KfK 3855
Februar 1985

MUSTAFA und MUSTAMO, Rechenprogramme zur Analyse von Fehlerbäumen mit Multistate-Komponenten

A. Wickenhäuser
Institut für Reaktorentwicklung
Projekt Scheller Brüter

Kernforschungszentrum Karlsruhe

KERNFORSCHUNGSZENTRUM KARLSRUHE
Institut für Reaktorentwicklung
Projekt Schneller Brüter

KfK 3855

MUSTAFA und MUSTAMO, Rechenprogramme zur Analyse von Fehlerbäumen mit Multistate-Komponenten

A. Wickenhäuser

Kernforschungszentrum Karlsruhe GmbH., Karlsruhe

Kernforschungszentrum Karlsruhe GmbH
ISSN 0303-4003

Zusammenfassung

Die beiden Rechenprogramme MUSTAFA und MUSTAMO bilden das Ergebnis und die Anwendung der in den vergangenen Jahren von L. Caldarola entwickelten Methoden zur Fehlerbaumanalyse. Mit den beiden Programmen können die wesentlichen Schwierigkeiten der analytischen Methode, nämlich mangelnde Flexibilität und Scheitern an den Problemen der Vermaschung, überwunden werden.
Ein wesentliches Hilfsmittel ist dabei die Boole'sche Algebra mit beschränkten Variablen, die es ermöglicht auf mehreren Ebenen Komponenten bzw. Superkomponenten mit mehr als zwei Zuständen zu behandeln.
Im einzelnen enthält das Programm MUSTAFA die Möglichkeit:

- Komponenten mit mehr als zwei Zuständen zu behandeln,
- auf verschiedene Weise statistische Abhängigkeiten zu modellieren,
- eine inkohärente Boole'sche Funktion zu erkennen und auf eine kleinere Basis zu reduzieren,
- eine Boole'sche Funktion zum Zweck der exakten Berechnung der Wahrscheinlichkeit zu erweitern,
- Übergangsraten zwischen Systemzuständen zu berechnen.

Das Programm MUSTAMO kann in Ergänzung hierzu umfangreiche sowie stark vermaschte Fehlerbäume in kleinere Einheiten unterteilen:
- durch horizontale Unterteilung in Module und Superkomponenten,
- durch vertikale Unterteilung mit Hilfe des Splitting-Verfahrens.

Der Bericht enthält:
- Erläuterungen zu den verwendeten Methoden
- die Beschreibung der Anwendung der Methoden mit Hilfe der Optionen der Programme
- Anwendungsbeispiele.

Abstract

MUSTAFA and MUSTAMO, computer programs for analysis of fault trees with multistate-components

The two codes MUSTAFA and MUSTAMO are result and application of the methods for fault tree analysis newly developed by Caldarola during the last years. The two programmes overcome the basic difficulties of the analytical methods: the limited flexibility and difficulties with the handling of large interconnected systems.

Basis of the new approach is the Boolean Algebra with restricted variables which allows for to handle components or supercomponents with more than two states.
The programme MUSTAFA provides the following capabilities:

- handling of components with more than two states
- modelling of statistical dependencies in various ways
- identification of an incoherent Boolean function and its reduction to a smaller basis
- extension of a Boolean function for exact calculation of probabilities
- calculation of transfer rates between different states of the system

In addition to these capabilities MUSTAMO is able to subdivide large and strongly interconnected fault trees into smaller subsystems:
- either by horizontal subdivision into modules and supercomponents
- or by vertical subdivision via the splitting procedure

The report consists of the following parts:
- explanation of the methods used
- description of the application using different options within the code structure
- sample cases

Inhalt Seite

Einleitung

Mit den beiden Rechenprogrammen MUSTAFA und MUSTAMO wurde die von L. Caldarola entwickelte Methode der Fehlerbaumanalyse realisiert. Das Ziel dieser Methodenentwicklung war es, die Vorteile der analytischen Behandlung von Fehlerbäumen aufzuzeigen und die bei der Analyse auftretenden Schwierigkeiten zu beseitigen.

Analytische Methoden eröffnen die Möglichkeit, die Strukturfunktion vollständig zu berechnen und aus der darauf basierenden Wahrscheinlichkeitsrechunung den exakten Zahlenwert der Eintrittswahrscheinlichkeit eines Top-Ereignisses zu ermitteln.

Die Schwierigkeiten der Analyse, die dazu führten, daß die Methode nur eingeschränkt angewandt werden konnte, sind hauptsächlich

- mangelnde Flexibilität, z.B. bei Vorhandensein statistischer Abhängigkeiten,

- Ungewißheit, ob große, sowie stark vermaschte Fehlerbäume innerhalb der von Speicherplatzkapazität und Rechenzeit gegebenen Grenzen gelöst werden können.

Bei der Überwindung der genannten Schwierigkeiten hat sich besonders die von Caldarola entwickelte Boole'sche Algebra mit beschränkten Variablen (Boolean Algebra With Restricted Variables, BAWRV) als geeignetes Werkzeug erwiesen um sog. Multi-State Probleme zu lösen, zu deren Behandlung andernorts Versuche mit komplizierter mehrwertiger Logik gemacht wurden. Die BAWRV erlaubt es, auf mehreren Ebenen Komponenten bzw. Superkomponenten mit mehr als 2 Zuständen zu bilden unter Verwendung von Variablen, die unter Beachtung nur weniger Einschränkungen der gewöhnlichen binären Boole'schen Algebra unterliegen.

Auch die Namensgebung für die beide Programme ist von dieser wesentlichen Eigenschaft, mit Hilfe der BAWRV multi-state Probleme zu behandeln, bestimmt; es bedeutet:

MUSTAFA (Multi-State Fault-Tree Analysis)
MUSTAMO (Multi-State Modularisation).

Das Programm MUSTAFA entwickelte sich zu einem Instrument zur möglichst vielseitigen und vollständigen Analyse von Systemen, die mit Fehlerbäumen dargestellt werden können. Es enthält die Möglichkeit

- Komponenten mit mehr als zwei Zuständen zu behandeln

- auf verschiedene Weise statistische Abhängigkeit zu modellieren

- eine inkohärente Boole'sche Funktion zu erkennen und auf eine kürzere Basis zu reduzieren

- eine Boole'sche Funktion so umzuformen, daß der exakte Wert der Wahrscheinlichkeit berechnet werden kann

- Übergangsraten zwischen mehreren, mit je einem Fehlerbaum dargestellten Zuständen eines Systems zu berechnen.

Das Programm MUSTAMO kann in Ergänzung hierzu sehr umfangreiche sowie stark vermaschte Fehlerbäume durch verschiedene Methoden horizontal und vertikal in kleinere Einheiten aufteilen. Das in Form von Modulen vorliegende Gesamtergebnis kann in der Regel in einem übersichtlichen Blockdiagramm dargestellt werden.

Eine optimale Anwendung der Methoden zur Unterteilung großer Fehlerbäume erfordert Vorbereitung; diese besteht in der sorgfältigen Analyse der Fehlerbaumstruktur, um die kritischen Vermaschung verursachende Knoten aufzufinden und geeignete Schnittstellen festzulegen. Der dazu erforderliche Überblick über den Fehlerbaum kann meist nur anhand einer übersichtlichen graphischen Darstellung gewonnen werden. Es wurde aber auch ein Rechenprogramm erstellt, das die Fehlerbaumstrukturen automatisch analysiert, die kritischen Knoten sowie alle möglichen Schnittstellen auflistet. Mit Hilfe dieses Vorschaltprogrammes kann der Aufwand zur Vorbereitung eines Rechenlaufs des Programms MUSTAMO weitgehend verringert werden. Eine Beschreibung dieses Programms wird in nächster Zeit erscheinen.

Der vorliegende Bericht ist eine Zusammenfassung und Erweiterung der Primärberichte /13/, /14/. Er enthält alle in der Zwischenzeit vorgenommenen Korrekturen, Erweiterungen und Ergänzungen der Programme. Er ist vor allem für Anwender der Programme gedacht. Daher wurden Angaben über die Interna der Programme (Aufbau, Datenorganisation etc.) nicht aufgenommen, dafür aber eine ausführlichere Dokumentation von Anwendungsbeispielen.

