *Schlüsselkomponenten* muß gesagt werden:

Wenn die vermaschte Struktur eines Fehlerbaums eine günstige Einteilung in abgetrennte Unterbäume nicht erlaubt, so daß die Auflösung des Baumes mit der Spitze an dem in E3) angegebenen Gatter Schwierigkeiten befürchten läßt, kann der Algorithmus mit Hilfe von *Stop-Gattern* etwas beschleunigt werden. Als Stop-Gatter eignen sich kritische Gatter mit mehr als einem Ausgang in Richtung zur Spitze. Der Algorithmus von oben her wird dann in zwei Stufen durchgeführt, wobei in der ersten Stufe von der Spitze bis zu den Stop-Gattern durch Anwendung des Idempotenz- und des Absorptionsgesetzes zahlreiche redundante Wege eliminiert werden können.

Schlüsselkomponenten werden Komponenten genannt, deren Variable bei der Bildung von Modulen ausgeklammert werden. Das Ausklammern von Variablen aus der Boole'schen Funktion kann

- notwendig sein, weil sie kritischen, die logische Abhängigkeit von Unterbäumen verursachenden, Komponenten angehören,

- frei gewählt sein, weil die Fehlerbaumfunktion nach bestimmten Gesichtspunkten aufgegliedert werden soll.

NOPT = 1 NGT = 1 NKE = 0 IGT(1) Abtrennung eines logisch unabhängigen Unterbaums. Das Gatter IGT(1) wird in der Folgerechnung wie eine binäre Variable behandelt.

NOPT = 2, NGT = 1, NKE ≠ 0 , IGT(1) , (KEY(J),J+1,NKE)

Abtrennung eines Unterbaums unterhalb des Gatters IGT(1). Die Funktion des Unterbaums wird durch Ausklammern der Schlüsselkomponenten KEY(J) gegliedert Module kombiniert mit Variablen der Schlüsselkomponenten. Das Gatter IGT(1) wird in der Folgerechnung wie eine binäre Variable behandelt.

Diese Option wird auch zur Berechnung an der Spitze des Fehlerbaums verwendet, wobei auch unkritische Komponenten als Schlüsselkomponenten zur Gliederung der Funktion eingesetzt werden können.

NOPT=3, NGT > 1, NKE = 0, (IGT(J),J+1,NGT)

Bildung einer Superkomponente mit 2^{NGT} Zuständen. Diese Option kann angewandt werden, wenn die Gatter IGT(J) lediglich untereinander aber nicht vom übrigen Baum logisch abhängig sind. In der Folgerechnung werden für die Gatter IGT(J) die entsprechenden Variablen der Superkomponente eingesetzt. In der Reihenfolge der Eingabe werden Superkomponenten vom Programm automatisch mit SC01, SC02 .... bezeichnet. Superkomponenten können mit diesen Bezeichnungen in folgenden Optionen als Schlüsselkomponenten angegeben werden.

NOPT=4, NGT > 1, NKE = 0, (IGT(J),J+1,NGT)

Bildung einer Superkomponente mit NGT Zuständen. Die Funktionen der Zustände der Superkomponente sind durch die unterhalb der Gatter IGT(J) befindlichen Bäume definiert und werden so vom Programm übernommen.

Diese Option mit den vom Benutzer selbst definierten Zuständen einer Superkomponente kann zur Vermeidung überflüssiger Rechenoperationen angewandt werden, wenn nicht alle 2^n möglichen Zustände einer aus n Gattern des ursprünglichen Fehlerbaums gebildeten Superkomponente existieren, oder wenn zwei oder mehr Zustände zu einem kondensiert werden können.

NOPT = 12 oder NOPT = 13 oder NOPT = 14, KF = 0 oder 1

Es besteht die Möglichkeit, die Optionen 2, 3 und 4 zur schnelleren Auflösung vermaschter Strukturen *vorläufig* einzusetzen. In diesem Falle können in den mit diesen Optionen abgetrennten Modulen auch logische Abhängigkeiten enthalten sein, denn zur Berechnung der Top-Funktion werden am Ende diese vorläufigen Module aufgelöst, d.h. mit ihrer expliziten Zusammensetzung in die Top-Funktion eingebracht, so daß Idempotenz erkannt und Absorption durchgeführt wird.

Wenn NOPT = 12 verlangt wurde, muß IGT(1), wenn NOPT = 13 bzw. 14, verlangt wurde muß 'SCnn' bei der Berechnung der Top-Funktion als Schlüsselkomponente (KEY(J)) angegeben werden. KF = 0 bewirkt die Auflösung der gesamten Funktion, KF = 1 bewirkt, daß die in den vorläufigen Superkomponenten enthaltenen Kombinationen von Schlüsselkomponenten und Modulen nicht aufgelöst werden.

NOPT = 10 oder NOPT = 20

Die Berechnung der Top-Funktion muß mit NOPT = 10 oder 20 aufgerufen werden, wenn in der vorangegangenen Rechnung mindestens eine vorläufige Superkomponente gebildet wurde. NOPT = 20 bewirkt, daß die zur Aufgliederung der Top-Funktion eingegebenen Schlüsselkomponenten schon vor Auflösung der vorläufigen Superkomponenten ausgeklammert werden, im Falle NOPT = 10 erfolgt die Aufschlüsselung erst nach der Auflösung der vorläufigen Superkomponenten.

Rechnung mit deterministischen Komponenten

Es gibt zwei Möglichkeiten Rechnungen mit deterministischen Komponenten durchzuführen:

- Wenn nur ein Durchlauf durch den Fehlerbaum erfolgen soll, wird im Eingaberekord **B2** der deterministischen Komponente NT < 0, N3 = deterministischer Zustand = 1 eingegeben, Eingaberecord **B4** entfällt für diese Komponente.

- Wenn mit allen Zuständen, bzw. allen möglichen Kombinationen von Zuständen Durchgänge durch den Fehlerbaum erfolgen sollen, wird **IU3 = 50** gesetzt, jede Komponente, deren Zustände hierbei abwechselnd 1 bzw. 0 gesetzt werden sollen benötigt 2 Eingaberecords **B2**, von denen der erste nur **NAMI = '*DET'**, der folgende die üblichen Daten für die Komponente enthält.

Im ersten Falle muß die Eintrittswahrscheinlichkeit des TOP in einer Handrechnung mit der Eintrittswahrscheinlichkeit des als deterministisch deklarierten Zustandes der Komponente multipliziert werden, im zweiten Falle übernimmt dies das Programm, das außerdem die Ergebnisse aller gerechneten zum Gesamtergebnis addiert.

3.2.3 Die Druckausgabe

Die Druckausgabe gliedert sich in:

1) Ausdruck der Fehlerbaumdaten in der Reihenfolge der Eingabe,

2) Ausdruck der Komponenten und Gatter mit zusätzlicher Angabe der Nachfolger,

3) Vom Programm geordnete Liste der Gatter,

4) Liste der Komponenten und Gatter, die logische Abhängigkeit verursachen,

5) Liste der Boole'schen Funktion und der Eintrittswahrscheinlichkeiten eines Gatters bzw. der Zustände einer Superkomponente.

Bemerkungen zu den einzelnen Gliedern der Druckausgabe:

zu 1) Dieser Teil enthält für die Komponenten nicht nur die eingegebenen Wahrscheinlichkeitsdaten, sondern auch, wenn Übergangsraten eingegeben wurden, die vom Programm errechneten Eintrittswahrscheinlichkeiten.

zu 2) Der Ausdruck dieser Liste kann mit *NTP* ≠ *blanc* auf der Endmarkierung nach Eingabe der Gatter unterdrückt werden.

zu 4) diese Liste entfällt, wenn NPRF = 0.

Die Abschnitte 1) bis 4) werden nur einmal ausgedruckt; sie werden bei wiederholtem Durchgang durch den Fehlerbaum mit abwechselnd deterministisch auf 1 gesetzten Komponentenzustände nicht wiederholt ausgedruckt.

zu 5) Nach jeder Eingabe von **E3, E4 ,E5** wird die verlangte Option ausgeführt und die Ergebnisse vor Einlesen der folgenden Option ausgedruckt. Bei Deklarierung einer Komponente als deterministisch werden in jedem erneuten Durchgang alle Optionen in der Reihenfolge der Eingabe durchgeführt.

3.3 Das Programm PASPI

Das Computerprogramm PASPI erleichtert die Anwendung der verschiedenen Optionen des Programms MUSTAMO, indem es einen Überblick über die Struktur eines Fehlerbaums verschafft.

Die Eingabe besteht aus dem Teil der Eingabe des Programms MUSTAMO, der die Fehlerbaumdaten enthält (Eingabeblock B). Deswegen kann PASPI auch zur Prüfung der Fehlerbaumdaten verwendet werden.

Die Ausgabe besteht aus

1. Ausdruck der Eingabe,

2. Liste der nach dem *Level* geordneten Fehlerbaumknoten,

3. Liste der kritischen Komponenten, der unabhängigen Module und der möglichen Superkomponenten mit mehr als zwei Zuständen.

Mit Level ist hier die Entfernung eines Knotens von der Spitze des Fehlerbaums definiert. Die Zahl Level n an einem Knoten bedeutet, daß der Weg vom Knoten bis zur Spitze des Fehlerbaums n-1 Knoten durchläuft. Die unter 3. angegebene Liste wird nach jedem von 3 Durchgängen durch den Algorithmus des Programms ausgedruckt. Sie enthält:

- eine Liste aller Knoten, deren Funktion als einfache Module vom Fehlerbaum abgetrennt werden können,

- eine Liste von Knoten, die gemeinsam als eine Superkomponente mit mehr als zwei Zuständen vom Fehlerbaum abgetrennt werden können.

Zu jeder möglichen Abtrennung werden folgende Informationen gegeben:

LEV Zahl, die den Level eines Knotens angibt,

RED Zahl, die die maximale Anzahl aller Wege von den Basisvariablen bis zu den Knoten angibt,

KEY(J) die innerhalb der abgetrennten Module und Superkomponenten befindlichen kritischen Knoten und Komponenten.

Das Programm gibt viele Möglichkeiten des horizontalen Schneidens an, aus denen der Benutzer eine Auswahl treffen muß.

Es ist übersichtlicher, wenn zu Superkomponenten eine möglichst kleine Zahl von Knoten, zudem vom gleichen Level vereinigt werden.

Die maximale Anzahl der Wege eines Moduls oder einer Superkomponente sollte nicht viel mehr als 10^6 betragen. Module oder Superkomponenten dieser Größenordnung können vom Programm MUSTAMO in einer Zeit < 30 Sekunden gelöst werden.

Im ersten Durchgang werden keine, oder nur die vom Benutzer in der Eingabe als deterministisch deklarierten, Schlüsselkomponenten angegeben. Die ausgedruckte Liste enthält nur logisch unabhängige Module und Superkomponenten, die in den folgenden Durchgängen als Basisvariable eingesetzt werden, wodurch die Zahl der Wege (RED) des verbleibenden Baumes reduziert wird.

Im zweiten Durchgang werden Module und Superkomponenten ausgedruckt, die nur unter Berücksichtigung der angegebenen Schlüsselkomponenten gebildet werden können.

3.4 Das Programm SIMUST

Zur Berechnung der Eintrittswahrscheinlichkeit des Top-Ereignisses eines Fehlerbaums werden die Wahrscheinlichkeiten der Komponentenzustände benötigt. In vielen Fällen können aber die Wahrscheinlichkeitsdaten der Komponenten nicht exakt, sondern nur innerhalb einer oberen und unteren Grenze angegeben werden. Um die daraus resultierende Streubreite der Wahrscheinlichkeit der Fehlerbaumrechnung zu ermitteln, könnte man die Fehlerbaumrechnung mit einem Analyseprogramm mehrmals mit variierten Eingabedaten der Komponenten durchführen. Dies ist aber sehr zeitaufwendig. Es liegt daher nahe, mit dem Ergebnis der Analyse in ein Anschlußprogramm zu gehen, in dem die Wahrscheinlichkeitsdaten der Komponenten an Hand einer Verteilungsfunktion variiert werden. Vorhandene Programme /19/ /18/ sind jedoch in ihrer Kapazität beschränkt und erlauben zudem nicht die Anwendung der in den Programmen MUSTAFA und MUSTAMO vorhandenen Möglichkeiten, Multi-State-Komponenten und statistische Abhängigkeiten zu berücksichtigen. Deshalb wurde das Programm SIMUST als Anschlußprogramm an die Fehlerbaumprogramme zur Simulation der Streuung der Wahrscheinlichkeitsdaten der Komponenten entwickelt.

3.4.1 Die Darstellung der Strukturfunktion

Um die Eintrittswahrscheinlichkeit des Top-Ereignisses eines Fehlerbaums in kurzer Zeit viele Male rechnen zu können, muß die Strukturfunktion in geeigneter Form vorliegen. Das Programm SIMUST verlangt als Eingabe einer Strukturfunktion eine Folge von Boole'schen Gleichungen. Die i-te Gleichung hat die Form

$$K_i = P_{i,1} \oplus P_{i,2} \oplus \ldots \oplus P_{i,m}$$

Als Verknüpfung $\oplus$ *kann* $\wedge$ *bzw.* $\vee$ eingesetzt werden.

Als Vorgänger $P_{i,j}$ können $K_{k<l}$ verwendet werden. Summen von Minimalschnitten können zu Modulen zusammengefaßt werden, wenn sie als Bezeichnung für K_i den gleichen Identifikationsnamen erhalten und mit einer Endmarkierung versehen sind. Die ersten Gleichungen $(i=1,2..)$ enthalten als Vorgänger Komponentenzustände, die letzte Gleichung K_n definiert die Spitze des Fehlerbaums.

Zur schnelleren Berechnung der Eintrittswahrscheinlichkeit des Top-Ereignisses geht das Programm durch den eingegebenen Fehlerbaum ohne Abfrage nach Idempotenz und nach etwa zu absorbierenden Termen. Daher darf die aus der eingegebene Folge von Gleichungen errechnete

Funktion keine logischen Abhängigkeiten enthalten, so daß die für jede Gleichung errechnete Eintrittswahrscheinlichkeit in nachfolgende Gleichungen eingesetzt werden kann.

Es gibt drei Eingabemöglichkeiten:

- Direkte Eingabe der Gleichungen der Knoten des Fehlerbaums, wenn dieser einfach ist und keine logischen Abhängigkeiten enthält.
- Eingabe von Modulen und ihrer Verknüpfung nach Maßgabe eines Blockdiagramms.
- Eingabe einer Summe von Minimalschnitten, deklariert als Modul.

3.4.2 *Simulation und Berechnung der Streuung*

Die Streuung von Zahlenwerten < 1 folgt in der Regel der Lognormalverteilung. Das Programm SIMUST simuliert die Lognormalverteilung mit Hilfe von Zufallszahlen. Die Lognormalverteilung hat die Form:

$$y = \frac{1}{\sqrt{2\pi v^2}} \cdot \frac{1}{x} \cdot e^{\frac{(\ln x - \mu)^2}{2 v^2}}$$

Der Streufaktor SF ist der Quotient aus $x_1/Median = Median/x_2$. Als Vertrauensgrenzen x_1 *und* x_2 werden im normierten Integral der Verteilungsfunktion 0,05 und 0,95 angesetzt. Daraus errechnet sich die Breite der Verteilung $v = \frac{\ln(SF)}{1{,}645}$

Der arithmetische Mittelwert $\bar{x}$ und der Median e^{μ} der Verteilung fallen nicht wie bei der Normalverteilung zusammen. Zwischen dem Mittelwert $\bar{x}$ und der Varianz σ^2 und den Parametern der Lognormalverteilung μ, dem Logarithmus der Median, und v^2 bestehen die Beziehungen:

$$\mu = \ln(\bar{x}) - \frac{v^2}{2} \quad ; \quad \sigma^2 = \bar{x}^2 (e^{v^2} - 1) \quad \textit{und daraus} \quad v^2 = \ln \left[\frac{\sigma^2}{\bar{x}^2} + 1 \right]$$

Diese Beziehungen liegen der Berechnung der Streuung der Eintrittswahrscheinlichkeit des Top-Ereignisses zu Grunde. Aus den Ergebnissen von n Durchläufen werden der Mittelwert und die Varianz errechnet; der Größe nach geordnet liefern sie den Median und die 5%- bzw. die 95%-Grenze. Nimmt man an, daß die Ergebnisse annähernd lognormal-verteilt sind, so kann Streufaktor und Median auf verschiedene Weise aus den oben erwähnten Beziehungen ermittelt werden. Die vom Programm ausgedruckte Ergebnisliste enthält daher mehrere Zahlenwerte für den Median, den Parameter v der Lognormalverteilung und für den Streufaktor. Der Begleittext gibt an, auf welchem Wege der Wert errechnet wurde.

Das Programm enthält nach neuestem Stand ein Unterprogramm, das eine Graphik erzeugt, in der die Häufigkeitsverteilung über der Wahrscheinlichkeit aufgetragen ist. Hierzu wurden die Logarithmen der Abszissenwerte in äquidistante Intervalle unterteilt. Die Verteilung erscheint daher angenähert als Normalverteilung. Die so in diskreten Wertepaaren vorliegende Funktion ermöglicht die Berechnung der Parameter einer Normalverteilung als Anpassungsfunktion mit der

Methode der kleinsten Fehlerquadrate. Aus diesen Parametern können die Parameter der gesuchten Lognormalverteilung errechnet werden.

Zur Simulation der Streuung der Wahrscheinlichkeiten der Komponentenzustände prüft das Programm folgende in der Eingabe enthaltene Informationen:

- Sind die eingegebenen Zahlenwerte der Wahrscheinlichkeitsdaten aller Komponenten als arithmetischer Mittelwert oder als Median der Verteilung aufzufassen?
- Liegen die Wahrscheinlichkeitsdaten einer Komponente als Übergangsraten oder als Eintrittswahrscheinlichkeiten der Zustände vor?
- Ist der zusammen mit der Eingabe der einzelnen Übergangsrate bzw. Wahrscheinlichkeit verlangte Streufaktor > 1.

Diejenigen Größen, deren zugeordneter Streufaktor mit ≤ 1 eingegeben wurde, bleiben bei jeder Wiederholung konstant. Die Berechnung der Wahrscheinlichkeiten aus den Übergangsraten wurde in Abschnitt 2.1 behandelt. Die Lognormalverteilung wird mit:

$$x = \exp\left[\mu + \left(\sum_{i=1}^{12} R_i - 6\right) \cdot v\right]$$

berechnet. R_i sind gleichverteilte Zufallszahlen. Die Verteilung der Summen von je 12 Zufallszahlennähert sich bei häufiger Ziehung der Normalverteilung. /20/.

3.4.3 Die Eingabe

Für einen zu rechnenden Fall ist einzugeben:

Einmal	F1) Fallidentifikation, F2) allgemeine Eingabeparameter
Für jede Komponente	F3) Definition der Komponente, F4) Wahrscheinlichkeitsdaten,
Abschluß	Endmarkierung vom Typ F3)
Für jede Gleichung der Strukturfunktion	F5) Definition einer Gleichung
Abschluß	Endmarkierung vom Typ F5)

F1) **ID** **Format(A64)**

ID maximal 64 Zeichen umfassender Kommentar zur Identifikation

F2) **NTR,MITT,NGRAPH,RIN Format(I8,A8,I2,2x,F10.4)**

NTR Anzahl der Simulationsversuche, default option (wenn NTR=0) = 4000

MITT — Wenn MITT = 'MEDIAN ' eingegeben wird, werden die Wahrscheinlichkeitsdaten der Komponenten als Median der Verteilung angenommen, sonst als arithmetische Mittelwerte.

NGRAPH — Steuergröße, kein Aufruf des Unterprogramms GRAPH, wenn NGRAPH > 0,

RIN — Initialisierung des Zufallszahlengenerators, ($0 < RIN < 100$) (kann verwendet werden, um bei Parametervariationen identische Zufallszahlensequenzen zu erzeugen, wenn RIN = 0 eingegeben wird, initialisiert das Programm zu Beginn und setzt die Sequenz mit jedem neuen Fall fort.)

F3) **NAME,NAM1,NSTT,NL,INH** **Format(2A4,3I2)**

NAME — Vier Zeichen umfassende Identifikation einer Komponente

NAM1 — in der Regel blanc, Ausnahme siehe unter Erläuterungen 1)

NSTT — Anzahl der Zustände der Komponente

NL — Zahl der in F4) eingegebenen Wahrscheinlichkeitsdaten

INH — in der Regel 0, Ausnahmen siehe unter Erläuterungen 2)

Die Zustände einer Komponente werden vom Programm mit 0, 1, ..., NSTT-1 bezeichnet, wobei 0 den Intaktzustand indiziert.

F4) **(N1(L),N2(L),AI(L),S(L),L=1,NL)** **Format(4(2A2,E9.3,F5.2))**

N1(L),N2(L) — Identifikation eines Zustandes

Wenn N1(L) und N2(L) $\neq$ *blanc*, dann ist AI(L) Übergangsrate von Zustand N1(L) zu N2(L)
Wenn N2(L) = blanc, dann ist AI(L) Eintrittswahrscheinlichkeit des unter N1(L) angegebenen Zustands

AI(L) — Übergangsrate $[\frac{1}{t}]$ bzw. Eintrittswahrscheinlichkeit [*o.D.*]

S(L) — Streufaktor, AI(L) wird variiert,wenn S(L) > 1.

Endmarke F3) mit NAME='END ' nach F4) der letzten eingegebenen Komponente

F5) **NAME,NTYP,LG,(INP(L),NS(L),NO(L),L+1,LG)**
Format(2A4,I2,8(A4,A2,I1))

NAME — Identifikation eines Terms

NTYP — Verknüpfung der Vorgänger. Erlaubt ist 'AND ','OR ' und 'MODL' Wird die Bezeichnung 'MODL' verwendet, dann wird diese Eingabe von F5) als Endmarkierung mehrerer mit der gleichen Identifikation NAME unmittelbar vorher eingegebener Terme aufgefaßt. LG ist in diesem Falle= 0.

LG	Zahl der im Term verknüpften Vorgänger INP(L)
INP(L)	Identifikation eines Vorgängers, entweder Komponente oder Identifikation eines davor eingegebenen Terms
NS(L)	Identifikation des Zustandes, nur bei Komponenten anzugeben.
NO(L)	Identifikation des Zustandes einer Inhibitorfunktion, siehe Erläuterungen 2).

Endmarke F5) mit NAME = 'END ' nach Eingabe von F5) der Spitze des Fehlerbaums.

Erläuterungen 1)

Für Komponenten gleichen Typs müssen die Wahrscheinlichleitsdaten nicht wiederholt eingegeben werden. NAM1 benennt die zuvor eingegebene Komponente, deren Wahrscheinlichkeitsdaten übernommen werden können. Die Eingabe F4) entfällt, wenn NAM1 $\neq$ blanc. Wenn in diesem Falle NL $>$ 0 eingegeben wird, wird für die Komponenente NAME eine wiederholte Simulation mit den Daten der Komponente NAM1 durchgeführt, wenn NL $=$ 0 eingegeben wird, wird das Ergebnis der Ziehung von NAM1 übernommen.

Erläuterungen 2)

Sind im Programm MUSTAFA Komponenten verwendet worden, deren statistische Abhängigkeiten mit Inhibitorfunktionen definiert wurden, dann wird mit INH in F3) die Zahl der Zustände der Inhibitorfunktion angegeben. Die Wahrscheinlichkeitsdaten einer solchen Komponente müssen durch wiederholte Eingabe von F4) INH-mal eingegeben werden und in F5) muß angegeben werden, welcher Zustand der Inhibitorfunktion identifiziert wurde, so daß in der Wahrscheinlichkeitsrechnung die korrekte Wahrscheinlichkeit eingesetzt werden kann. Die letztere Information kann der in MUSTAFA ausgedruckten Liste der Minimalschnitte entnommen werden.

Schlußbemerkungen

Die in diesem Bericht vorgestellten Methoden und Algorithmen und ihre Anwendung in Rechenprogrammen erlauben es, Fehlerbäume vollständig zu analysieren, auch wenn die von den Fehlerbäumen dargestellten Systeme sehr groß und stark vermascht sind.
Schon die Menge der Minimalschnitte eines mittleren Fehlerbaums kann so groß sein, daß sie nicht mehr zusammenfassend überschaubar ist. Die Zahl der Auflösungswege kann wegen starker Vermaschung noch einmal um Größenordnungen höher sein, so daß die Analyse wegen zu hoher Rechenzeiten zu scheitern droht.
Beide Nachteile kann das Programm MUSTAMO mit verschiedenen Zerlegungsmethoden ausgleichen. Als Ergebnis der Rechnung liegt die Boolesche Strukturfunktion in vollständiger und kompakter Form vor, denn aus der Zerlegung ergeben sich unabhängige Module, deren Minimalschnitte überschaubar sind, und die sich zu einem Blockdiagramm verknüpfen lassen.

Diejenigen Module, die im Gesamtergebnis in's Gewicht fallen, können mit den im Programm MUSTAFA gebotenen Möglichkeiten im Einzelnen analysiert oder mit dem Programm SIMUST auf die Streuung der Wahrscheinlichkeit hin untersucht werden.

Als vielseitig verwendbares Hilfsmittel zur Durchführung der dargestellten Methoden zeigt sich die Möglichkeit, Boole'sche Variable mit Einschränkungen (BVME) zu bilden. Mit Hilfe der BVME kann

- eine Komponente mit mehr als 2 Zuständen exakt dargestellt,
- statistische Abhängigkeit zwischen Komponenten modelliert,
- die Möglichkeit der Zerlegung eines Fehlerbaums in unabhängige Module erheblich erweitert werden.

Die Bildung der BVME sowie der im zweiten Teil dieses Berichtes vorgestellte, bisher noch nicht veröffentlichte, Algorithmus zur Suche nach der kleinsten irredundanten Basis einer inkohärenten Boole'schen Funktion berühren allgemeine, theoretische Fragen der Boole'schen Algebra. Beide Probleme könnten für die Theorie und für andere Anwendungsgebiete der Boole'schen Algebra von Interesse sein.

Literaturverzeichnis

/1/ L. Caldarola "Grundlagen der Boole'schen Algebra mit beschränkten Variablen", KfK 2915, EUR 6405d, Febr. 1980

/2/ L. Caldarola, A. Wickenhäuser "Recent advancements in Fault Tree Methodology at Karlsruhe", International conference on Nuclear Systems Reliability, Engineering and Risk Assessment, Gatlinburg, SIAM, 518 - 542, June 1977

/3/ L. Caldarola, "Fault Tree Analysis of Multistate Systems with Multistate Components", ANS topical meeting on probabilistic analysis of nucl. reactor safety, Los Angeles, Calif. Paper VIII.1, May 1978

/4/ L. Caldarola, "Fault Tree Analysis with Multistate Components". KfK 2761, EUR 5756e, Febr. 1979

/5/ L. Caldarola, "Coherent Systems with Multistate Components". Nuclear Engineering and Design 58 (1980) pp. 127-139

/6/ L. Caldarola, "Generalized Fault Tree Analysis combined with State Analysis". KfK 2530, EUR 5754e, Febr. 1980

/7/ L. Caldarola, A. Wickenhäuser, "The Boolean Algebra with Restricted Variables as a Tool for Fault Tree Modularization", KfK 3190, EUR 7056e, August 1981

/8/ J.B. Fussell u.a., "MOCUS - A computer program to obtain minimal cut set from fault trees", Aerojet Nucl. Company, March 1974

/9/ B.L. Hulme and R.B.Worrell, "A Prime Implicant Algorithm with Factoring", IEEE Transactions on Computers, Nov. 1975, pp. 1129-1131

/10/ H. Nakazawa, "A decomposition method for computing systems reliability by a Boolean expression", IEEE Transactions on Reliability, Vol. R-26, No. 4, Oct. 1977, 250-252

/11/ R.G. Bennett, "On the analysis of fault trees". IEEE Transactions on Reliability, Vol. R-24, No. 3, Aug. 1975

/12/ G. Weber, unveröffentlichter Bericht, 1981

/13/ L. Caldarola, H. Schnauder, A. Wickenhäuser, unveröffentlichter Bericht, 1982

/14/ L. Caldarola,H. Schnauder, A. Wickenhäuser, E. Pamfilie, "Fault Tree Analysis of a Reactor Protective System". International ANS/ENS Topical Meeting on Probabilistic Risk Assessment, New York, Sept. 20-24, 1981.