1. Definitionen und Erläuterung zu den verwendeten Symbolen

Ein Fehlerbaum ist ein gerichteter Graph ohne Schleifen, seine Knoten sind Primärknoten und Gatter. Die Wege (Kanten) führen von den Primärknoten zu einem Gatter an der Spitze, dem Top-Ereignis. Ein Fehlerbaum dient dazu, den Zustand eines technischen Sytems (Top- Funktion) als Funktion der an den Primärknoten definierten Zustände von Komponenten darzustellen. Der Zustand einer Komponente wird mit einer Boole'schen Variablen bezeichnet, deren zwei Werte 0,1 dem Eintritt bzw. Nicht-Eintritt des Zustandes entsprechen. Aufgrund der (im folgenden Abschnitt erläuterten) Boole'schen Algebra mit beschränkten Variablen wird jeder Zustand einer Komponente eines Systems mit je einer Variablen bezeichnet; eine Komponente umfasst in dieser Darstellungsweise alle Variablen, die einen Zustand dieser Komponente beschreiben, und zwischen denen Einschränkungen bestehen. Von den Primärknoten führen Wege (Kanten) zum Top über Knoten, die Gatter genannt werden. Gatter haben in der Regel mehrere Eingänge, d.h. Wege von Vorgängern; sie werden nach dem Booleschen Operator mit dem ihre unmittelbaren Vorgänger verknüpft sind, Und-Gatter bzw. Oder-Gatter genannt.

Der Vereinfachung halber können im Fehlerbaum auch Verknüpfungen, die nach den Regeln der Kombinatorik erfolgen, als solche im Fehlerbaum gekennzeichnet und in die Programme eingegeben werden; es sind dies M-aus-N-Systeme, in denen die möglichen Kombinationen mit $\vee$ (oder), jeweils M Vorgängen innerhalb der Kombinationen mit $\wedge$ (und) verknüpft sind.

Komponenten, deren Übergang von einem Zustand in einen anderen, vom Zustand anderer Komponenten abhängig sind, werden als abhängige Komponenten besonders deklariert. Es existiert im Programm MUSTAFA eine besondere Methode Abhängigkeiten darzustellen. Bei gegenseitiger Abhängigkeit wird der von Bedingungen abhängige Übergang einer von mehreren Komponenten zugeschrieben, während die Bedingungen, von denen der Übergang abhängig ist, von den

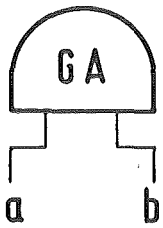
anderen Komponenten definiert werden. Die letzteren werden nach einer von Caldarola eingeführten Bezeichnung /6/ privilegierte Komponenten genannt, da sie in der Rechnung wie unabhängige Komponenten mit in- variantem Übergang von einem ihrer Zustände zum andern behandelt werden können. Die als abhängig deklarierte Komponente wird 'nicht-privilegierte' Komponente genannt, die Wahrscheinlichkeit der Übergänge von Zustand zu Zustand kann unter bestimmten Bedingungen variieren.

Die in Abb. 1 gezeigten Symbole entsprechen den in den Veröffentlichungen von Caldarola verwendeten; diese zum großen Teil auch in den USA gebräuchlichen Symbole enthalten deutliche Unterscheidungsmerkmale und sind daher für eine übersichtliche Darstellung gut geeignet. Boolesche Funktionen werden im folgenden Text an einigen Stellen durch 'Tableaus' dargestellt. In diesen Tableaus, die eine Ähnlichkeit mit Matrizen haben, sind die Variablen innerhalb einer Zeile mit $\wedge$ (und) verknüpft. Spalten werden nicht berücksichtigt; die ganzen Zeilen sind untereinander mit $\vee$ (oder) verknüpft. Jede Zeile eines Tableaus entspricht daher einem Monom einer mit disjunkten Termen dargestellten Booleschen Funktion.

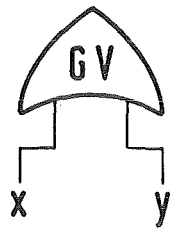
Bei der Wiedergabe Boole'scher Funktionen werden der leichteren und übersichtlicheren Darstellung wegen an mancher Stelle dieses Berichtes die Verknüpfung $\wedge$ (und) mit (-) und die Verknüpfung V (oder) mit (+) bezeichnet.

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

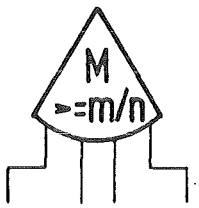
d_j = Primärvariable, einer als abhängig deklarierten Komponente d



UND-Gatter $GA = a \wedge b$



ODER-Gatter $GV = x \vee y$



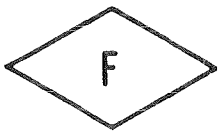
M, System aus n:

$> m/n$, mindestens $\left(\frac{m}{n}\right)$ Vorgänger

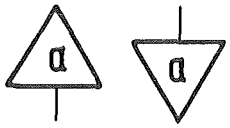
$= m/n$, exakt $\left(\frac{m}{n}\right)$ Vorgänger

$< m/n$, nicht mehr als $\left(\frac{m}{n}\right)$ Vorgänger

In den Abbildungen derFehlerbäume können zur Vereinfachung auch die folgenden Symbole erscheinen:



F = unaufgelöster Unterbaum



Anschlußstellen bei nicht durchgezogenen Kanten.

Abb. 1: Erläuterung der verwendeten Symbole

2. Angewandte Methoden und Algorithmen im Boole'schen Teil der Berechnung

2.1 Die Boole'sche Algebra mit beschränkten Variablen (BAWRV)

Die von Caldarola eingeführte Bezeichnung 'Boolean Algebra with restricted Variables' (BAWRV) wird in allen seinen Veröffentlichungen in deutscher Sprache mit Boole'sche Algebra mit beschränkten Variablen' wiedergegeben. Der Ausdruck 'beschränkt' sagt aus, daß die innerhalb dieser Algebra verwendeten Variablen Beschränkungen unterliegen.

Die Beschränkungen ergeben sich physikalisch aus der Tatsache, daß eine Komponente oder ein Untersystem eines größeren technischen Systems sich immer nur in einem Zustand befinden kann. Das charakteristische bei der Anwendung dieser Algebra ist es, daß nicht, wie bisher üblich, eine Komponente c mit einer einzigen Boole'schen Variablen definiert wird, deren beiden Werte c bzw. $\bar{c}$ den beiden Zuständen c = ausgefalln bzw. $\bar{c}$ = intakt entsprechen, bezeichnet wird, sondern, daß für jeden Zustand eine Boole'sche Variable definiert wird:

c_o = Komponente c intakt
c_1 = Komponente c ausgefallen.

Dabei ergeben sich die Einschränkungen $c_o \wedge c_1 = 0$;
$c_o \vee c_1 = 1$ und die Regel für die Negation $\bar{c}_o = c_1$;
$\bar{c}_1 = c_o$.

Für eine binäre Komponente erscheint die Definition je einer Variablen für einen Zustand als überflüssige Komplikation, die Anwendung auf Komponenten mit mehr als zwei Zuständen jedoch (Multi-state-components) zeigt den Vorteil dieses Vorgehens. So hat z.B. ein Schalter die 3 Zustände

SW_0 = Schalter intakt
SW_1 = offen ausgefallen
SW_2 = geschlossen ausgefallen;

eine Komponente K mit mehreren Ausfallarten kann die drei Zustände haben

K_0 = Komponente intakt
K_1 = Komponente teilweise ausgefallen
K_2 = Totalausfall der Komponente.

Enthält ein Fehlerbaum solche Multistate-Komponenten, dann muß vor Beginn der Rechnung bekannt sein,

- welche Variable zu ein- und derselben Komponente gehören,
- wieviele Zustände jede Komponente hat.