/15/ A. Wickenhäuser, "MUSTAFA und MUSTAMO, Rechenprogramme zur Analyse von Fehlerbäumen mit Multistate-Komponenten". KfK 3855, Febr. 1985

/16/ L. Caldarola, unveröffentlichter Bericht, 1985

/17/ L. Caldarola, unveröffentlichter Bericht, 1985

/18/ Y.T. Lee, G.E. Apostolakis, "Probability Intervals for the Top Event of Fault Trees". UCLA-ENG-7663, June 1976

/19/ Y.T. Lee, S.T. Salem, "Probability Intervals for the Reliability of complex Systems using Monte Carlo Simulation". UCLA-ENG-7758, Dec. 1977

/20/ Naylor et al. "Computer Simulation Techniques", John Wiley, N.Y. 1965.

Erläuterungen zu den Beispielrechnungen

Rechnungen zur Suche nach der kleinsten irredundanten Basis

Der Algorithmus zur Suche nach der kleinsten irredundanten Basis kann auch außerhalb des Programms MUSTAFA mit einem eigenen Rechenprogramm durchgeführt werden, in dem zu Test- und Demonstrationszwecken die Druckausgabe erheblich erweitert wurde.

I,1 - I,2 — Beispiel mit BVME. Die Buchstaben der Variablen A,B,C,D stehen für die natürlichen Zahlen 1,2,3,4. Die Ziffer gibt an, wie oft die Zahl als Summand in einer Summe enthalten ist oder mit welchem Faktor sie multipliziert wird. Die eingegebene Funktion besteht aus allen Kombinationen, die die Summe 5 ergeben. Mit dem Rechenprogramm wurde das Komplement dieser Funktion errechnet. Diese in der Computerliste F genannte Funktion enthält als Minimalschnitte bzw. Primimplikanten diejenigen Kombinationen von Variablen, die nicht die Summe 5 ergeben oder zur Summe 5 ergänzt werden können. Aus der Gesamtzahl dieser Primimplikanten errechnet das Programm die kleinste irredundante Basis.

I,3 - I,6 — Beispiel mit 9 binären Variablen und 81 Primimplikanten

Rechnungen mit den Fehlerbaumprogrammen

Zur Demonstration des Zusammenwirkens der vier Fehlerbaumprogramme wird in allen Rechnungen ein unter der Bezeichnung *Fehlerbaum Nr.2* hier vorhandenes System ganz oder teilweise behandelt.

II,1 — Abbildung des Fehlerbaums.

III,1 — Die Rechnung mit dem Programm PASPI zeigt mögliche Schnittstellen. Die auch in der graphischen Darstellung deutlich sich anbietenden Gatter '10X', '10Y', '10Z' zur Bildung der ersten Superkomponente SC01 und die Gatter '06', '07' zur Bildung der zweiten Superkomponente SC02 erscheinen hier deutlich mit dem gemeinsamen Level 5 und Level 7.

III,2 - III,9 — Mit dem Programm MUSTAMO wurde der vollständige Fehlerbaum gerechnet, die Top-Funktion wurde mit den Komponenten C030 und SC02 zu Reihen eines Blockdiagramms aufgeschlüsselt. Es wird hier die vollständige Computerliste mit den Fehlerbaumdaten und den Ergebnissen für SC01 vorgestellt.

III,10 — Block-Diagramm der Top-Funktion.

Mit dem Programm MUSTAFA wurden mehrere Zustände der in MUSTAMO mit SC01 bezeichneten Superkomponente nachgerechnet. Die Liste 2 wurde nur in der ersten Rechnung ausgedruckt, in den folgenden Rechnungen unterdrückt.

IV,1 - IV,3 Die Rechnung enthält die Zusammenfassung der in MUSTAMO unter der Bezeichnung SC01 3, SC01 5, SC01 6 und SC01 7 berechneten Module. Der hierfür konstruierte Fehlerbaum hat als Spitze ein Gatter vom Typ 'MAJ' (zwei aus drei), gebildet aus den Gattern '10X','10Y','10Z'. Die Berechnung der Eintrittswahrscheinlichkeit wurde mit dem *Keystone-Algorithmus* durchgeführt. Der errechnete Zahlenwert ist daher etwas kleiner als die Summe der Wahrscheinlichkeiten der in MUSTAMO errechneten Module.

IV,4 - IV,7 Nachrechnung der Module SC01 1 und SC01 2. Hierzu wurden zwei Fehlerbäume gebildet mit der Spitze $10X \wedge \overline{10Y} \wedge \overline{10Z}$ und $\overline{10X} \wedge 10Y \wedge \overline{10Z}$ Zu beobachten ist, daß die Negation dieser Fehlerbäume sehr viel weniger Auflösungswege zu verzeichnen haben, so daß das Programm den negierten Fehlerbaum mit dem Up-down-Algorithmus aufgelöst und die Funktion des Top-Elementes durch Negation der Liste der Minimalschnitte der negierten Funktion errechnet hat. Die exakten Eintrittswahrscheinlichkeiten wurden mit dem *Keystone-Algorithmus* berechnet.

IV,8 - IV,9 Nachrechnung des Moduls SC01 4 durch Bildung eines Fehlerbaums mit der Spitze $10X \wedge \overline{10Y} \wedge 10z$ Die exakte Eintrittswahrscheinlichkeit wurde zum Methodenvergleich mit dem *Bennett-Algorithmus* berechnet.

Für die Module SC01 2 und SC01 4 berechnen die verschiedenen Algorithmen den gleichen Zahlenwert, da die Struktur der Module und die Wahrscheinlichkeitsdaten der Komponenten identisch sind.

V,1 - V,3 Zur Simulation der Streuung der Wahrscheinlichkeitsdaten mit dem Programm SIMUST wurden die für das Ergebnis der Wahrscheinlichkeit des Top maßgebenden Reihen 3, 4, und 5 der MUSTAMO-Rechnung herangezogen.

DAS KOMPLEMENT EINER FUNKTION F BESTEHEND AUS ALLEN KOMBINATIONEN
EINSCHLIESSLICH VON PRODUKTSUMMEN DER NATUERLICHEN ZAHLEN 1 BIS 4,
AUS DENEN DIE SUMME 5 GEBILDET WERDEN KANN WIRD EINGEGEBEN.
ERRECHNET WERDEN SOLL DIE KLEINSTE BASIS DER FUNKTION F.

****
* DARSTELLUNG DER PRODUKTSUMMEN MIT VARIABLEN M. EINSCHR. *
****

VARIABLE DER FUNKTION F

A 1	A 2	A 3	A 0
B 1	B 2	B 0	
C 1	C 0		
D 1	D 0		

TERME DER KOMPLEMENTFUNKTION

1)	A 3	B 1	C 0	D 0
2)	A 2	B 0	C 1	D 0
3)	A 1	B 0	C 0	D 1
4)	A 1	B 2	C 0	D 0
5)	A 0	B 1	C 1	D 0

PRIMIMPLIKANTEN P(I) DER VOLLSTAEND.BASIS DER FUNKTION F

I				
1)	A 1	B 1		
2)	A 1	C 1		
3)	A 2	B 1		
4)	A 2	B 2		
5)	A 2	C 0		
6)	A 2	D 1		
7)	A 3	B 2		
8)	A 3	B 0		
9)	A 3	C 1		
10)	A 3	D 1		
11)	A 0	B 2		
12)	A 0	B 0		
13)	A 0	C 0		
14)	A 0	D 1		
15)	B 1	D 1		
16)	B 2	C 1		
17)	B 2	D 1		
18)	C 1	D 1		
19)	A 1	B 0	D 0	
20)	B 0	C 0	D 0	

```
KOMBINATIONEN DER MIT ALPHA(I) BEZEICHNETEN PRIMIMPLIKANTEN
  18    15    14
  15    10
  18     6
  20     5
  16     2
  18     2
  20    19
  20     8
  17
  16    11
  16     4
  13
  12
   9
  10     8
   7
   6     5
   5     4
   3
  19     2
   1

KLEINSTE IRREDUNDANTE BASIS

   1     3     5     7     8     9    10    12    13    16    17    18
  19

TEST 1:KOMPLEMENT IDENTISCH MIT KOMPLEMENT DER FUNKTION  F?  JA
TEST 2: PRIMIMPLIKANTEN DER ERRECHNETEN BASIS  13
        OBLIGATORISCH FUER DIESE BASIS         13
ZEIT(SEC)=  0.21
```

```
**********************************************************************
* IEEE TRANSACT.ON COMP.,NOV.75,S.1131, EXAMPLE 7                    *
**********************************************************************
```

VARIABLE DER FUNKTION F

A 1	A 0
B 1	B 0
C 1	C 0
D 1	D 0
E 1	E 0
F 1	F 0
G 1	G 0
H 1	H 0
I 1	I 0

EINGEGEBENE FUNKTION F

1)	A 1	B 1	C 0	I 1
2)	A 1	C 1	F 0	H 1
3)	A 1	C 0	D 1	F 1
4)	A 1	F 1	G 1	I 1
5)	A 1	F 1	G 0	H 1
6)	A 1	G 1	H 0	I 0
7)	A 0	C 0	F 1	I 1
8)	A 0	D 0	E 0	G 1
9)	B 1	C 1	F 1	I 1
10)	B 1	D 1	E 0	F 1
11)	B 1	D 1	G 1	I 0
12)	B 1	D 0	E 1	G 1
13)	B 0	C 0	F 1	G 0
14)	B 0	E 1	G 1	H 1
15)	C 1	D 1	G 1	H 1
16)	C 1	E 1	G 1	I 1
17)	C 1	F 0	H 0	I 1
18)	C 0	D 0	G 0	H 0
19)	C 0	G 0	H 0	I 0
20)	F 0	G 1	H 0	I 1

TERME DER KOMPLEMENTFUNKTION

1)	A 0	F 0	G 0	H 1		
2)	A 0	C 1	D 0	G 0	I 0	
3)	A 0	C 1	E 1	G 0	I 0	
4)	A 0	B 0	C 1	F 1	G 0	
5)	A 0	B 0	C 1	G 0	H 1	
6)	A 0	B 0	C 1	G 0	I 0	
7)	A 0	C 1	F 0	G 0	I 0	
8)	B 0	C 1	F 1	G 0	H 0	
9)	B 0	C 1	G 0	H 0	I 0	
10)	B 0	C 0	F 0	G 0	H 1	
11)	C 1	E 1	G 0	H 0	I 0	
12)	C 1	D 0	G 0	H 0	I 0	
13)	C 1	F 0	G 0	H 0	I 0	
14)	C 0	F 0	G 0	H 1	I 0	
15)	A 1	B 0	C 0	E 0	F 0	H 1
16)	A 0	B 0	D 1	G 1	H 0	I 0
17)	A 0	B 0	E 1	G 1	H 0	I 0
18)	A 0	B 0	C 1	D 1	H 0	I 0

19)	A 0	B 0	C 1	E 1	H 0	I 0	
20)	A 0	B 1	D 0	G 0	H 1	I 0	
21)	A 0	B 1	E 1	G 0	H 1	I 0	
22)	A 0	C 0	D 1	F 0	G 0	I 1	
23)	B 0	C 0	D 1	E 0	F 0	H 1	
24)	B 0	C 0	D 1	F 0	G 0	I 1	
25)	A 1	C 0	D 0	E 0	G 1	H 1	I 0
26)	A 1	D 0	E 0	F 1	G 1	H 1	I 0
27)	A 1	C 0	D 0	E 0	F 0	H 1	I 0
28)	A 0	B 0	C 0	D 1	E 0	G 1	I 0
29)	A 0	B 0	C 1	D 1	E 0	F 1	H 0
30)	A 0	B 1	C 0	D 1	F 0	H 1	I 1
31)	A 0	C 0	D 1	E 0	F 0	H 1	I 1

PRIMIMPLIKANTEN P(I) DER VOLLSTAEND.BASIS DER FUNKTION F

I							
1)	A 1	B 1	I 1				
2)	A 1	E 1	G 1				
3)	A 1	G 1	H 0				
4)	B 1	F 1	I 1				
5)	B 1	G 1	H 0				
6)	C 0	F 1	I 1				
7)	A 1	C 1	F 0	I 1			
8)	A 1	B 1	C 0	H 0			
9)	A 1	C 0	D 0	H 0			
10)	A 1	C 0	H 0	I 0			
11)	A 1	D 1	F 1	H 1			
12)	A 1	E 1	F 1	H 1			
13)	A 1	F 1	G 0	H 1			
14)	A 1	F 1	H 1	I 1			
15)	A 1	C 0	D 1	F 1			
16)	A 1	C 0	E 1	F 1			
17)	A 1	C 0	F 1	G 0			
18)	A 1	C 0	F 1	H 0			
19)	A 1	D 1	F 1	G 1			
20)	A 1	C 1	D 1	H 1			
21)	A 1	C 1	E 1	H 1			
22)	A 1	C 1	F 0	H 1			
23)	A 1	C 1	G 0	H 1			
24)	A 1	C 1	H 1	I 1			
25)	A 1	C 1	D 1	G 1			
26)	A 1	C 1	F 0	G 1			
27)	A 1	C 1	G 1	I 1			
28)	A 1	F 1	G 1	I 1			
29)	A 1	B 1	D 1	G 1			
30)	A 0	D 0	E 0	G 1			
31)	A 0	D 0	G 1	H 1			
32)	A 0	D 0	G 1	I 1			
33)	A 0	C 1	G 1	H 1			
34)	A 0	B 1	C 1	G 1			
35)	A 0	B 1	D 0	G 1			
36)	A 0	B 1	F 1	G 1			
37)	A 0	B 1	G 1	I 0			
38)	B 1	D 1	E 0	F 1			
39)	B 1	C 1	H 0	I 1			
40)	B 1	D 0	H 0	I 1			

41)	B 1	C 0	D 0	H 0		
42)	B 1	C 0	F 1	H 0		
43)	B 1	C 0	H 0	I 0		
44)	B 1	C 1	F 0	G 1		
45)	B 1	C 1	D 1	G 1		
46)	B 1	D 1	F 1	G 1		
47)	B 1	D 1	G 1	I 0		
48)	B 1	C 1	E 1	G 1		
49)	B 1	D 0	E 1	G 1		
50)	B 1	E 1	F 1	G 1		
51)	B 1	E 1	G 1	I 0		
52)	B 1	C 1	G 1	I 1		
53)	B 1	D 0	G 1	I 1		
54)	B 0	C 0	F 1	G 0		
55)	B 0	E 1	G 1	H 1		
56)	B 0	E 1	G 1	I 1		
57)	C 1	F 0	H 0	I 1		
58)	C 1	D 1	G 1	H 1		
59)	C 1	F 0	G 1	H 1		
60)	C 1	D 0	G 1	I 1		
61)	C 1	F 0	G 1	I 1		
62)	C 1	G 1	H 1	I 1		
63)	C 1	E 1	G 1	H 1		
64)	C 1	E 1	G 1	I 1		
65)	C 0	D 0	E 0	H 0		
66)	C 0	D 0	G 0	H 0		
67)	C 0	F 1	G 0	H 0		
68)	C 0	G 0	H 0	I 0		
69)	C 0	D 0	H 0	I 1		
70)	C 0	G 1	H 0	I 1		
71)	D 0	E 0	G 1	H 0		
72)	D 0	F 0	H 0	I 1		
73)	D 0	F 1	G 1	I 1		
74)	D 0	G 1	H 0	I 1		
75)	D 0	E 1	G 1	H 1		
76)	D 0	E 1	G 1	I 1		
77)	E 1	F 1	G 1	H 1		
78)	E 1	G 1	H 1	I 0		
79)	E 1	F 1	G 1	I 1		
80)	E 1	G 1	H 0	I 1		
81)	F 1	G 1	H 1	I 1		
82)	F 0	G 1	H 0	I 1		
83)	B 0	C 0	E 1	F 1	H 1	
84)	C 1	D 0	E 0	F 0	G 1	
85)	C 0	D 1	E 0	F 1	G 0	
86)	A 0	B 0	C 0	D 0	E 0	F 1
87)	A 0	B 0	C 0	D 0	F 1	H 1

KOMBINATIONEN DER MIT ALPHA(I) BEZEICHNETEN PRIMIMPLIKANTEN

```
 81   73   62   60   28   27   24   14
 82   70
 78   75   51   49   37   35   31
 71    3
 81    6
 80   79   64   56
 78   55
 78   75   51   49    2
 72   69   66
 76   75   53   49   35   32   31
 71   30
 58   25   20   19   11
 68
 85   54
 67   54    6
 70    6
 57
 58   33
 59   58   26   25   22   20
 84   59   26   22
 56   55
 83   54
 87   86   54
 51   50   49   48   37   36   35   34    5
 51   49   48   44   37   35   34    5
 51   49   43   41   37   35    5
 47   29
 47   37
 38
  4
 31   30
  3    2
 25   19    3
 26   25    3
 10    3
 17   13
 23   13
 19   15   11
 23   22
```

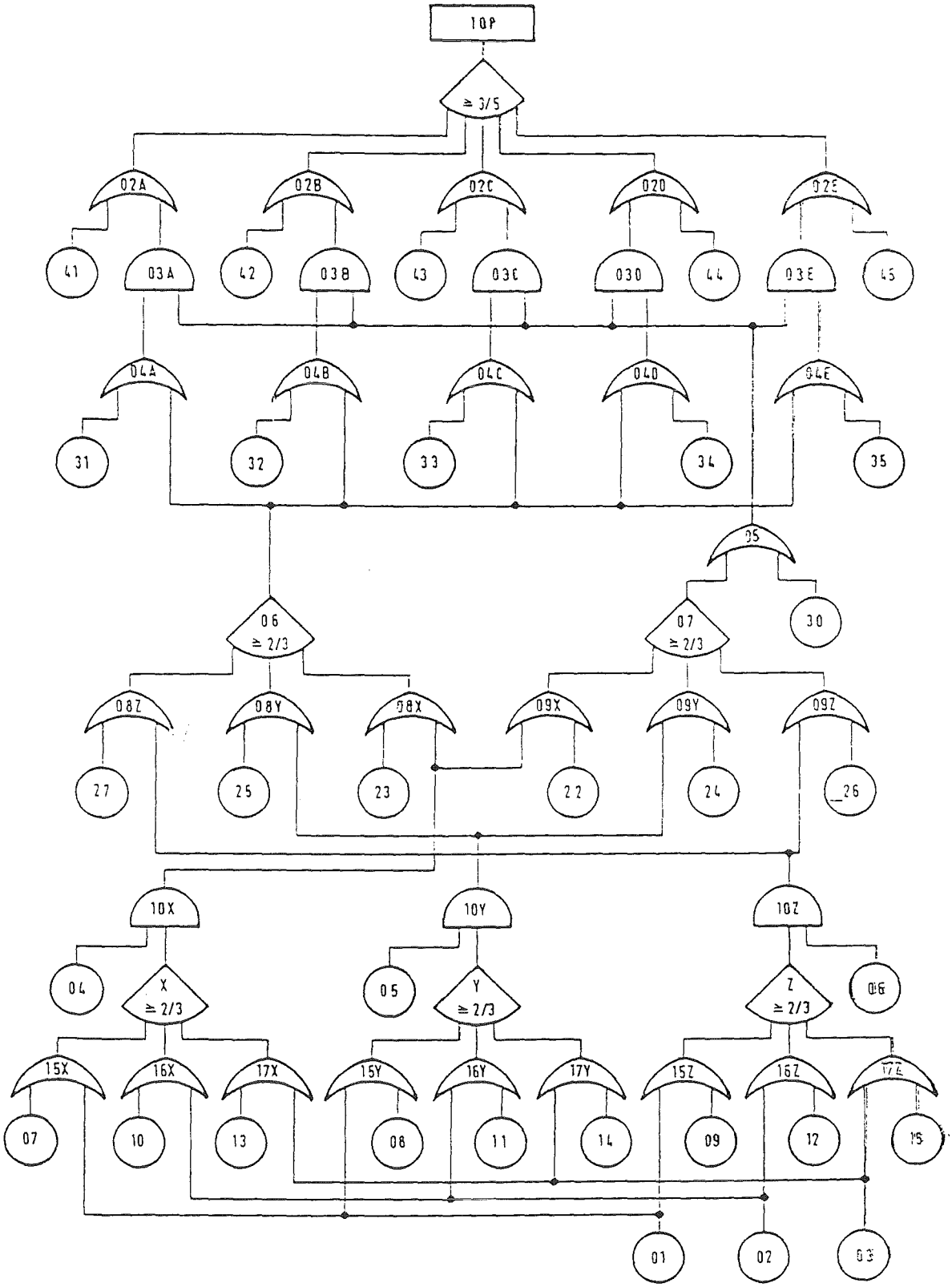
KLEINSTE IRREDUNDANTE BASIS

```
  1    3    4   11   13   22   30   38   47   49   54   56
 57   58   66   68   70   78   81
```

TEST 1:KOMPLEMENT IDENTISCH MIT KOMPLEMENT DER FUNKTION F? JA

TEST 2: PRIMIMPLIKANTEN DER ERRECHNETEN BASIS 19
OBLIGATORISCH FUER DIESE BASIS 19

ZEIT(SEC)= 0.82


TOP
≥ 3/5
02A
02B
02C
02D
02E
41
03A
42
03B
43
03C
03D
44
03E
45
04A
04B
04C
04D
04E
31
32
33
34
35
05
06
≥ 2/3
07
≥ 2/3
30
Z08
Y08
X08
X09
Y09
Z09
27
25
23
22
24
26
10X
10Y
10Z
04
X
≥ 2/3
05
Y
≥ 2/3
Z
≥ 2/3
06
15X
16X
17X
15Y
16Y
17Y
15Z
16Z
17Z
07
10
13
08
11
14
09
12
15
01
02
03

FAULT TREE IDENT.= FEHLERBAUM NR.2

REDUCIBLE BOOLEAN FUNCTION OF TOP= 1.71865E+17

SUPERCOMPONENTS WITH TWO STATES(SIMPLE TREES)

LEV GATE	LEV GATE	LEV GATE	LEV GATE	REDUC.FCT.	CRIT.GATES AND COMPON.	KEY-COMP.

BOOL.FUNCT.REDUCED BY SIMPLE TREES= 1.71865E+17

SUPERCOMPONENTS FOUND INNER TOTAL HORIZONTAL CUTS

LEV GATE	LEV GATE	LEV GATE	LEV GATE	REDUC.FCT.	CRIT.GATES AND COMPON.	KEY-COMP.
4 05	5 06			257556	10Z C003 C002 C001 10Y 10X 06	
5 06	5 07			257049	10Z C003 C002 C001 10Y 10X	
5 07	6 08Y	6 08X	6 08Z	1113879	10Z C003 C002 C001 10Y 10X	
6 09Y	6 08Y	7 10X	7 10Z	24336	10Y C003 C002 C001 10X 10Z	
7 10Z	7 10X	7 10Y		1728	C003 C002 C001 10X 10Y	
7 10Y	7 10X	8 Z		1728	C003 C002 C001 10X	
7 10X	8 Y	8 Z		1728	C003 C002 C001	
8 Z	8 Y	8 X		1728	C003 C002 C001	
9 17Z	9 17X	9 17Y		8	C003	
9 16Y	9 16X	9 16Z		8	C002	
9 15Y	9 15Z	9 15X		8	C001	

PROBLEM: TESTFEHLERBAUM NR.2

NAME	STATE	TRANS.RATES RESP. OCCURRENCE PROBABILITIES
C001	2	0 1 0.40000E-04 1 0 0.14000E-02
		0 0.97222E+00 1 0.27778E-01
C002	2	0 1 0.40000E-04 1 0 0.14000E-02
		0 0.97222E+00 1 0.27778E-01
C003	2	0 1 0.60000E-04 1 0 0.14000E-02
		0 0.95890E+00 1 0.41096E-01
C004	2	0 1 0.70000E-04 1 0 0.14000E-02
		0 0.95238E+00 1 0.47619E-01
C005	2	0 1 0.50000E-04 1 0 0.14000E-02
		0 0.96552E+00 1 0.34483E-01
C006	2	0 1 0.50000E-04 1 0 0.14000E-02
		0 0.96552E+00 1 0.34483E-01
C007	2	0 1 0.10000E-03 1 0 0.14000E-02
		0 0.93333E+00 1 0.66667E-01
C008	2	0 1 0.10000E-03 1 0 0.14000E-02
		0 0.93333E+00 1 0.66667E-01
C009	2	0 1 0.10000E-03 1 0 0.14000E-02
		0 0.93333E+00 1 0.66667E-01
C010	2	0 1 0.80000E-04 1 0 0.14000E-02
		0 0.94595E+00 1 0.54054E-01
C011	2	0 1 0.80000E-04 1 0 0.14000E-02
		0 0.94595E+00 1 0.54054E-01
C012	2	0 1 0.80000E-04 1 0 0.14000E-02
		0 0.94595E+00 1 0.54054E-01
C013	2	0 1 0.12000E-03 1 0 0.14000E-02
		0 0.92105E+00 1 0.78947E-01
C014	2	0 1 0.12000E-03 1 0 0.14000E-02
		0 0.92105E+00 1 0.78947E-01
C015	2	0 1 0.12000E-03 1 0 0.14000E-02
		0 0.92105E+00 1 0.78947E-01
C022	2	0 1 0.10000E-03 1 0 0.60000E-02
		0 0.98361E+00 1 0.16393E-01
C023	2	0 1 0.10000E-03 1 0 0.60000E-02
		0 0.98361E+00 1 0.16393E-01
C024	2	0 1 0.10000E-03 1 0 0.60000E-02
		0 0.98361E+00 1 0.16393E-01
C025	2	0 1 0.70000E-04 1 0 0.60000E-02
		0 0.98847E+00 1 0.11532E-01
C026	2	0 1 0.70000E-04 1 0 0.60000E-02
		0 0.98847E+00 1 0.11532E-01
C027	2	0 1 0.70000E-04 1 0 0.60000E-02
		0 0.98847E+00 1 0.11532E-01
C030	2	0 1 0.33000E-03 1 0 0.42000E-01
		0 0.99220E+00 1 0.77959E-02
C031	2	0 1 0.40000E-03 1 0 0.42000E-01
		0 0.99057E+00 1 0.94340E-02
C032	2	0 1 0.40000E-03 1 0 0.42000E-01
		0 0.99057E+00 1 0.94340E-02
C033	2	0 1 0.40000E-03 1 0 0.42000E-01
		0 0.99057E+00 1 0.94340E-02
C034	2	0 1 0.40000E-03 1 0 0.42000E-01
		0 0.99057E+00 1 0.94340E-02
C035	2	0 1 0.40000E-03 1 0 0.42000E-01
		0 0.99057E+00 1 0.94340E-02
C041	2	0 1 0.60000E-03 1 0 0.42000E-01
		0 0.98592E+00 1 0.14085E-01
C042	2	0 1 0.60000E-03 1 0 0.42000E-01
		0 0.98592E+00 1 0.14085E-01
C043	2	0 1 0.60000E-03 1 0 0.42000E-01
		0 0.98592E+00 1 0.14085E-01
C044	2	0 1 0.60000E-03 1 0 0.42000E-01
		0 0.98592E+00 1 0.14085E-01
C045	2	0 1 0.60000E-03 1 0 0.42000E-01

0 0.98592E+00 1 0.14085E-01

INPUT GATES

GATE	TYP	PREDECESSORS							
15X	OR	C001	1	C007	1				
16X	OR	C002	1	C010	1				
17X	OR	C003	1	C013	1				
15Y	OR	C001	1	C008	1				
16Y	OR	C002	1	C011	1				
17Y	OR	C003	1	C014	1				
15Z	OR	C001	1	C009	1				
16Z	OR	C002	1	C012	1				
17Z	OR	C003	1	C015	1				
X	MAJ	15X	0	16X	0	17X	0		
Y	MAJ	15Y	0	16Y	0	17Y	0		
Z	MAJ	15Z	0	16Z	0	17Z	0		
10X	AND	C004	1	X	0				
10Y	AND	C005	1	Y	0				
10Z	AND	C006	1	Z	0				
08X	OR	C023	1	10X	0				
09X	OR	C022	1	10X	0				
08Y	OR	C025	1	10Y	0				
09Y	OR	C024	1	10Y	0				
08Z	OR	C027	1	10Z	0				
09Z	OR	C026	1	10Z	0				
06	MAJ	08X	0	08Y	0	08Z	0		
07	MAJ	09X	0	09Y	0	09Z	0		
05	OR	C030	1	07	0				
04A	OR	C031	1	06	0				
04B	OR	C032	1	06	0				
04C	OR	C033	1	06	0				
04D	OR	C034	1	06	0				
04E	OR	C035	1	06	0				
03A	AND	05	0	04A	0				
03B	AND	05	0	04B	0				
03C	AND	05	0	04C	0				
03D	AND	05	0	04D	0				
03E	AND	05	0	04E	0				
02A	OR	C041	1	03A	0				
02B	OR	C042	1	03B	0				
02C	OR	C043	1	03C	0				
02D	OR	C044	1	03D	0				
02E	OR	C045	1	03E	0				
TOP	MAJ	02A	0	02B	0	02C	0	02D	0	02E	0