Es hat sich als übersichtliche und eindeutige Schreibweise herausgestellt, die Komponenten mit einem Buchstaben, bzw. mit einer Buchstabenkombination zu bezeichnen und daran eine Zahl für die Identifikation des Zustandes anzuhängen. Nach der in diesem Bericht und in den Programmen gültigen Konvention wird der intakte Zustand mit 0, die Ausfallzustände mit 1,2,...n-1 angegeben. In früheren Veröffentlichungen von Caldarola wird teilweise eine Numerierung der Zustände verwendet, die mit 1 beginnt und mit der jeweils n-ten Variablen den intakten Zustand der Komponente bezeichnet. Binäre Komponenten werden zwecks einheitlicher Verarbeitung in den Algorithmen der Programme ebenfalls nach dieser Konvention dargestellt. Bei der Eingabe in die Programme muß für die Komponente eine Buchstabenkombination und die Anzahl ihrer Zustände eingegeben werden. Für jeden der n Zustände einer Komponente wird vom Programm eine Variable definiert und die entsprechenden Speicherplätze belegt.

Die Beschränkungen zwischen den Variablen einer Komponente lassen sich in allgemeiner Form, wie folgt, darstellen:

Beschränkung 1. Art $C_j \wedge C_K = 0$ für alle $j \neq K$

Beschränkung 2. Art $\bigvee_{j=0}^{n-1} C_j = 1$

Aus diesen Beschränkungen folgt die Regel der Negation:

$$C_j = \bigvee_{\substack{K=0 \\ K \neq j}}^{n-1} C_K$$

Im übrigen gelten die Axiome und Regeln der Boole'schen Algebra uneingeschränkt.
Die Beschränkungen können auch als vereinfachte Durchführung eines Algorithmus aufgefaßt werden bei dem jeder Term einer Boole'schen Funktion mit einer Filterfunktion (im Booleschen Sinne) multipliziert wird, die aus den Mintermen der Variablen der Komponenten gebildet wird. Diese Minterme nehmen den Wert 0 oder 1 an; im folgenden Beispiel hat die Komponente C die drei Zustände C_0, C_1, C_2. Die aus den Mintermen gebildete Filterfunktion lautet

$$\begin{aligned}
\overline{C_0} \wedge \overline{C_1} \wedge \overline{C_2} &= 0\\
\overline{C_0} \wedge C_1 \wedge \overline{C_2} &= 1\\
\overline{C_0} \wedge C_1 \wedge C_2 &= 1\\
C_0 \wedge \overline{C_1} \wedge \overline{C_2} &= 1\\
\overline{C_0} \wedge C_1 \wedge C_2 &= 0\\
C_0 \wedge \overline{C_1} \wedge C_2 &= 0\\
C_0 \wedge C_1 \wedge \overline{C_2} &= 0\\
C_0 \wedge C_1 \wedge C_2 &= 0
\end{aligned}$$

Die BAWRV läßt sich auf höherer Ebene anwenden, um mehrere Komponenten zu einer Superkomponente zusammenzufassen. So kann aus den binären Komponenten a,b,c eine Superkomponente S mit 8 Zuständen gebildet werden; jeder Zustand von S entspricht einem Minterm der aus den drei Komponenten gebildeten Funktion.

$$\begin{aligned}
S_0 &= \overline{a} \wedge \overline{b} \wedge \overline{c}\\
S_1 &= a \wedge \overline{b} \wedge \overline{c}\\
S_2 &= \overline{a} \wedge b \wedge \overline{c}\\
S_3 &= \overline{a} \wedge \overline{b} \wedge c\\
S_4 &= a \wedge b \wedge \overline{c}\\
S_5 &= a \wedge \overline{b} \wedge c\\
S_6 &= \overline{a} \wedge b \wedge c\\
S_7 &= a \wedge b \wedge c
\end{aligned}$$

Zwischen allen Variablen S_j der Superkomponente S gelten die Einschränkungen der BAWRV, ebenfalls zwischen den Variablen einer Superkomponente Z, deren vier Zustände durch Kondensation einiger Zustände von S gebildet wurden:

$Z_0 = S_0$ System intakt
$Z_1 = S_1 + S_2 + S_3$ genau eine von drei Komponenten ausgefallen
$Z_2 = S_4 + S_5 + S_6$ genau zwei von drei Komponenten ausgefallen
$Z_3 = S_7$ alle drei Komponenten ausgefallen.

2.2 Reduzierung der Boole'schen Strukturfunktion

Eine Boole'sche Strukturfunktion kann durch ihre Primimplikanten dargestellt werden. Die (im Boole'schen Sinne) Summe der notwendigen Primimplikanten wird Basis genannt. Kohärente Systeme haben die Eigenschaft, daß die Minimalschnitte der Funktion zugleich die Primimplikanten sind und nur eine einzige Basis existiert. Im allgemeinen sind Fehlerbäume, von Systemen, an die Sicherheitsanforderungen gestellt werden, kohärent, da inkohärente Systeme sicherheitstechnisch nicht sinnvoll wären. Ein kohärentes System ist intakt, wenn alle seine Komponenten intakt sind und ausgefallen, wenn sich alle seine Komponenten in einem Ausfallzustand befinden. Der Übergang einer Komponente vom intakten Zustand zu einem Ausfallzustand bedeutet daher eine Veschlechterung des Systems, umgekehrt der Übergang einer Komponente von einem Ausfallzustand zum intakten Zustand eine Verbesserung des Systems. Inkohärent ist ein System, bei dem die zuletzt genannte Regel nicht zutrifft. Caldarola hat gezeigt /5/, /6/, daß im Gegensatz zu Systemen mit ausschließlich binären Komponenten Systeme mit Multi-State-Komponenten zwar kohärente Funktionen für das Ausfallverhalten haben können, daß aber die Komplementfunktionen dazu nicht notwendigerweise der Symmetrie unterliegen, also inkohärent sein können.

Im Programm MUSTAFA können alle Systemzustände berechnet werden, also auch solche, die inkohäret sind, deswegen wurde bei der Entwicklung dieses Programmes der Frage nach der Kohärenz Bedeutung beigemessen.

Funktionen kohärenter Systeme besitzen nur eine einzige Basis, deren Primimplikanten mit den Minimalschnitten identisch sind. Funktionen inkohärenter Systeme besitzen eine sog. vollständige Basis; diese besteht aus der Gesamtzahl aller Primimplikanten, die die Funktion implizieren. Diese Prim-

implikanten sind entweder obligatorisch, nichtobligatorisch oder redundant. Es läßt sich also u.U. eine wesentlich kürzere Basis (d.h. eine Basis mit einer geringeren Anzahl an Primimplikanten) bilden, die sämtliche obligatorische und eine Teilmenge nichtobligatorischer Primimplikanten enthält.

Wenn nach der Auflösung eines Fehlerbaums die durch den Auflösungsalgorithmus berechneten Terme vorliegen, ergeben sich folgende Fragen:

- Ist die vorliegende Funktion kohärent oder nicht?
- Sind (im Falle der Kohärenz) alle vorliegenden Terme Minimalschnitte, oder gibt es Terme die zu Minimalschnitten reduziert werden können?
- Welches ist (im Falle der Inkohärenz) die vollständige Basis?
- Existiert eine wesentlich kürzere Basis?
- Welches ist die kürzeste Basis?

Die beiden ersten Fragen werden in einem Unterprogramm PREPAR innerhalb des Programmsystems MUSTAFA nach Durchführung eines von Caldarola entwickelten Algorithmus behandelt, /4/.

Im Falle der Inkohärenz wird die vollständige Basis mit Hilfe des Nelson-Algorithmus berechnet. Dieser besteht in der doppelten Negation einer Funktion. $G=\overline{F}$; $F_{tot} = \overline{G}$, wobei F_{tot}, die vollständige Basis, äquivalent zu F ist. Anschließend werden mit einem Algorithmus, den Caldarola /4/ beschrieben hat, die obligatorischen Primimplikanten ermittelt. Ist die Zahl der obligatorischen Primimplikanten wesentlich kleiner als die Anzahl aller Primimplikanten in der vollständigen Basis, dann erscheint es lohnenswert, eine kürzere oder gar die kürzeste Basis zu berechnen. Der Algorithmus zur Berechnung der kürzesten Basis wurde von Caldarola entworfen und konnte vom Verfasser beim Programmieren stark vereinfacht werden.

Die oben aufgeführten Fragen bilden einen Teil der Programmlogik, die mit den gegebenen Größen α, β, γ gesteuert werden kann.
Dabei bedeutet

α = dimensionslose Zahl<1 zum Vergleich mit dem Anteil an Komponenten, die mit allen ihren Zuständen als Primärknoten des Fehlerbaums erscheinen.

β = dimensionslose Zahl < 1 zum Vergleich mit dem Verhältnis der Größe der vollständigen Basis zu einer errechneten Basis

γ = dimensionslose Zahl < 1 zum Vergleich mit dem Verhältnis der Anzahl von obligatorischen Primimplikanten zur Anzahl der Primimplikanten in der vollständigen Basis.

Zu α) = Eine Funktion, die nur jeweils einen Zustand einer Komponente oder nur die Ausfallzustände der Komponenten enthält, ist mit Sicherheit kohärent.

Zu β) = Ist die nach Auflösung des Fehlerbaums vorliegende Funktion wesentlich kürzer als die vollständige Basis, dann ist die Suche nach einer kurzen Basis überflüssig.

Zu γ) = Ist die große Mehrzahl der Primimplikanten obligatorisch, dann kann keine wesenlich kürzere Basis existieren, da obligatorische Primimplikanten in jeder Basis enthalten sein müssen.

2.3 Äquivalente Erweiterungen derBoole'schen Strukturfunktion

Die bisher beschriebenen Umformungen dienten der Reduzierung von Funktionen bzw. Termen. Es kann aber auch notwendig werden, die Funktionen in eine erweiterte Form zu bringen, und zwar dann, wenn auf ihr aufbauend der exakte Zahlenwert der Wahrscheinlichkeit berechnet werden soll.

Angenommen eine Funktion F besteht aus den beiden Minimalschnitten a und b: $F = a + b$. Bei der Berechnung der Wahrscheinlichkeit als Summe der Wahrscheinlickeiten der Minimalschnitte erhält man $E\{F\}$ $E\{a\} + E\{b\}$. Der exakte Wert ist aber $E\{F\} = E\{a\} + E\{b\} - E\{a\} \cdot E\{b\}$.

Erweitert man $F = a \vee (b \wedge a) \vee (b \wedge \bar{a}) = a \vee (b \wedge \bar{a})$ und bildet die Summe der Wahrscheinlichkeiten der beiden Terme, dann erhält man $E\{F\} = E\{a\} + E\{b\} \cdot E\{\bar{a}\}$, dies entspricht der obigen Formel wegen der Beziehung $E\{\bar{a}\} = 1 - E\{a\}$.

Beziehung $E\{a\} = 1-E\{a\}$.
Im Programm MUSTAFA sind zwei Algorithmen für eine dementsprechende Erweiterung vorhanden:

- der Bennet-Algorithmus /11/
- der Keystone-Algorithmus /10/

Der Bennet-Algorithmus führt die im obigen Beispiel für zwei aus je einer Variablen bestehenden Minimalschnitte gezeigte Rechnung für eine ganze Funktion aus, wobei der kleinste Aufwand dann zu erreichen ist, wenn die kleinsten Terme (Terme mit der geringsten Anzahl darin enthaltener Variablen) unverändert bleiben und die größeren mit den Komplementzuständen der nicht gemeinsamen Variablen erweitert werden. Dieser Algorithmus wird auf die in 4.3 beschriebenen Inhibitorfunktionen automatisch angewendet.
Der Keystone-Algorithmus zerlegt die Funktion in einander ausschließende Teilfunktionen. Jede dieser Teilfunktionen besteht aus je einem sogenannten Keystone-Monom und einer mit dem Keystone-Monom verbundenen Simple Funktion, einer einfachen Funktion.

Die Keystone-Monome schließen alle einander aus, während die einfachen Funktionen die Eigenschaft haben, daß alle in ihr enthaltenen Variablen nur ein einziges Mal in ihr auftreten. Die Wahrscheinlichkeit einer einfachen Funktion kann daher mit einer vereinfachten Formel gerechnet werden.

Eine Funktion F = $a \cdot b \cdot c$
$a \cdot b \cdot d$
$a \cdot c \cdot d$
$b \cdot c \cdot d$

würde durch den Keystone Algorithmus in folgende Teilfunktionen zerlegt werden:

Nr.	Keystone-Mon.	einfache Funktionen
1)	$d \cdot a$	b
		c
2)	$d \cdot \overline{a}$	$b \cdot c$
3)	$\overline{d} \cdot a$	$b \cdot c$

4) $\overline{d} \cdot \overline{a} = 0$

Die Auswahl der Komponenten d und a erfolgte aufgrund der Häufigkeit des Auftretens in der Funktion. Der Algorithmus ist sehr effizient, die umgeformte Funktion ist weniger umfangreich als beim Bennet-Algorithmus, sie beansprucht wenig Rechenzeit und lieferte in Testrechnungen den exakten Zahlenwert der Wahrscheinlichkeit, auch wenn dieser sehr nahe bei 1 lag.

3. Methoden zur Unterteilung und Entmaschung umfangreicher und stark vermaschter Fehlerbäume

3.1 Modulbildung, einfache Module

Module sind Teile einer Boole'schen Strukturfunktion; wenn sie unverändert an mehreren Stellen in der Gesamtfunktion enthalten sind, können sie zur Vereinfachung durch eine einzige Variable ersetzt werden. Ihre interne Zusammensetzung muß dann nur einmal explicit angegeben werden. So kann z.B. die Funktion eines Unterbaums gesondert berechnet und als Modul betrachtet werden. In der Berechnung der Top-Funktion wird dieser Unterbaum wie eine Komponente behandelt, (Super-Komponente).

Dieses Vorgehen ist aber nur möglich, wenn der Modul logisch unabhängig ist. Logisch unabhängig ist ein Modul nur dann, wenn in ihm ausschließlich Variable von solchen Komponenten enthalten sind, deren Variable nirgendwo anders in der übrigen Topfunktion enthalten sind. Ist dies der Fall, dann ist nicht auszuschließen, daß bei Einsetzung des Moduls mit seiner expliziten Zusammensetzung das Idempotenzgesetz oder die Regel des gegenseitigen Ausschlusses angewendet werden müßte.

Zur exakten Definition hat Caldarola die Begriffe Route und Territorium benutzt /8/. Eine Route ergibt sich, wenn man von der Spitze eines Fehlerbaums oder eines Unterbaums jeweils einer der vorhandenen Kanten bis zu einer Primärvariablen folgt. Die Route ist definiert durch die bei einem solchen Weg durchlaufenen Gatter. Am Ende befindet sich eine Primärvariable. Bildet man von einem Gatter G aus alle möglichen Routen, so bilden die Liste der dabei erreichten Primärvariablen das sogenannte Territorium des Gatters G.

Zur Prüfung der logischen Unabhängigkeit eines Gatters G wird das Territorium dieses Gatters, (internes Territorium), mit dem Territorium der Spitze (des Top) des Fehlerbaums, dem sogenannten externen Territorium verglichen, wobei im externen Territorium alle Routen, die über das Gatter G laufen, nicht berücksichtigt werden. Enthalten die beiden Territorien keine gemeinsame Komponente, so ist das Gatter G unabhängig, seine Funktion kann gesondert berechnet werden, das Gatter G kann mit den zwei Zuständen (G0, G1) bei der Berechnung der Top-Funktion wie eine Komponente behandelt werden; seine Funktion ist ein unabhängiger Modul.

Im Programm MUSTAMO ist diese Abtrennung von Modulen als Option Nr. 1 vorgesehen. Gatter, deren Funktion als Modul abgetrennt werden sollen, muß der Benutzer eingeben; eine Prüfung der logischen Unabhängigkeit durch Vergleich des internen mit dem externen Territorium wird vom Programm vorgenommen.

3.2 Modulbildung durch Ausklammern von Schlüsselkomponenten

Es ist möglich, aus einer Anzahl von Monomen eine gemeinsame Variable auszuklammern; dies kann geschehen, um eine Strukturfunktion der Übersichtlichkeit wegen aufzugliedern. Der Teil der Funktion aus dem eine gemeinsame Variable ausgeklammert wurde, kann als Modul bezeichnet werden. Die Komponenten, deren Variable ausgeklammert werden sollen, werden hier Schlüsselkomponeten genannt.