LIST OF GATES,ORDERED BY PROGRAM

GATE	TYP	PREDECESSORS			
15X	OR	C001	1	C007	1
16X	OR	C002	1	C010	1
17X	OR	C003	1	C013	1
15Y	OR	C001	1	C008	1
16Y	OR	C002	1	C011	1
17Y	OR	C003	1	C014	1
15Z	OR	C001	1	C009	1
16Z	OR	C002	1	C012	1
17Z	OR	C003	1	C015	1
1011	AND	15X	1	16X	1
1012	AND	15X	1	17X	1
1013	AND	16X	1	17X	1
1015	AND	15Y	1	16Y	1

OPTION NR. 3 FUNCTION OF GATE 10X 10Y 10Z

SUPERCOMPONENT: SC01 STATE 1 COMB.OF GATES: 10X 1 10Y 0 10Z 0

ASSOCIATED COHERENT FUNCT. OF SUPEREVENT: SC01 STATE 1

1	C004	1	C001	1	C002	1					3.6743E-05
2	C004	1	C001	1	C010	1					7.1500E-05
3	C004	1	C007	1	C002	1					8.8183E-05
4	C004	1	C007	1	C010	1					1.7160E-04
5	C004	1	C002	1	C003	1					5.4360E-05
6	C004	1	C002	1	C013	1					1.0443E-04
7	C004	1	C010	1	C003	1					1.0578E-04
8	C004	1	C010	1	C013	1					2.0321E-04
9	C004	1	C001	1	C003	1					5.4360E-05
10	C004	1	C001	1	C013	1					1.0443E-04
11	C004	1	C007	1	C003	1					1.3046E-04
12	C004	1	C007	1	C013	1					2.5063E-04

NR.OF MONOMIALS= 12 ----------

SUM= 1.3757E-03

SUPERCOMPONENT: SC01 STATE 2 COMB.OF GATES: 10X 0 10Y 1 10Z 0

ASSOCIATED COHERENT FUNCT. OF SUPEREVENT: SC01 STATE 2

1	C005	1	C001	1	C002	1					2.6607E-05
2	C005	1	C001	1	C011	1					5.1776E-05
3	C005	1	C008	1	C002	1					6.3857E-05
4	C005	1	C008	1	C011	1					1.2426E-04
5	C005	1	C002	1	C003	1					3.9364E-05
6	C005	1	C002	1	C014	1					7.5620E-05
7	C005	1	C011	1	C003	1					7.6600E-05
8	C005	1	C011	1	C014	1					1.4715E-04
9	C005	1	C001	1	C003	1					3.9364E-05
10	C005	1	C001	1	C014	1					7.5620E-05
11	C005	1	C008	1	C003	1					9.4473E-05
12	C005	1	C008	1	C014	1					1.8149E-04

NR.OF MONOMIALS= 12 ----------

SUM= 9.9618E-04

SUPERCOMPONENT: SC01 STATE 3 COMB.OF GATES: 10X 1 10Y 1 10Z 0

ASSOCIATED COHERENT FUNCT. OF SUPEREVENT: SC01 STATE 3

1	C004	1	C001	1	C005	1	C002	1							1.2670E-06
2	C004	1	C001	1	C005	1	C011	1	C010	1					1.3327E-07
3	C004	1	C001	1	C005	1	C014	1	C010	1					1.9465E-07
4	C004	1	C007	1	C005	1	C008	1	C002	1					2.0272E-07
5	C004	1	C007	1	C005	1	C008	1	C010	1	C011	1			2.1323E-08
6	C004	1	C007	1	C005	1	C002	1	C014	1					2.4006E-07
7	C004	1	C007	1	C005	1	C011	1	C010	1	C014	1			2.5251E-08
8	C004	1	C007	1	C005	1	C008	1	C010	1	C014	1			3.1143E-08
9	C004	1	C002	1	C005	1	C008	1	C013	1					2.4006E-07
10	C004	1	C002	1	C005	1	C003	1							1.8745E-06
11	C004	1	C002	1	C005	1	C014	1	C013	1					2.8429E-07
12	C004	1	C010	1	C005	1	C008	1	C013	1	C011	1			2.5251E-08
13	C004	1	C010	1	C005	1	C011	1	C003	1					1.9717E-07
14	C004	1	C010	1	C005	1	C011	1	C013	1	C014	1			2.9903E-08
15	C004	1	C010	1	C005	1	C008	1	C003	1					2.4317E-07
16	C004	1	C010	1	C005	1	C008	1	C013	1	C014	1			3.6880E-08
17	C004	1	C001	1	C005	1	C011	1	C013	1					1.9465E-07
18	C004	1	C001	1	C005	1	C003	1							1.8745E-06
19	C004	1	C001	1	C005	1	C014	1	C013	1					2.8429E-07

20	C004 1	C007 1	C005 1	C008 1	C013 1	C011 1					3.1143E-08
21	C004 1	C007 1	C005 1	C011 1	C003 1						2.4317E-07
22	C004 1	C007 1	C005 1	C011 1	C013 1	C014 1					3.6880E-08
23	C004 1	C007 1	C005 1	C008 1	C003 1						2.9992E-07
24	C004 1	C007 1	C005 1	C008 1	C013 1	C014 1					4.5486E-08
						NR.OF MONOMIALS=		24			----------
										SUM=	8.0566E-06

SUPERCOMPONENT: SC01 STATE 4 COMB.OF GATES: 10X 0 10Y 0 10Z 1

ASSOCIATED COHERENT FUNCT. OF SUPEREVENT: SC01 STATE 4

1	C006 1	C001 1	C002 1								2.6607E-05
2	C006 1	C001 1	C012 1								5.1776E-05
3	C006 1	C009 1	C002 1								6.3857E-05
4	C006 1	C009 1	C012 1								1.2426E-04
5	C006 1	C002 1	C003 1								3.9364E-05
6	C006 1	C002 1	C015 1								7.5620E-05
7	C006 1	C012 1	C003 1								7.6600E-05
8	C006 1	C012 1	C015 1								1.4715E-04
9	C006 1	C001 1	C003 1								3.9364E-05
10	C006 1	C001 1	C015 1								7.5620E-05
11	C006 1	C009 1	C003 1								9.4473E-05
12	C006 1	C009 1	C015 1								1.8149E-04
						NR.OF MONOMIALS=		12			----------
										SUM=	9.9618E-04

SUPERCOMPONENT: SC01 STATE 5 COMB.OF GATES: 10X 1 10Y 0 10Z 1

ASSOCIATED COHERENT FUNCT. OF SUPEREVENT: SC01 STATE 5

1	C004 1	C001 1	C006 1	C002 1							1.2670E-06
2	C004 1	C001 1	C006 1	C012 1	C010 1						1.3327E-07
3	C004 1	C001 1	C006 1	C015 1	C010 1						1.9465E-07
4	C004 1	C007 1	C006 1	C009 1	C002 1						2.0272E-07
5	C004 1	C007 1	C006 1	C009 1	C010 1	C012 1					2.1323E-08
6	C004 1	C007 1	C006 1	C002 1	C015 1						2.4006E-07
7	C004 1	C007 1	C006 1	C012 1	C010 1	C015 1					2.5251E-08
8	C004 1	C007 1	C006 1	C009 1	C010 1	C015 1					3.1143E-08
9	C004 1	C002 1	C006 1	C009 1	C013 1						2.4006E-07
10	C004 1	C002 1	C006 1	C003 1							1.8745E-06
11	C004 1	C002 1	C006 1	C015 1	C013 1						2.8429E-07
12	C004 1	C010 1	C006 1	C009 1	C013 1	C012 1					2.5251E-08
13	C004 1	C010 1	C006 1	C012 1	C003 1						1.9717E-07
14	C004 1	C010 1	C006 1	C012 1	C013 1	C015 1					2.9903E-08
15	C004 1	C010 1	C006 1	C009 1	C003 1						2.4317E-07
16	C004 1	C010 1	C006 1	C009 1	C013 1	C015 1					3.6880E-08
17	C004 1	C001 1	C006 1	C012 1	C013 1						1.9465E-07
18	C004 1	C001 1	C006 1	C003 1							1.8745E-06
19	C004 1	C001 1	C006 1	C015 1	C013 1						2.8429E-07
20	C004 1	C007 1	C006 1	C009 1	C013 1	C012 1					3.1143E-08
21	C004 1	C007 1	C006 1	C012 1	C003 1						2.4317E-07
22	C004 1	C007 1	C006 1	C012 1	C013 1	C015 1					3.6880E-08
23	C004 1	C007 1	C006 1	C009 1	C003 1						2.9992E-07
24	C004 1	C007 1	C006 1	C009 1	C013 1	C015 1					4.5486E-08
						NR.OF MONOMIALS=		24			----------
										SUM=	8.0566E-06

SUPERCOMPONENT: SC01 STATE 6 COMB.OF GATES: 10X 0 10Y 1 10Z 1

ASSOCIATED COHERENT FUNCT. OF SUPEREVENT: SC01 STATE 6

1	C005 1	C001 1	C006 1	C002 1	9.1749E-07

2	C005	1	C001	1	C006	1	C012	1	C011	1				9.6507E-08
3	C005	1	C001	1	C006	1	C015	1	C011	1				1.4095E-07
4	C005	1	C008	1	C006	1	C009	1	C002	1				1.4680E-07
5	C005	1	C008	1	C006	1	C009	1	C011	1	C012	1		1.5441E-08
6	C005	1	C008	1	C006	1	C002	1	C015	1				1.7384E-07
7	C005	1	C008	1	C006	1	C012	1	C011	1	C015	1		1.8286E-08
8	C005	1	C008	1	C006	1	C009	1	C011	1	C015	1		2.2552E-08
9	C005	1	C002	1	C006	1	C009	1	C014	1				1.7384E-07
10	C005	1	C002	1	C006	1	C003	1						1.3574E-06
11	C005	1	C002	1	C006	1	C015	1	C014	1				2.0586E-07
12	C005	1	C011	1	C006	1	C009	1	C014	1	C012	1		1.8286E-08
13	C005	1	C011	1	C006	1	C012	1	C003	1				1.4278E-07
14	C005	1	C011	1	C006	1	C012	1	C014	1	C015	1		2.1654E-08
15	C005	1	C011	1	C006	1	C009	1	C003	1				1.7609E-07
16	C005	1	C011	1	C006	1	C009	1	C014	1	C015	1		2.6706E-08
17	C005	1	C001	1	C006	1	C012	1	C014	1				1.4095E-07
18	C005	1	C001	1	C006	1	C003	1						1.3574E-06
19	C005	1	C001	1	C006	1	C015	1	C014	1				2.0586E-07
20	C005	1	C008	1	C006	1	C009	1	C014	1	C012	1		2.2552E-08
21	C005	1	C008	1	C006	1	C012	1	C003	1				1.7609E-07
22	C005	1	C008	1	C006	1	C012	1	C014	1	C015	1		2.6706E-08
23	C005	1	C008	1	C006	1	C009	1	C003	1				2.1718E-07
24	C005	1	C008	1	C006	1	C009	1	C014	1	C015	1		3.2938E-08
											NR.OF MONOMIALS=		24	----------
													SUM=	5.8341E-06