Dieses Ausklammern kann auch dazu verwendet werden um innerhalb der Boole'schen Funktion eines Unterbaumes zwischen logisch abhängigen und unabhängigen Teilen zu unterscheiden. Existieren in einem Fehlerbaum Komponenten, deren Variable sowohl im internen als auch im externen Territorium eines Gatters G auftreten, so sind diese Komponenten obligatorische Schlüsselkomponenten. Aus der Funktion des Gatters G werden die Variablen von Schlüsselkomponenten ausgeklammert. Dabei entstehen gemischte Terme die aus Variablen von Schlüsselkomponenten und Modulen bestehen. Das Programm MUSTAMO hat eine Option, bei der die in Minimalschnitte vorliegende Funktion eines Gatters durch Bildung solcher gemischter Terme teilweise modularisiert werden kann. Da die Anzahl der gemischten Terme wesentlich geringer ist als die Anzahl der Minimalschnitte, kann dieses Verfahren zur Veringerung der Rechenzeit beitragen.

Bei diesem Vorgehen werden alle von obligatorischen Schlüsselkomponenten verursachten logischen Abhängigkeiten korrekt behandelt. Da die Variablen dieser Komponenten explizit in den Termen enthalten sind, können alle notwendigen Operationen (Idempotenz-Regel, Regel des gegenseitigen Ausschlusses nach der BAWRV) durchgeführt werden.In den Modulen befinden sich alle die Teile der Unterfunktion, die von der übrigen Top-Funktion unabhängig sind. Kleinere Unkorrektheiten können aber entstehen, weil diese Module zwar vom externen Territorium unabhängig, aber in den meisten Fällen untereinander nicht unabhängig sind. So können redundante Terme nicht absorbiert werden, nachdem wegen Idempotenz der Variablen von Schlüsselkomponenten einige mit gemischten Termen zusammengesetzte Minimalschnitte des Fehlerbaums kürzer geworden sind.

Betrachtet man die an die Boole'sche Rechnung sich anschließende Wahrscheinlichkeitsrechnung, so stellt man fest, daß der errechnete Zahlenwert in jedem Falle nur erhöht sein kann, aber niemals zu klein, denn alle in Frage kommenden Idempotenzen wurden berücksichtigt; diese allein können einen Fehler nach unten verursachen; nicht beseitigte Redundanzen können das Ergebnis nur erhöhen. Werden mehrmalige Berechnungen eines Fehlerbaums mit je feinerer Auflösung der Unterbäume, d.h. mit je geringerer Anzahl an redundanten Termen durchgeführt, dann nähert man sich dem exakten Ergebnis von oben her, d.h. bei Zuverlässigkeitsuntersuchungen von der sicheren Seite her.
In einem gerechneten Beispiel betrug der Fehler einer vorläufigen Rechnung gegenüber der Rechnung mit feinerer Auflösung ungefähr + 7%.

3 3 Die Bildung von Superkomponenten mit mehr als zwei Zuständen

Ein Unterbaum, dessen Funktion völlig unabhängig ist, kann als Superkomponente mit zwei Zuständen betrachtet werden. Häufig kommt es vor, daß nicht nur ein, sondern mehrere Unterbäume, wenn man ihre internen Territorien betrachtet, zusammen vom externen Territorium unabhängig sind /8/. In diesem Falle macht es die BAWRV möglich, aus diesen Gattern eine Superkomponente mit mehr als zwei Zuständen zu bilden. So können drei Gatter G1, G2 und G3 zu einer Superkomponente SC auf folgende Weise zusammengefaßt werden:

$SC_0 = \overline{G1} \wedge \overline{G2} \wedge \overline{G3}$ $\quad$ $SC_4 = G1 \wedge G2 \wedge \overline{G3}$

$SC_1 = G1 \wedge \overline{G2} \wedge \overline{G3}$ $\quad$ $SC_5 = G1 \wedge \overline{G2} \wedge G3$

$SC_2 = \overline{G1} \wedge G2 \wedge \overline{G3}$ $\quad$ $SC_6 = \overline{G1} \wedge G2 \wedge G3$

$SC_3 = \overline{G1} \wedge \overline{G2} \wedge G3$ $\quad$ $SC_7 = G1 \wedge G2 \wedge G3$

Der Fehlerbaum wird auf diese Weise unterteilt (geschnitten), die Berechnung der einzelnen Zustände der Superkomponente erfolgt gesondert. In den oberen Teil des Fehlerbaums gehen für ein bisheriges Gatter G_j nach der Umformung alle Zustände der Superkomponente ein, in denen G_j nicht-negiert enthalten ist.

Es gibt im Programm MUSTAMO zwei Möglichkeiten, eine Superkomponente aus n Gattern ($2 \leq n \leq 4$) zu bilden.

- Der Benutzer benennt die Gatter, aus denen vom Programm automatisch eine Superkomponente mit 2^n Zuständen gebildet werden soll.

- Der Benutzer formt den Fehlerbaum vor Eingabe so um, daß so viele Gatter mit den entsprechenden Eingängen vorhanden sind, wie die Superkomponente Zustände hat.

Bei automatischem Bau einer Superkomponente werden die Gatter G_j (s. obige Formel) geschnitten. Die aus den Mintermen sich ergebenden Funktionen, sowie deren Wahrscheinlichkeiten werden berechnet und als SC_j bezeichnet. Von SC_0 (dem Intakt-Zustand) wird nur die Wahrscheinlichkeit als Differenz zwischen 1 und der Summe der Wahrscheinlichkeiten der Ausfallzustände berechnet. Die Gatter G_j werden zu Oder-Gattern umgeformt. Diese Oder-Gatter haben alle diejenigen Zustände der Superkomponente SC zu Vorgängern, in denen das ursprüngliche G_j nicht negiert enthalten ist.
Für das obige Beispiel werden folgende Umformungen vorgenommen:

G_1 (umgeformt) $= SC_1 + SC_4 + SC_5 + SC_7$

G_2 (umgeformt) $= SC_2 + SC_4 + SC_6 + SC_7$

G_3 (umgeformt) $= SC_3 + SC_5 + SC_6 + SC_7$

Bei Berechnung von Superkomponenten, deren Zustände vom Benutzer durch Umformung des Fehlerbaums selbst definiert wurden, muß der Schnitt vorher erfolgen. Ein Beispiel (Abb. 2) zeigt die vom Benutzer eingebauten Gatter

X,Y,Z. Diese Gatter werden in der Eingabe als Zustände einer Superkomponente deklariert und vom Programm dementsprechend als SC_1, SC_2, SC_3 definiert und berechnet.

Im gezeigten Falle war die Umformung des Fehlerbaums vorteilhaft, da bei automatischem Bau der Superkomponente vom Programm die Berechnungen

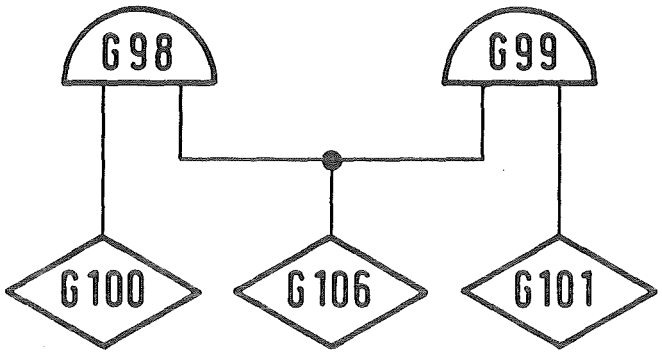
$$SC_1 = G98 \cdot \overline{G99} = (G100 \cdot G106) \cdot (\overline{G101} + \overline{G106})$$
$$SC_2 = \overline{G98} \cdot G99 = (G101 \cdot G106) \cdot (\overline{G100} + \overline{G106})$$
$$SC_3 = G98 \quad G99 = (G101 \cdot G106) \cdot (G102 + G106)$$

vorgenommen worden wären. Die Boole'schen Funktionen von G106 und G106 waren in dem ausgewählten Beispiel so umfangreich, daß die in den drei Zuständen überflüssigen Berechnungen $G106 \cdot \overline{G106} = 0$ und $G106 \cdot G106 = G106$ Rechenzeit gekostet hätten. Eine Vorüberlegung führte dann zu der Vereinfachung

$SC_1 = X = G100 \cdot G106 \cdot \overline{G101}$; $SC_2 = Y = G101 \cdot G106 \cdot \overline{G100}$
und $SC_3 = Z = G100 \cdot G101 \cdot G106$.

Durch Umformung des Fehlerbaums definierte Zustände von Superkomponenten sind ebenfalls geboten, wenn einzelne der 2^n möglichen Zustände in der Top-Funktion nicht existieren, oder wenn mehrere zu einem Zustand kondensiert werden können. Ein Beispiel hierfür findet sich in einem Anwendungsbeispiel, Abschnitt 7.

Die für die Zustände einer Superkomponente errechneten Funktionen enthalten Negationen, so daß nach (in der BAWRV notwendigen) Auflösung dieser Negationen auch die Komplementzustände zum Ausfall der Komponenten enthalten sind. Caldarola hat nachgewiesen /5/, daß für die Funktionen kohärenter Systeme, die den Systemausfall beschreiben, die sogenannten assozierten kohärenten Funktionen äquivalent sind. Die assozierte kohärente Funktion wird gebildet, in dem man die Variablen, die dem Intakt-Zustand der Komponente entsprechen, 1 setzt und danach die entstandenen Absorptionen berücksichtigt. Das Programm MUSTAMO nimmt diese Prozedur vor; die in die Top-Funktion eingesetzten assozierten kohärenten Funktionen der Zustände einer Superkomponenten werden durch diese Operation in Form der Minimalschnitte geliefert.



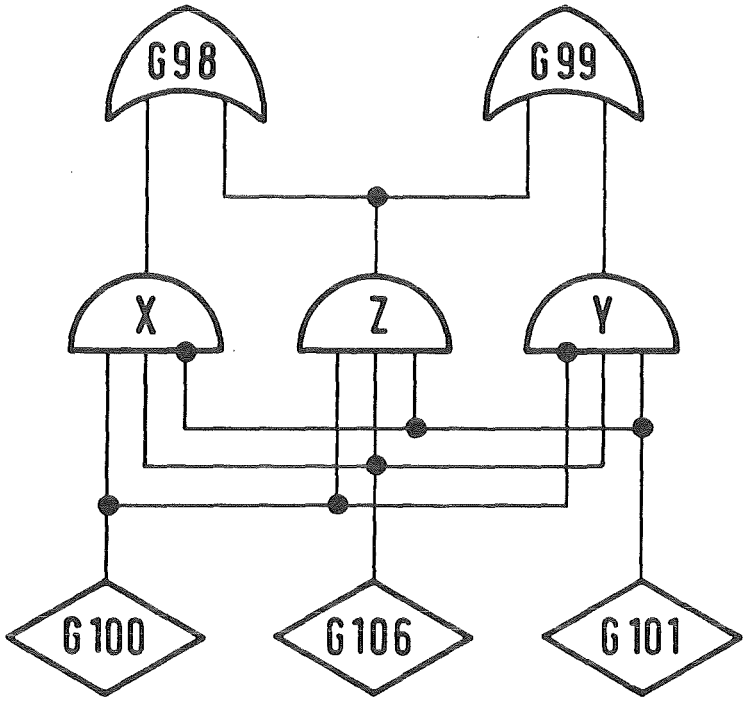


Abb. 2: Umformung zur Bildung einer Superkomponente.

3.4 Vorläufige Superkomponenten

Die in 3.2 und 3.3 beschriebenen Optionen können im Programm MUSTAMO zwecks schnellerer Auflösung stark vermaschter Fehlerbäume auch auf logisch abhängige Unterbäume angewendet werden. Bei der Auflösung des Fehlerbaums werden durch diese horizontalen Schnitte die Zahl der überflüssigen Auflösungswege stark reduziert. Die dabei berechneten Funktionen der vorläufigen Superkomponeten enthalten logische Abhängigkeiten. Aus diesem Grund werden sie entweder in ihrer expliziten Zusammensetzung (oder bei Superkomponenten mit mehr als zwei Zuständen wahlweise auch als gemischte Terme mit Schlüsselkomponenten und Modulen) auf eine externe Speichereinheit gespeichert. Am Ende, bei der Berechnung der Top-Funktion werden diese Funktionen wieder abgerufen und in die Top-Funktion eingesetzt, wobei Idempotenz innerhalb eines Monoms berücksichtigt und Absorption redundanter Terme vorgenommen wird.

Die Zahl der zeitraubenden Absorptionsabfragen kann bei diesem Verfahren stark reduziert werden.

3.5 Das Splitting-Verfahren mit Hilfe deterministischer Komponenten

Zur Ergänzung der bisher beschriebenen Methoden des horizontalen Schneidens von Fehlerbäumen wurde zur leichteren Berechnung bei Vermaschung das sogenannte Splitting-Verfahren in das Programm MUSTAMO übernommen /12/. Dieses Verfahren bedeutet vertikale Aufspaltung von Fehlerbäumen.

Eine Boole'sche Funktion F kann mit der Operation $G1 = F \wedge K$;
$G2 = F \wedge \overline{K}$ in die zwei sich ausschließende Teilfunktionen G1 und G2 gespalten werden. Nach den Gesetzen der BAWRV muß F mit allen Variablen einer Komponente K mit den Variablen K_j, $j = 0, n-1$ - logisch multipliziert werden.

Diese Aufspaltung kann vor Beginn der Berechnung geschehen, denn mit der Setzung $K_j = 1$ (deterministisch) und $K_j = 0$ kann der Fehlerbaum umgeformt werden. Bei dieser Umformung wird ein Oder-Gatter GV = X + Y mit einem deterministischen Vorgänger Y:

$GV = X + 0 = X$

$GV = X + 1 = 1$

Ein Und-Gatter GA = X Y wird

$GA = X \cdot 0 = 0$

$GA = X \cdot 1 = X$

Wird diese Methode auf eine logische Abhängigkeiten verursachende Komponente (obligatorische Schlüsselkomponente) angewendet, so verschwindet diese, da ihre Variablen deterministisch 1 bzw. 0 gesetzt wurden. Der Wert der Wahrscheinlichkeitsrechnung einer solchen abgespaltenen Teilrechnung muß mit der Wahrscheinlichkeit der 1 gesetzten Variablen multipliziert werden.

Durch das Verschwinden logischer Abhängigkeiten können bei dieser Methode die Optionen der horizontalen Unterteilung auf große Teile des Fehlerbaums angewendet werden.

Das Programm MUSTAMO hat zwei Möglichkeiten eine Komponente zur deterministischen zu deklarieren:

- Ein bestimmter Zustand einer Komponente wird als deterministisch 1 deklariert, die Wahrscheinlichkeitsdaten der betreffenden Komponente werden nicht ins Programm eingegeben.

- Eine Komponente wird zur deterministischen Komponente deklariert, das Programm führt mehrere Rechnungen aus, bei denen jeweils ein Zustand der Komponente 1, die übrigen 0 gesetzt werden. Das Ergebnis der Wahrscheinlichkeitsrechnung wird mit der Eintrittswahrscheinlichkeit des auf 1 gesetzten Zustands multipliziert.

Im ersten Falle wird der Fehlerbaum, wie bei gewöhnlicher Benutzung des Programmes nur einmal gerechnet, das Ergebnis der Wahrscheinlichkeitsrechnung muß in einer nachfolgenden Handrechnung mit der Wahrscheinlichkeit des auf 1 gesetzten Komponentenzustandes multipliziert werden.

Im zweiten Falle führt das Programm soviele Rechnungen durch, wie sich aus dem Produkt der Anzahl der Zustände der deterministischen Komponenten 2. Art ergibt. Die notwendige Multiplikation bei der Wahrscheinlichkeitsrechnung wird ebenfalls vom Programm durchgeführt.