SUPERCOMPONENT: SC01 STATE 7 COMB.OF GATES: 10X 1 10Y 1 10Z 1

1	C004	1	C001	1	C006	1	C002	1	C005	1						4.3690E-08
2	C004	1	C001	1	C006	1	C012	1	C010	1	C005	1	C011	1		2.4841E-10
3	C004	1	C001	1	C006	1	C012	1	C010	1	C005	1	C014	1		3.6281E-10
4	C004	1	C001	1	C006	1	C015	1	C010	1	C005	1	C011	1		3.6281E-10
5	C004	1	C001	1	C006	1	C015	1	C010	1	C005	1	C014	1		5.2989E-10
6	C004	1	C007	1	C006	1	C009	1	C002	1	C005	1	C008	1		4.6602E-10
7	C004	1	C007	1	C006	1	C009	1	C002	1	C005	1	C014	1		5.5187E-10
8	C004	1	C007	1	C006	1	C009	1	C010	1	C005	1	C008	1	C012 1 C011 1	2.6497E-12
9	C004	1	C007	1	C006	1	C009	1	C010	1	C005	1	C011	1	C012 1 C014 1	3.1378E-12
10	C004	1	C007	1	C006	1	C009	1	C010	1	C005	1	C008	1	C012 1 C014 1	3.8699E-12
11	C004	1	C007	1	C006	1	C002	1	C015	1	C005	1	C008	1		5.5187E-10
12	C004	1	C007	1	C006	1	C002	1	C015	1	C005	1	C014	1		6.5353E-10
13	C004	1	C007	1	C006	1	C012	1	C010	1	C005	1	C008	1	C015 1 C011 1	3.1378E-12
14	C004	1	C007	1	C006	1	C012	1	C010	1	C005	1	C011	1	C015 1 C014 1	3.7158E-12
15	C004	1	C007	1	C006	1	C012	1	C010	1	C005	1	C008	1	C015 1 C014 1	4.5828E-12
16	C004	1	C007	1	C006	1	C009	1	C010	1	C005	1	C008	1	C015 1 C011 1	3.8699E-12
17	C004	1	C007	1	C006	1	C009	1	C010	1	C005	1	C011	1	C015 1 C014 1	4.5828E-12
18	C004	1	C007	1	C006	1	C009	1	C010	1	C005	1	C008	1	C015 1 C014 1	5.6522E-12
19	C004	1	C002	1	C006	1	C009	1	C013	1	C005	1	C008	1		5.5187E-10
20	C004	1	C002	1	C006	1	C009	1	C013	1	C005	1	C014	1		6.5353E-10
21	C004	1	C002	1	C006	1	C003	1	C005	1						6.4637E-08
22	C004	1	C002	1	C006	1	C015	1	C013	1	C005	1	C008	1		6.5353E-10
23	C004	1	C002	1	C006	1	C015	1	C013	1	C005	1	C014	1		7.7392E-10
24	C004	1	C010	1	C006	1	C009	1	C013	1	C005	1	C008	1	C012 1 C011 1	3.1378E-12
25	C004	1	C010	1	C006	1	C009	1	C013	1	C005	1	C011	1	C012 1 C014 1	3.7158E-12
26	C004	1	C010	1	C006	1	C009	1	C013	1	C005	1	C008	1	C012 1 C014 1	4.5828E-12
27	C004	1	C010	1	C006	1	C012	1	C003	1	C005	1	C011	1		3.6751E-10
28	C004	1	C010	1	C006	1	C012	1	C003	1	C005	1	C008	1		4.5326E-10
29	C004	1	C010	1	C006	1	C012	1	C013	1	C005	1	C008	1	C015 1 C011 1	3.7158E-12
30	C004	1	C010	1	C006	1	C012	1	C013	1	C005	1	C011	1	C015 1 C014 1	4.4003E-12
31	C004	1	C010	1	C006	1	C012	1	C013	1	C005	1	C008	1	C015 1 C014 1	5.4270E-12
32	C004	1	C010	1	C006	1	C009	1	C003	1	C005	1	C011	1		4.5326E-10
33	C004	1	C010	1	C006	1	C009	1	C003	1	C005	1	C008	1		5.5902E-10
34	C004	1	C010	1	C006	1	C009	1	C013	1	C005	1	C008	1	C015 1 C011 1	4.5828E-12
35	C004	1	C010	1	C006	1	C009	1	C013	1	C005	1	C011	1	C015 1 C014 1	5.4270E-12
36	C004	1	C010	1	C006	1	C009	1	C013	1	C005	1	C008	1	C015 1 C014 1	6.6933E-12
37	C004	1	C001	1	C006	1	C012	1	C013	1	C005	1	C011	1		3.6281E-10
38	C004	1	C001	1	C006	1	C012	1	C013	1	C005	1	C014	1		5.2989E-10

39	C004 1	C001 1	C006 1	C003 1	C005 1						6.4637E-08
40	C004 1	C001 1	C006 1	C015 1	C013 1	C005 1	C011 1				5.2989E-10
41	C004 1	C001 1	C006 1	C015 1	C013 1	C005 1	C014 1				7.7392E-10
42	C004 1	C007 1	C006 1	C009 1	C013 1	C005 1	C008 1	C012 1	C011 1		3.8699E-12
43	C004 1	C007 1	C006 1	C009 1	C013 1	C005 1	C011 1	C012 1	C014 1		4.5828E-12
44	C004 1	C007 1	C006 1	C009 1	C013 1	C005 1	C008 1	C012 1	C014 1		5.6522E-12
45	C004 1	C007 1	C006 1	C012 1	C003 1	C005 1	C011 1				4.5326E-10
46	C004 1	C007 1	C006 1	C012 1	C003 1	C005 1	C008 1				5.5902E-10
47	C004 1	C007 1	C006 1	C012 1	C013 1	C005 1	C008 1	C015 1	C011 1		4.5828E-12
48	C004 1	C007 1	C006 1	C012 1	C013 1	C005 1	C011 1	C015 1	C014 1		5.4270E-12
49	C004 1	C007 1	C006 1	C012 1	C013 1	C005 1	C008 1	C015 1	C014 1		6.6933E-12
50	C004 1	C007 1	C006 1	C009 1	C003 1	C005 1	C011 1				5.5902E-10
51	C004 1	C007 1	C006 1	C009 1	C003 1	C005 1	C008 1				6.8946E-10
52	C004 1	C007 1	C006 1	C009 1	C013 1	C005 1	C008 1	C015 1	C011 1		5.6522E-12
53	C004 1	C007 1	C006 1	C009 1	C013 1	C005 1	C011 1	C015 1	C014 1		6.6933E-12
54	C004 1	C007 1	C006 1	C009 1	C013 1	C005 1	C008 1	C015 1	C014 1		8.2551E-12
						NR.OF MON.IN MOD.=		54		SUM=	1.8574E-07

ZEIT (SEC) = 0.5832

OPTION NR. 3 FUNCTION OF GATE 06 07

SUPERCOMPONENT: SC02 STATE 1 COMB.OF GATES: 06 1 07 0

ASSOCIATED COHERENT FUNCT. OF SUPEREVENT: SC02 STATE 1

1	C023 1	C025 1			1.8905E-04
2	C025 1	C027 1			1.3299E-04
3	C023 1	C027 1			1.8905E-04
4	SC01 4	C025 1			1.1488E-05
5	SC01 4	C023 1			1.6331E-05
6	SC01 2	C023 1			1.6331E-05
7	SC01 2	C027 1			1.1488E-05
8	SC01 1	C025 1			1.5865E-05
9	SC01 1	C027 1			1.5865E-05
			NR.OF MONOMIALS=	75	---------
				SUM=	5.9846E-04

SUPERCOMPONENT: SC02 STATE 2 COMB.OF GATES: 06 0 07 1

ASSOCIATED COHERENT FUNCT. OF SUPEREVENT: SC02 STATE 2

1	C022 1	C024 1			2.6874E-04
2	C022 1	SC01 2			1.6331E-05
3	SC01 1	C024 1			2.2552E-05
4	C024 1	C026 1			1.8905E-04
5	C024 1	SC01 4			1.6331E-05
6	SC01 2	C026 1			1.1488E-05
7	C022 1	C026 1			1.8905E-04
8	C022 1	SC01 4			1.6331E-05
9	SC01 1	C026 1			1.5865E-05
			NR.OF MONOMIALS=	75	---------
				SUM=	7.4574E-04

SUPERCOMPONENT: SC02 STATE 3 COMB.OF GATES: 06 1 07 1

1	C022 1	C024 1	C023 1	C025 1		5.0807E-08
2	C022 1	C024 1	C025 1	C027 1		3.5740E-08
3	C022 1	C024 1	C023 1	C027 1		5.0807E-08
4	C022 1	SC01 2	C023 1			2.6772E-07
5	C022 1	SC01 2	C027 1			1.8833E-07
6	SC01 1	C024 1	C025 1			2.6007E-07
7	SC01 1	C024 1	C027 1			2.6007E-07
8	SC01 7					1.8574E-07
9	SC01 3					8.0566E-06
10	C024 1	C026 1	C023 1	C025 1		3.5740E-08
11	C024 1	C026 1	C025 1	C027 1		2.5142E-08
12	C024 1	C026 1	C023 1	C027 1		3.5740E-08
13	C024 1	SC01 4	C025 1			1.8833E-07
14	C024 1	SC01 4	C023 1			2.6772E-07
15	SC01 2	C026 1	C023 1			1.8833E-07
16	SC01 2	C026 1	C027 1			1.3248E-07
17	SC01 6					5.8341E-06
18	C022 1	C026 1	C023 1	C025 1		3.5740E-08
19	C022 1	C026 1	C025 1	C027 1		2.5142E-08
20	C022 1	C026 1	C023 1	C027 1		3.5740E-08
21	C022 1	SC01 4	C025 1			1.8833E-07
22	C022 1	SC01 4	C023 1			2.6772E-07
23	SC01 1	C026 1	C025 1			1.8295E-07
24	SC01 1	C026 1	C027 1			1.8295E-07
25	SC01 5					8.0566E-06
				NR.OF MON.IN MOD.=	279	---------
					SUM=	2.5039E-05

OPTION NR. 2 FUNCTION OF GATE TOP

KEY-COMP.: C030 SC02

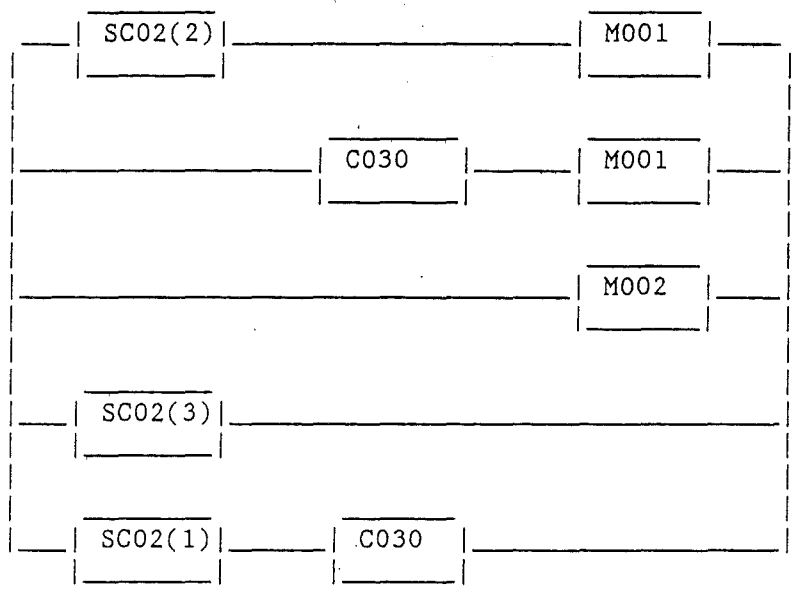
ROW 1 (FCT.OF GATE TOP) = MODUL M001 AND KEY-COMP.: SC02 2

1	C031 1	C034 1	C032 1	8.3962E-07
2	C031 1	C034 1	C042 1	1.2535E-06
3	C031 1	C044 1	C032 1	1.2535E-06
4	C031 1	C044 1	C042 1	1.8714E-06
5	C041 1	C032 1	C034 1	1.2535E-06
6	C041 1	C032 1	C044 1	1.8714E-06
7	C041 1	C042 1	C034 1	1.8714E-06
8	C031 1	C035 1	C032 1	8.3962E-07
9	C031 1	C035 1	C042 1	1.2535E-06
10	C031 1	C045 1	C032 1	1.2535E-06
11	C031 1	C045 1	C042 1	1.8714E-06
12	C041 1	C032 1	C035 1	1.2535E-06
13	C041 1	C032 1	C045 1	1.8714E-06
14	C041 1	C042 1	C035 1	1.8714E-06
15	C031 1	C034 1	C033 1	8.3962E-07
16	C031 1	C034 1	C043 1	1.2535E-06
17	C031 1	C044 1	C033 1	1.2535E-06
18	C031 1	C044 1	C043 1	1.8714E-06
19	C041 1	C033 1	C034 1	1.2535E-06
20	C041 1	C033 1	C044 1	1.8714E-06
21	C041 1	C043 1	C034 1	1.8714E-06
22	C031 1	C035 1	C033 1	8.3962E-07
23	C031 1	C035 1	C043 1	1.2535E-06
24	C031 1	C045 1	C033 1	1.2535E-06
25	C031 1	C045 1	C043 1	1.8714E-06
26	C041 1	C033 1	C035 1	1.2535E-06
27	C041 1	C033 1	C045 1	1.8714E-06
28	C041 1	C043 1	C035 1	1.8714E-06
29	C031 1	C035 1	C034 1	8.3962E-07
30	C031 1	C035 1	C044 1	1.2535E-06
31	C031 1	C045 1	C034 1	1.2535E-06
32	C031 1	C045 1	C044 1	1.8714E-06
33	C041 1	C034 1	C035 1	1.2535E-06
34	C041 1	C034 1	C045 1	1.8714E-06
35	C041 1	C044 1	C035 1	1.8714E-06
36	C032 1	C034 1	C033 1	8.3962E-07
37	C032 1	C034 1	C043 1	1.2535E-06
38	C032 1	C044 1	C033 1	1.2535E-06
39	C032 1	C044 1	C043 1	1.8714E-06
40	C042 1	C033 1	C034 1	1.2535E-06
41	C042 1	C033 1	C044 1	1.8714E-06
42	C042 1	C043 1	C034 1	1.8714E-06
43	C032 1	C035 1	C034 1	8.3962E-07
44	C032 1	C035 1	C044 1	1.2535E-06
45	C032 1	C045 1	C034 1	1.2535E-06
46	C032 1	C045 1	C044 1	1.8714E-06
47	C042 1	C034 1	C035 1	1.2535E-06
48	C042 1	C034 1	C045 1	1.8714E-06
49	C042 1	C044 1	C035 1	1.8714E-06
50	C033 1	C035 1	C034 1	8.3962E-07
51	C033 1	C035 1	C044 1	1.2535E-06
52	C033 1	C045 1	C034 1	1.2535E-06
53	C033 1	C045 1	C044 1	1.8714E-06
54	C043 1	C034 1	C035 1	1.2535E-06
55	C043 1	C034 1	C045 1	1.8714E-06
56	C043 1	C044 1	C035 1	1.8714E-06
57	C032 1	C035 1	C033 1	8.3962E-07
58	C032 1	C035 1	C043 1	1.2535E-06
59	C032 1	C045 1	C033 1	1.2535E-06

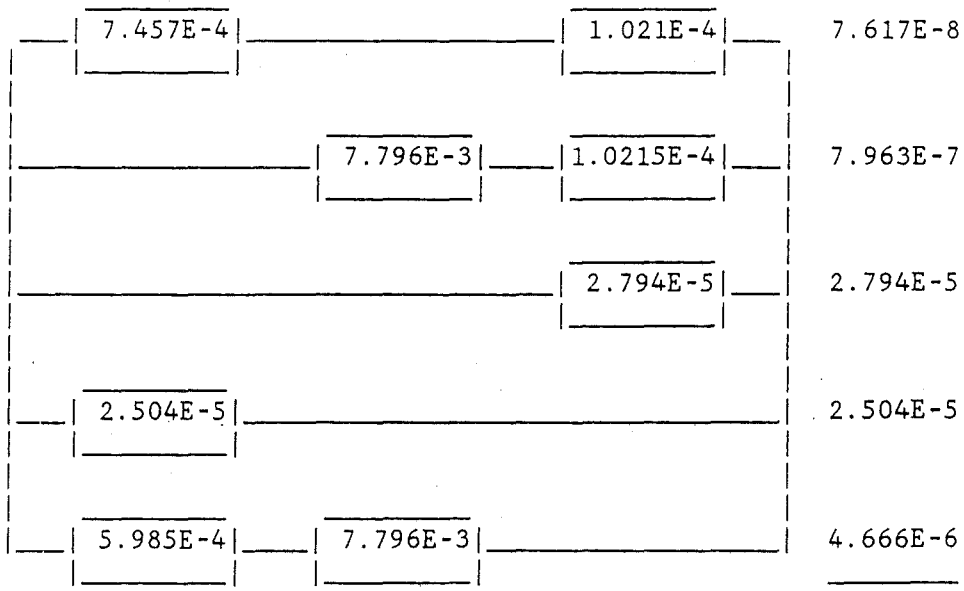
```
60   C032 1   C045 1   C043 1                                                    1.8714E-06
61   C042 1   C033 1   C035 1                                                    1.2535E-06
62   C042 1   C033 1   C045 1                                                    1.8714E-06
63   C042 1   C043 1   C035 1                                                    1.8714E-06
64   C031 1   C033 1   C032 1                                                    8.3962E-07
65   C031 1   C033 1   C042 1                                                    1.2535E-06
66   C031 1   C043 1   C032 1                                                    1.2535E-06
67   C031 1   C043 1   C042 1                                                    1.8714E-06
68   C041 1   C032 1   C033 1                                                    1.2535E-06
69   C041 1   C032 1   C043 1                                                    1.8714E-06
70   C041 1   C042 1   C033 1                                                    1.8714E-06
                                                                                ----------
                                        NR OF MON.IN MOD.=          70          SUM=    1.0215E-04
                                        NR OF MON.IN ROW=         5250          P(ROW)=              7.6174E-08

 ROW   2 ( FCT.OF GATE TOP  ) = MODUL  M001   AND KE--COMP.:   C030  1
                                        NR OF MON.IN ROW=           70          P(ROW)=              7.9631E-07

 ROW   3 ( FCT.OF GATE TOP  ) = MODUL  M002

 1   C041 1   C042 1   C044 1                                                    2.7940E-06
 2   C041 1   C042 1   C045 1                                                    2.7940E-06
 3   C041 1   C043 1   C044 1                                                    2.7940E-06
 4   C041 1   C043 1   C045 1                                                    2.7940E-06
 5   C041 1   C044 1   C045 1                                                    2.7940E-06
 6   C042 1   C043 1   C044 1                                                    2.7940E-06
 7   C042 1   C044 1   C045 1                                                    2.7940E-06
 8   C043 1   C044 1   C045 1                                                    2.7940E-06
 9   C042 1   C043 1   C045 1                                                    2.7940E-06
10   C041 1   C042 1   C043 1                                                    2.7940E-06
                                                                                ----------
                                        NR OF MON.IN MOD.=          10          SUM=    2.7940E-05
                                        NR OF MON.IN ROW=           10          P(ROW)=              2.7940E-05

 ROW   4 ( FCT.OF GATE TOP  ) = MODUL  -NO-   AND KE--COMP.:   SC02  3
                                        NR OF MON.IN ROW=          279          P(ROW)=              2.5039E-05

 ROW   5 ( FCT.OF GATE TOP  ) = MODUL  -NO-   AND KE--COMP.:   SC02  1   C030  1
                                        NR OF MON.IN ROW=           75          P(ROW)=              4.6655E-06

=========   OCCURRENCE PROBABILITY OF GATE:   TOP   =  5.85166E-05        TOTALNR.OF MONOMIALS=        5684

  ZEIT (SEC) =     0.8536
```

Blockdiagramm der Top-Funktion des Fehlerbaums Nr.2

a) modularisierte Strukturfunktion



b) eingesetzte Wahrscheinlichkeiten



7.617E-8

7.963E-7

2.794E-5

2.504E-5

4.666E-6

5.852E-5

```
PROGRAMM  MUSTAFA (VERSION MIT ARBEITSFELD 96000 INTEGER*4, NOV.88)

EXTERNAL UNITS       10   11   12   13    0    0    0    0

NOPT,NTR,NSC,NPRP=    2    0    0    0                          KPR1-KPR4=    -1    1    0    0

EPS,ALPH,BET,GAM=  1.000E-06 1.660E-01 7.500E-01 1.250E-01

CARD INPUT
        SC01 3,5,6 = MAJ 2/3 (10X,10Y,10Z)
C001    2    2
 0 1  0.00004   1 0  0.0014
C002    2    2
 0 1  0.00004   1 0  0.0014
C003    2    2
 0 1  0.00006   1 0  0.0014
C004    2    2
 0 1  0.00007   1 0  0.0014
C005    2    2
 0 1  0.00005   1 0  0.0014
C006    2    2
 0 1  0.00005   1 0  0.0014
C007    2    2
 0 1  0.0001    1 0  0.0014
C008    2    2
 0 1  0.0001    1 0  0.0014
C009    2    2
 0 1  0.0001    1 0  0.0014
C010    2    2
 0 1  0.00008   1 0  0.0014
C011    2    2
 0 1  0.00008   1 0  0.0014
C012    2    2
 0 1  0.00008   1 0  0.0014
C013    2    2
 0 1  0.00012   1 0  0.0014
C014    2    2
 0 1  0.00012   1 0  0.0014
C015    2    2
 0 1  0.00012   1 0  0.0014
END
15X OR    2C001 1 C007 1
16X OR    2C002 1 C010 1
17X OR    2C003 1 C013 1
15Y OR    2C001 1 C008 1
16Y OR    2C002 1 C011 1
17Y OR    2C003 1 C014 1
15Z OR    2C001 1 C009 1
16Z OR    2C002 1 C012 1
17Z OR    2C003 1 C015 1
X    MAJ 2315X     16X     17X
Y    MAJ 2315Y     16Y     17Y
Z    MAJ 2315Z     16Z     17Z
10X AND   2C004 1 X
10Y AND   2C005 1 Y
10Z AND   2C006 1 Z
SC01MAJ 2310X     10Y     10Z
END
--------

NR.OF PATHS TO TOP  4.320E+02   TO NOT TOP  5.832E+03

DOWNWARD ALGORITHM EXECUTED FOR TREE UNDER ELEMENT:  SC010R

CUTS STORED TO UNIT  10      NR.OF MIN.CUTS=    72
```

FAULT TREE IDENT.= SC01 3,5,6 = MAJ 2/3 (10X,10Y,10Z)

COMPOSITION OF MINIM.CUTS RESP.PRIME IMPLIC.