Daß nach Einbau der zweiten Möglichkeit mit automatischer Spaltung die erste Möglichkeit einer einmaligen Rechnung im Programm belassen wurde, kann damit begründet werden, daß in vielen Fällen nicht alle theoretisch möglichenAufspaltungen gerechnet werden müssen; so ist es z.B. überflüssig bei der Konstellation zweier binärer Komponenten a und b, die nur in der Konjunktion $a_1 \wedge b_1$ im Fehlerbaum auftreten, alle 4 möglichen Aufspaltungen zu rechnen. Man wird in diesem Falle b_1 konstant 1 setzen (b = deterministische Komponente 1. Art) und mit der Komponente a (deterministisch 2. Art) die automatische Aufspaltung vom Programm vornehmen lassen, da das Produkt $a_1 \wedge b_1$ zu 0 wird, wenn nur eine der beiden Variablen 0 gesetzt wird. Wenn die Anzahl der als deterministisch zu behandelnden Komponenten relativ groß ist, können durch geschickte Anordnung deterministischer Komponenten erster und zweiter Art an mehreren Rechenläufen diejenigen Kombinationen weggelassen werden, deren Wahrscheinlichkeit (Produkt der Wahrscheinlichkeiten auf 1 gesetzten Komponentenzustände) so klein ist, daß sie für das Gesamtergebnis keinen merklichen Beitrag liefern.

3.6 Die Auflösung von Fehlerbaumeinheiten mit dem Top-down-Algorithmus

Ganze Fehlerbäume oder die durch horizontale Schnitte entstandenen Teilfehlerbäume werden hier als Einheiten verstanden, die in beiden Programmen mit Hilfe des Top-down-Algorithmus gelöst werden. Der Top-down-Algorithmus wurde zum ersten Male von Fussel in einem Fehlerbaumprogramm angewandt /9/. Der Top-down-Algorithmus ersetzt beginnend mit dem Gatter an der Spitze einer Einheit jedes Gatter durch seine Vorgänger, wobei die Vorgänger eines Und-Gatters beide in die betreffende Zeile der Tableaus eingetragen werden; die Auflösung eines Oder-Gatter bewirkt Verdoppelung der Zeile, wobei das aufgelöste Gatter jeweils durch einen seiner Vorgänger ersetzt wird.

In beiden Programmen wird der Algorithmus zeilenweise durchgeführt, d.h. angefangen bei der ersten Zeile werden in jeder Zeile alle Gatter aufgelöst, solange, bis nur noch Primärvariablen bzw. die Anfangsknoten einer Fehlerbaumeinheit enthalten sind. Der Vergleich der in den Zeilen enthaltenen Terme zur Feststellung von Absorptionen wird zwischen Termen mit Primärvariablen durchgeführt, d.h. nach Auflösung aller Gatter innerhalb einer Zeile.

Es existieren kritische Gatter; dies sind diejenigen, die mehr als einen Ausgang in Richtung zur Spitze haben. Bei Auflösung von der Spitze her können sie in ein- und demselben Term mehrmals erscheinen. Um das mehrfache Auflösen eines Unterbaums unterhalb eines kritischen Gatters innerhalb ein- und desselben Terms zu vermeiden, wurde eine Prioritätsregel eingeführt, nach der unkritische Gatter vorrangig aufgelöst werden; kritische Gatter werden zurückgestellt, so daß bei Zusammentreffen die Idempotenzregel angewendet werden kann. Auf diese Weise können zahlreiche Absorptionsabfragen durch eine einzige Operation ausgeschaltet werden. Mit dieser Prioritätsregel konnte der Top-down-Algorithmus erheblich beschleunigt werden.

Bei Auftreten mehrerer kritischer Gatter in ein- und demselben Term kann die automatisch vom Programm durchgeführte Prioritätsregel teilweise versagen. Daher wurde im Programm MUSTAMO die Möglichkeit eingeführt, kritische Gatter zu sog. Stop-Gattern zu deklarieren. Bei Eingabe von Stop-Gattern wird der Algorithmus in zwei Stufen durchgeführt. In der ersten Stufe werden die Stop-Gatter wie Primärvariabeln behandelt und erst in der zweiten Stufe erfolgt die vollständige Auflösung. Es gibt Fälle von starker Vermaschung, bei denen die Nennung von geeigneten Stop-Gattern die Rechenzeiten um Faktoren verringern kann. In der bisherigen Praxis konnte ein Fehlerbaum mit einem 3- aus 5-System an der Spitze nur mit Hilfe dieser zweistufigen Auflösung in einer vertretbaren Rechenzeit gelöst werden.

4. Die Wahrscheinlichkeitsrechnung in den Programmen MUSTAFA und MUSTAMO

4.1 Die Eintrittswahrscheinlichkeit der Komponentenzustände

Es ist in den Programmen MUSTAFA und MUSTAMO möglich, die zur Berechnung der Eintrittswahrscheinlichkeit notwendigen Daten auf zweierlei Weise einzugeben:

- Für jeden Zustand einer Komponente wird die Eintrittswahrscheinlichkeit direkt eingegeben.

- Für die Übergänge von einem Zustand i einer Komponente zu Zustand j werden Übergangsraten eingegeben.

(In einem besonderen Falle ist es notwendig, beides zusammen einzugeben.)

Die Übergangsraten $\lambda_{i \to j}$ von Zustand i zu Zustand j einer Komponente haben die Dimension $[1/t]$, t = Zeiteinheit.
Es sind dies

MTTF = meantime fo failure	Ausfallrate, Übergangsrate vom intakten Zustand zu einem Ausfallzustand j.
MTTR = meantime to repair	Reparaturrate, Übergangsrate von einem Ausfallzustand zum intakten Zustand bei sofort entdecktem Ausfall.
TI = inspection time	Regenerierungsrate einer in regelmäßigen Abständen inspizierte Komponente.

Aus den Übergangsraten werden in den Programmen asymptotische Werte für die Eintrittswahrscheinlichkeiten Pi nach dem Ansatz einer Differentialgleichung gerechnet:

$$\frac{dP_i}{dt} = \sum_{\substack{j=0 \\ j \neq i}}^{n-1} \lambda_{j \to i} \cdot P_j - P_i \cdot \sum_{\substack{j=0 \\ j \neq i}}^{n-1} \lambda_{i \to j}$$

Für eine binäre Komponente ergibt sich daraus:

$$P_0 = \frac{\lambda_{1 \to 0}}{\lambda_{1 \to 0} + \lambda_{0 \to 1}} \quad ; \quad P_1 = \frac{\lambda_{0 \to 1}}{\lambda_{1 \to 0} + \lambda_{0 \to 1}}$$

wobei $\lambda_{1 \to 0}$ = Reparaturrate, $\lambda_{0 \to 1}$ = Ausfallrate.

Ist das Zeitverhalten der Ausfallrate einer Komponente nicht exponentiell, wie im obigen Ansatz vorausgesetzt, empfiehlt es sich die zuvor errechnete Eintrittswahrscheinlichkeit direkt einzugeben.

Die Übergangsraten $\lambda_{i \to j}$ von Zustand i zu Zustand j einer Komponente haben die Dimension $[1/t]$, t = Zeiteinheit.
Es sind dies

MTTF = meantime fo failure	Ausfallrate, Übergangsrate vom intakten Zustand zu einem Ausfallzustand j.
MTTR = meantime to repair	Reparaturrate, Übergangsrate von einem Ausfallzustand zum intakten Zustand bei sofort entdecktem Ausfall.
TI = inspection time	Regenerierungsrate einer in regelmäßigen Abständen inspizierte Komponente.

Aus den Übergangsraten werden in den Programmen asymptotische Werte für die Eintrittswahrscheinlichkeiten Pi nach dem Ansatz einer Differentialgleichung gerechnet:

$$\frac{dP_i}{dt} = \sum_{\substack{j=0 \\ j \neq i}}^{n-1} \lambda_{j \to i} \cdot P_j - P_i \cdot \sum_{\substack{j=0 \\ j \neq i}}^{n-1} \lambda_{i \to j}$$

Für eine binäre Komponente ergibt sich daraus:

$$P_0 = \frac{\lambda_{1 \to 0}}{\lambda_{1 \to 0} + \lambda_{0 \to 1}} \quad ; \quad P_1 = \frac{\lambda_{0 \to 1}}{\lambda_{1 \to 0} + \lambda_{0 \to 1}}$$

wobei $\lambda_{1 \to 0}$ = Reparaturrate, $\lambda_{0 \to 1}$ = Ausfallrate.