1	C002 1	C003 1	C004 1	C006 1				1.87447E-06
2	C002 1	C003 1	C004 1	C005 1				1.87447E-06
3	C001 1	C003 1	C004 1	C006 1				1.87447E-06
4	C001 1	C003 1	C004 1	C005 1				1.87447E-06
5	C002 1	C003 1	C005 1	C006 1				1.35737E-06
6	C001 1	C003 1	C005 1	C006 1				1.35737E-06
7	C001 1	C002 1	C004 1	C006 1				1.26700E-06
8	C001 1	C002 1	C004 1	C005 1				1.26700E-06
9	C001 1	C002 1	C005 1	C006 1				9.17485E-07
10	C003 1	C004 1	C005 1	C007 1	C008 1			2.99915E-07
11	C003 1	C004 1	C006 1	C007 1	C009 1			2.99915E-07
12	C001 1	C004 1	C006 1	C013 1	C015 1			2.84286E-07
13	C002 1	C004 1	C005 1	C013 1	C014 1			2.84286E-07
14	C002 1	C004 1	C006 1	C013 1	C015 1			2.84286E-07
15	C001 1	C004 1	C005 1	C013 1	C014 1			2.84286E-07
16	C003 1	C004 1	C006 1	C009 1	C010 1			2.43175E-07
17	C003 1	C004 1	C005 1	C008 1	C010 1			2.43175E-07
18	C003 1	C004 1	C005 1	C007 1	C011 1			2.43175E-07
19	C003 1	C004 1	C006 1	C007 1	C012 1			2.43175E-07
20	C002 1	C004 1	C005 1	C007 1	C014 1			2.40064E-07
21	C002 1	C004 1	C005 1	C008 1	C013 1			2.40064E-07
22	C002 1	C004 1	C006 1	C007 1	C015 1			2.40064E-07
23	C002 1	C004 1	C006 1	C009 1	C013 1			2.40064E-07
24	C003 1	C005 1	C006 1	C008 1	C009 1			2.17180E-07
25	C002 1	C005 1	C006 1	C014 1	C015 1			2.05862E-07
26	C001 1	C005 1	C006 1	C014 1	C015 1			2.05862E-07
27	C002 1	C004 1	C005 1	C007 1	C008 1			2.02721E-07
28	C002 1	C004 1	C006 1	C007 1	C009 1			2.02721E-07
29	C003 1	C004 1	C006 1	C010 1	C012 1			1.97169E-07
30	C003 1	C004 1	C005 1	C010 1	C011 1			1.97169E-07
31	C001 1	C004 1	C006 1	C012 1	C013 1			1.94646E-07
32	C001 1	C004 1	C005 1	C010 1	C014 1			1.94646E-07
33	C001 1	C004 1	C006 1	C010 1	C015 1			1.94646E-07
34	C001 1	C004 1	C005 1	C011 1	C013 1			1.94646E-07
35	C003 1	C005 1	C006 1	C008 1	C012 1			1.76092E-07
36	C003 1	C005 1	C006 1	C009 1	C011 1			1.76092E-07
37	C002 1	C005 1	C006 1	C008 1	C015 1			1.73839E-07
38	C002 1	C005 1	C006 1	C009 1	C014 1			1.73839E-07
39	C002 1	C005 1	C006 1	C008 1	C009 1			1.46798E-07
40	C003 1	C005 1	C006 1	C011 1	C012 1			1.42777E-07
41	C001 1	C005 1	C006 1	C012 1	C014 1			1.40951E-07
42	C001 1	C005 1	C006 1	C011 1	C015 1			1.40951E-07
43	C001 1	C004 1	C005 1	C010 1	C011 1			1.33271E-07
44	C001 1	C004 1	C006 1	C010 1	C012 1			1.33271E-07
45	C001 1	C005 1	C006 1	C011 1	C012 1			9.65068E-08
46	C004 1	C006 1	C007 1	C009 1	C013 1	C015 1		4.54858E-08
47	C004 1	C005 1	C007 1	C008 1	C013 1	C014 1		4.54858E-08
48	C004 1	C006 1	C007 1	C012 1	C013 1	C015 1		3.68804E-08
49	C004 1	C005 1	C008 1	C010 1	C013 1	C014 1		3.68804E-08
50	C004 1	C005 1	C007 1	C011 1	C013 1	C014 1		3.68804E-08
51	C004 1	C006 1	C009 1	C010 1	C013 1	C015 1		3.68804E-08
52	C005 1	C006 1	C008 1	C009 1	C014 1	C015 1		3.29380E-08
53	C004 1	C006 1	C007 1	C009 1	C012 1	C013 1		3.11434E-08
54	C004 1	C005 1	C007 1	C008 1	C011 1	C013 1		3.11434E-08
55	C004 1	C006 1	C007 1	C009 1	C010 1	C015 1		3.11434E-08
56	C004 1	C005 1	C007 1	C008 1	C010 1	C014 1		3.11434E-08
57	C004 1	C005 1	C010 1	C011 1	C013 1	C014 1		2.99030E-08
58	C004 1	C006 1	C010 1	C012 1	C013 1	C015 1		2.99030E-08
59	C005 1	C006 1	C008 1	C012 1	C014 1	C015 1		2.67065E-08
60	C005 1	C006 1	C009 1	C011 1	C014 1	C015 1		2.67065E-08
61	C004 1	C006 1	C007 1	C010 1	C012 1	C015 1		2.52514E-08
62	C004 1	C005 1	C007 1	C010 1	C011 1	C014 1		2.52514E-08
63	C004 1	C006 1	C009 1	C010 1	C012 1	C013 1		2.52514E-08

```
64  C004 1      C005 1      C008 1      C010 1      C011 1      C013 1                    2.52514E-08
65  C005 1      C006 1      C008 1      C009 1      C012 1      C014 1                    2.25521E-08
66  C005 1      C006 1      C008 1      C009 1      C011 1      C015 1                    2.25521E-08
67  C005 1      C006 1      C011 1      C012 1      C014 1      C015 1                    2.16539E-08
68  C004 1      C005 1      C007 1      C008 1      C010 1      C011 1                    2.13234E-08
69  C004 1      C006 1      C007 1      C009 1      C010 1      C012 1                    2.13234E-08
70  C005 1      C006 1      C009 1      C011 1      C012 1      C014 1                    1.82855E-08
71  C005 1      C006 1      C008 1      C011 1      C012 1      C015 1                    1.82855E-08
72  C005 1      C006 1      C008 1      C009 1      C011 1      C012 1                    1.54411E-08

   SUM OF OCC.PROB.OF NEGLECTED CUTS=  0.00000E+00

   FUNCTION IS REPRESENTED BY KEYSTONE MONOMIALS

   OCCUR.PROBABILITY OF FAULT TREE < SC01 3,5,6 = MAJ 2/3 (10X,10Y,10Z)      >=  2.02450E-05
    MAX.DEVIAT. =  0.00000E+00

    CPU-TIME(SEC)=      0.588
```

```
PROGRAMM  MUSTAFA (VERSION MIT ARBEITSFELD 96000 INTEGER*4, NOV.88)

EXTERNAL UNITS       10   11   12   13    0    0    0    0

NOPT,NIR,NSE,NPRP=    2    0    0    0                          KPR1-KPR4=    -1   -1    0    0

EPS,ALPH,BET,GAM=   1.000E-06 1.660E-01 7.500E-01 1.250E-01

CARD INPUT
        SC01 STATE 1 (= 10X AND 10Y NOT AND 10 Z NOT)
C001    2    2
 0 1   0.00004   1 0   0.0014
C002    2    2
 0 1   0.00004   1 0   0.0014
C003    2    2
 0 1   0.00006   1 0   0.0014
C004    2    2
 0 1   0.00007   1 0   0.0014
C005    2    2
 0 1   0.00005   1 0   0.0014
C006    2    2
 0 1   0.00005   1 0   0.0014
C007    2    2
 0 1   0.0001    1 0   0.0014
C008    2    2
 0 1   0.0001    1 0   0.0014
C009    2    2
 0 1   0.0001    1 0   0.0014
C010    2    2
 0 1   0.00008   1 0   0.0014
C011    2    2
 0 1   0.00008   1 0   0.0014
C012    2    2
 0 1   0.00008   1 0   0.0014
C013    2    2
 0 1   0.00012   1 0   0.0014
C014    2    2
 0 1   0.00012   1 0   0.0014
C015    2    2
 0 1   0.00012   1 0   0.0014
END
15X OR    2C001 1 C007 1
16X OR    2C002 1 C010 1
17X OR    2C003 1 C013 1
15Y OR    2C001 1 C008 1
16Y OR    2C002 1 C011 1
17Y OR    2C003 1 C014 1
15Z OR    2C001 1 C009 1
16Z OR    2C002 1 C012 1
17Z OR    2C003 1 C015 1
X    MAJ 2315X     16X     17X
Y    MAJ 2315Y     16Y     17Y
Z    MAJ 2315Z     16Z     17Z
10X AND   2C004 1 X
10Y AND   2C005 1 Y
10Z AND   2C006 1 Z
10YNNOT   110Y
10ZNNOT   110Z
SC01AND   310X     10YN    10ZN
END
--------

NR.OF PATHS TO TOP  9.720E+02   TO NOT TOP  3.300E+01

DOWNWARD ALGORITHM EXECUTED FOR TREE UNDER ELEMENT:  SC01NOT
```

```
CUTS STORED TO UNIT  11       NR.OF MIN.CUTS=     28

NELSON ALGOR. STEP  2   NR.OF MONOM.=    75

 BOOLEAN FUNCTION OF FAULT TREE IS REPRESENTED BY  PRIMIMPLIC.OF TOTAL BASE

 SUM OF OCC.PROB.OF NEGLECTED CUTS=  0.00000E+00

 FUNCTION IS REPRESENTED BY KEYSTONE MONOMIALS

 OCCUR.PROBABILITY OF FAULT TREE < SC01 STATE 1 (= 10X AND 10Y NOT AND 10 Z NOT)                    >=  1.21828E-03
  MAX.DEVIAT. =  0.00000E+00

  CPU-TIME(SEC)=      1.523
```

```
PROGRAMM  MUSTAFA (VERSION MIT ARBEITSFELD 96000 INTEGER*4, NOV.88)

EXTERNAL UNITS       10  11  12  13   0   0   0   0

NOPT,NTR,NSE,NPRP=    2   0   0   0                          KPR1-KPR4=    -1  -1   0   0

EPS,ALPH,BET,GAM=   1.000E-06 1.660E-01 7.500E-01 1.250E-01

CARD INPUT
        SC01 STATE 1 (= 10X NOT AND 10Y AND 10 Z NOT)
C001    2    2
 0 1  0.00004   1 0  0.001
C002    2    2
 0 1  0.00004   1 0  0.001
C003    2    2
 0 1  0.00006   1 0  0.001
C004    2    2
 0 1  0.00007   1 0  0.001
C005    2    2
 0 1  0.00005   1 0  0.001
C006    2    2
 0 1  0.00005   1 0  0.001
C007    2    2
 0 1  0.0001    1 0  0.001
C008    2    2
 0 1  0.0001    1 0  0.001
C009    2    2
 0 1  0.0001    1 0  0.001
C010    2    2
 0 1  0.00008   1 0  0.001
C011    2    2
 0 1  0.00008   1 0  0.001
C012    2    2
 0 1  0.00008   1 0  0.001
C013    2    2
 0 1  0.00012   1 0  0.001
C014    2    2
 0 1  0.00012   1 0  0.001
C015    2    2
 0 1  0.00012   1 0  0.001
END
15X OR    2C001 1 C007 1
16X OR    2C002 1 C010 1
17X OR    2C003 1 C013 1
15Y OR    2C001 1 C008 1
16Y OR    2C002 1 C011 1
17Y OR    2C003 1 C014 1
15Z OR    2C001 1 C009 1
16Z OR    2C002 1 C012 1
17Z OR    2C003 1 C015 1
X    MAJ 2315X     16X     17X
Y    MAJ 2315Y     16Y     17Y
Z    MAJ 2315Z     16Z     17Z
10X AND   2C004 1 X
10Y AND   2C005 1 Y
10Z AND   2C006 1 Z
10XNNOT   110X
10ZNNOT   110Z
SC01AND   310XN    10Y     10ZN
END
--------

NR.OF PATHS TO TOP  9.720E+02   TO NOT TOP  3.300E+01

DOWNWARD ALGORITHM EXECUTED FOR TREE UNDER ELEMENT:  SC01NOT
```

```
CUTS STORED TO UNIT  11      NR.OF MIN.CUTS=    28

NELSON ALGOR. STEP  2   NR.OF MONOM.=   75

 BOOLEAN FUNCTION OF FAULT TREE IS REPRESENTED BY  PRIMIMPLIC.OF TOTAL BASE

 SUM OF OCC.PROB.OF NEGLECTED CUTS=  0.00000E+00

 FUNCTION IS REPRESENTED BY KEYSTONE MONOMIALS

 OCCUR.PROBABILITY OF FAULT TREE < SCO1 STATE 1 (= 10X NOT AND 10Y AND 10 Z NOT)                    >=  8.80163E-04
  MAX.DEVIAT. =  0.00000E+00

  CPU-TIME(SEC)=     2.483
```

```
PROGRAMM  MUSTAFA (VERSION MIT ARBEITSFELD 96000 INTEGER*4, NOV.88)

EXTERNAL UNITS       10   11   12   13    0    0    0    0

NOPT,NTR,NSE,NPRP=    3    0    0    0                              KPRI-KPRM=    -1   -1    0    0

EPS,ALPH,BET,GAM=  1.000E-06 1.660E-01 7.500E-01 1.250E-01

CARD INPUT
        SC01 STATE 1 (= 10X NOT AND 10Y NOT AND 10 Z )
C001    2    2
 0 1  0.00004  1 0  0.001
C002    2    2
 0 1  0.00004  1 0  0.001
C003    2    2
 0 1  0.00006  1 0  0.001
C004    2    2
 0 1  0.00007  1 0  0.001
C005    2    2
 0 1  0.00005  1 0  0.001
C006    2    2
 0 1  0.00005  1 0  0.001
C007    2    2
 0 1  0.0001   1 0  0.001
C008    2    2
 0 1  0.0001   1 0  0.001
C009    2    2
 0 1  0.0001   1 0  0.001
C010    2    2
 0 1  0.00008  1 0  0.001
C011    2    2
 0 1  0.00008  1 0  0.001
C012    2    2
 0 1  0.00008  1 0  0.001
C013    2    2
 0 1  0.00012  1 0  0.001
C014    2    2
 0 1  0.00012  1 0  0.001
C015    2    2
 0 1  0.00012  1 0  0.001
END
15X OR   2C001 1 C007 1
16X OR   2C002 1 C010 1
17X OR   2C003 1 C013 1
15Y OR   2C001 1 C008 1
16Y OR   2C002 1 C011 1
17Y OR   2C003 1 C014 1
15Z OR   2C001 1 C009 1
16Z OR   2C002 1 C012 1
17Z OR   2C003 1 C015 1
X   MAJ 2315X    16X    17X
Y   MAJ 2315Y    16Y    17Y
Z   MAJ 2315Z    16Z    17Z
10X AND  2C004 1 X
10Y AND  2C005 1 Y
10Z AND  2C006 1 Z
10XNNOT  110X
10YNNOT  110Y
SC01AND  310XN   10YN   10Z
END
--------

NR.OF PATHS TO TOP  9.720E+02   TO NOT TOP  3.300E+01

DOWNWARD ALGORITHM EXECUTED FOR TREE UNDER ELEMENT:  SC01AND
```

```
CUTS STORED TO UNIT  10      NR.OF MIN.CUTS=     75


BENNETT ALGORITHM, FAULT TREE IDENT.=  SC01 STATE 1 (= 10X NOT AND 10Y NOT AND 10 Z )
-------------------------------------------------------------------------------------------------------------------------
                                                                                                    SUM=       8.80157E-04

OCCUR.PROBABILITY OF FAULT TREE < SC01 STATE 1 (= 10X NOT AND 10Y NOT AND 10 Z )                  >=  8.80157E-04
 MAX.DEVIAT. =  6.32879E-09

 CPU-TIME(SEC)=      3.956
```

SYSTEM: FEHLERBAUM NR.2

NR.OF TRIALS 4000 RAND.INIT.= 0.564000

PROB.VALUES INPUT = MITTELW..

INPUT COMPONENTS

NAME	DATA FROM	NST	NL	INH	TRANSITION RATES RESP.OCCURRENCE PROBABILITIES WITH SPREAD FACTOR
SC02		4	4	0	0 0.9984E+00 0.00 1 0.5770E-03 3.50 2 0.7150E-03 3.50 3 0.2200E-04 3.60
C030		2	2	0	0 1 0.3300E-03 10.00 1 0 0.4200E-01 0.00
C041		2	2	0	0 1 0.6000E-03 3.00 1 0 0.4200E-01 0.00
C042		2	2	0	0 1 0.6000E-03 3.00 1 0 0.4200E-01 0.00
C043		2	2	0	0 1 0.6000E-03 3.00 1 0 0.4200E-01 0.00
C044		2	2	0	0 1 0.6000E-03 3.00 1 0 0.4200E-01 0.00
C045		2	2	0	0 1 0.6000E-03 3.00 1 0 0.4200E-01 0.00

INPUT TOP-FUNCTION BY BOOLEAN TERMS

B0010R	C041	1	0	C042	1	0			
B0020R	C043	1	0	B001		0			
B0030R	C043	1	0	C044	1	0	C045	1	0
A001AND	C043	1	0	C044	1	0			
A002AND	C043	1	0	C045	1	0			
A003AND	C044	1	0	C045	1	0			
A004AND	C041	1	0	C042	1	0			
ROW3AND	B001		0	A001		0			
ROW3AND	B001		0	A002		0			
ROW3AND	B002		0	A003		0			
ROW3AND	B003		0	A004		0			
ROW3MODL									
ROW5AND	SC02	1	0	C030	1	0			
TOP OR	ROW3		0	SC02	3	0	ROW5		0

SYSTEM: FEHLERBAUM NR.2

OCCUR.PROBAB. CALCULATED FROM MEAN VALUES OF TRANS.RATES 5.44978E-05

TRIALS	MEAN VALUE	2ND MOMENT	3RD MOMENT	4TH MOMENT	LIMIT 5 %	LIMIT 50 %	LIMIT 95 %	MIN	MAX
2000	5.3351E-05	1.3203E-09	1.0080E-13	1.7247E-17	1.6949E-05	4.3363E-05	1.2309E-04	4.9997E-06	3.4865E-04
2400	5.3231E-05	1.2949E-09	9.7427E-14	1.6363E-17	1.6949E-05	4.3409E-05	1.2278E-04	4.9997E-06	3.4865E-04
2800	5.3214E-05	1.2560E-09	9.2838E-14	1.5447E-17	1.7021E-05	4.4108E-05	1.2174E-04	4.9997E-06	3.4865E-04
3200	5.2870E-05	1.2274E-09	8.9515E-14	1.4796E-17	1.6902E-05	4.3929E-05	1.2037E-04	4.9997E-06	3.4865E-04
3600	5.2892E-05	1.2217E-09	8.8955E-14	1.4651E-17	1.7034E-05	4.4063E-05	1.2002E-04	4.9997E-06	3.4865E-04
4000	5.2891E-05	1.2437E-09	9.5776E-14	1.6755E-17	1.7124E-05	4.3757E-05	1.1969E-04	4.9997E-06	3.5310E-04

XM = MEAN VALUE OF OCC.PROB. = 5.28911E-05 SIGMA = STANDARD DEVIATION = 3.52661E-05

SKEWNESS = 2.18365E+00 KURTOSIS = 7.83206E+00 UNIMODALITY = 3.33434E-01

NY**2 = LN((SIGMA/XM)**2)+1) = 3.67817E-01 MY = LN(XM)-(NY**2)/2 = -1.00312E+01

MEDIAN:
X50 = 4.37568E-05
SQRT(X05*X95) = 4.52722E-05
EXP(MY) = 4.40060E-05

SPREAD FACTOR:
X95/X50 = 2.73527E+00
X50/X05 = 2.55522E+00
EXP(NY*1.645) = 2.71193E+00

OCC. PROB.		DENSITY	INTEGRAL
1.000E-06			
1.155E-06		0	0
1.334E-06		0	0
1.540E-06		0	0
1.778E-06		0	0
2.054E-06		0	0
2.371E-06		0	0
2.738E-06		0	0
3.162E-06		0	0
3.652E-06		0	0
4.217E-06		0	0
4.870E-06		0	0
5.623E-06	:	2	2
6.494E-06		0	2
7.499E-06	:	3	5
8.660E-06	*	7	12
1.000E-05	**	13	25
1.155E-05	***	23	48
1.334E-05	****	36	84
1.540E-05	********	56	140
1.778E-05	**********	95	235
2.053E-05	**********	148	383
2.371E-05	**********	202	585
2.738E-05	**********	270	855
3.162E-05	**********	333	1188
3.652E-05	**********	346	1534
4.217E-05	**********	348	1882
4.870E-05	**********	397	2279
5.623E-05	**********	357	2636
6.494E-05	**********	336	2972
7.499E-05	**********	290	3262
8.659E-05	**********	215	3477
1.000E-04	**********	184	3661
1.155E-04	**********	107	3768
1.333E-04	**********	85	3853
1.540E-04	********	63	3916
1.778E-04	****	32	3948
2.053E-04	****	29	3977
2.371E-04	**	13	3990
2.738E-04	*	5	3995
3.162E-04	:	3	3998
3.652E-04	:	2	4000

LEAST SQUARE FIT, AVERAGE DEVIATION OF NORMALIZED FUNCTION = 0.2788E-01

NY**2 = 3.58567E-01 MY =-1.00387E+01
SPREAD FACT. = 2.67790E+00 MEDIAN = 4.36773E-05 CALCULATED MEAN VALUE XM = 5.22538E-05