Ist das Zeitverhalten der Ausfallrate einer Komponente nicht exponentiell, wie im obigen Ansatz vorausgesetzt, empfiehlt es sich die zuvor errechnete Eintrittswahrscheinlichkeit direkt einzugeben.

Für Komponenten mit mehr als zwei Zuständen ergibt sich aus diesem Ansatz eine Matrix, die wegen ihrer Besonderheiten (alle $\lambda_{i \to j} \geqslant 0$, alle $P_i \geqslant 0$) mit einem einfachen Iterationsverfahren gelöst werden kann.

4.2 Die Modellierung statistischer Abhängigkeiten mit einer Superkomponente

Die Modellierung einer Superkomponente kann an den in Abb. 2 dargestellten Zustandsdiagrammen abgelesen werden.

Im ersten Fall besteht für die beiden binären Komponenten a und b ein sog. Common-Mode-Ausfall, der nicht berücksichtigt werden kann, wenn beide Komponenten als unabhängige Komponenten in den Fehlerbaumn eingegeben werden. In einer Superkomponente S, die den Regeln der BAWRV unterliegt, ist es möglich den Übergang von
$S_0 = \bar{a} \wedge \bar{b}$ zu $S_2 = a \wedge b$ mit λ_c zu berücksichtigen.

Das zweite Diagramm stellt den Fall dar, daß die Komponente X sofort repariert wird, wenn eine Warnung erfolgt; fällt aber das Warnsystem y aus, so kann der Fehler erst nach Ablauf des Inspektionsintervalls behoben werden, daher existieren zwei Übergangsraten von Zustand X_1 zu X_0, die vom Zustand y_0 oder y_1 abhängig sind (μ_x und μ_I)

Beim Einsetzen einer Superkomponente in den Fehlerbaum muß beachtet werden, daß überall da, wo im originalen Fehlerbaum a_1 bzw. X_1 als Eingang vorhanden war, jetzt $S_1 \vee S_3$ bzw. $U_1 \vee U_3$, also alle die Zustände der Superkomponente eingehen müssen, in denen a_1 bzw. X_1 nicht negiert vorhanden ist.

4.3 Die Methode der Inhibitorfunktionen

Die Methode der Inhibitorfunktionen geht von der Überlegung aus, daß die Bedingungen, unter denen die Übergangsraten einer abhängigen Komponente wechseln, definiert werden können. Je nachdem, ob eine Bedingung, dargestellt als Funktion anderer Komponenten, eintritt oder ob die Komplementfunktionen eintreten, wird die zugeordnete Wahrscheinlichkeit eingesetzt. Die Funktion zusammen mit einer oder mehreren Komplementfunktionen kann Inhibitor-Komponente genannt werden; sie ist eine Komponente mit 2 oder mehr Zuständen; da sie aber mit der internen Zusammensetzung ihrer Zustände bekannt sein muß, werden ihre einzelnen Zustände als einander ausschließen-

de Boole'sche Funktionen gegeben. Jedem Zustand der Inhibitorfunktion ist ein Satz von Wahrscheinlichkeitsdaten zuzuordnen, der bei der Berechnung der Wahrscheinlichkeit herangezogen wird, je nachdem, ob die mit einem Zustand der Inhibitorfunktion definierte Bedingung eingetreten ist oder nicht.

In das Programm MUSTAFA werden die Zustände der Inhibitorkomponente in Form von Minimalschnitten der den Zustand der Komponente beschreibenden Funktion eingegeben. Nach Berechnung der Topfunktion wird jedes Monom, das eine als abhängig deklarierte Komponente d enthält mit den Zuständen seiner Inhibitorkomponente Id verglichen. Trifft die Bedingung für ein Monom M_i zu, so

$$M_i \wedge \overline{Id_j} = 0$$

wird die dem j-ten Zustand der Inhibitorkomponente zugeordnete Eintrittswahrscheinlichkeit für die Komponente d eingesetzt. Trifft diese Bedingung für keinen Zustand j der Inhibitorkomponente zu, so wird M_i mit den Funktionen aller Zustände der Inhibitorkomponente multipliziert.

Wenn also $M_i \wedge \overset{m}{\underset{j=1}{\bigvee}} \overline{Id_j} \neq 0$ wird $M_i^* = M_i \wedge \overset{n}{\underset{j=1}{\bigvee}} Id_j$ berechnet

und in den dabei entstehenden Monomen die für das laufende j geltende Wahrscheinlichkeit eingesetzt.

Die in den beiden Zustandsdiagrammen abgebildeten Beispiele werden mit der Methode der Inhibitorfunktionen auf folgende Weise behandelt:
Die Komponente a wird als abhängige Komponente deklariert (nicht-privilegiert), die Komponente b wird als privilegierte Komponente, die wie eine unabhängige Komponente behandelt wird, deklariert; sie muß in den notwendigen Inhibitorfunktion Ia der Komponente a erscheinen.

$$Ia_1 = b_0;\ Ia_2 = b_1$$

In diesem Falle müßten für den Ausfall der Komponente a eine korrigierte Eintrittswahrscheinlichkeit eingegeben werden, damit für die Bedingung a_1 b_1 (gemeinsamer Ausfall/ nicht das Produkt $E\{a_1\} \cdot E\{b_1\}$ sondern $E\{cm\}$ der Erwartungswert des Common-Mode-Ausfalls errechnet wird.

Dies geschieht im Programm dann, wenn für diese Bedingung
$E\{a_1\} = E\{c_m\} \ / E\{b_1\}$ eingegeben wird.

Ohne Korrektur kann dagegen das zweite Beispiel mit der Methode der Inhibitorfunktionen modelliert werden, da es sich um eine einseitige Abhängigkeit handelt. Der Ausfall von y geschieht in jedem Falle unabhängig von X, lediglich die Reparaturrate von X ist abhängig vom Zustand der Komponente y. Die Inhibitorfunktion der abhängigen Komponente X lautet $Ix_1 = Y_0$; $Ix_2 = Y_1$. In diesem Falle würde man für X die Übergangsraten eingeben, wobei die Ausfallrate unter beiden Bedingungen der Inhibitorfunktion die selbe ist, lediglich die Reparaturrate wechselt mit dem Zustand der Inhibitorfunktion. Das Beispiel zeigt, daß die Modellierung statistischer Abhängigkeiten mit Hilfe von Inhibitorfunktionen eine sorgfältige und Vorbereitung erfordert, die am besten mit Hilfe der Zustandsanalyse erfolgt.

4.4 Die Methode der Teilungskomponente

Auch die Darstellung statistischer Abhängigkeiten mit einer Teilungskomponente erfordert eine sorgfältige Vorbereitung. Eine Teilungskomponente ist wie die Inhibitorfunktion eine Komponente mit 2 oder mehr eingeschränkten Boole'schen Variablen. Wie die Inhibitorfunktion wird für jeden Zustand eine Funktion angegeben. Bei der Rechnung wird der Top eines Fehlerbaums mit jedem Zustand P_j der Teilungskomponenten verknüpft.

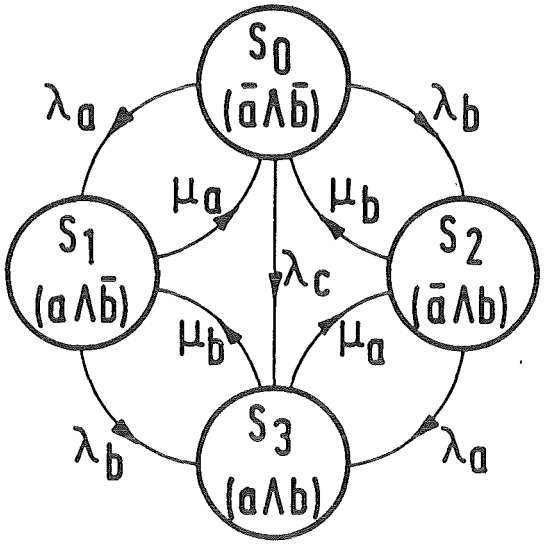
$$F_{top} = \bigvee_{j=1}^{n} (F_{top} \wedge P_j)$$

Für jeden Zustand P_j wird ein dimensionloser Faktor S_j eingegeben; das Ergebnis der Wahrscheinlichkeitsrechnung wird

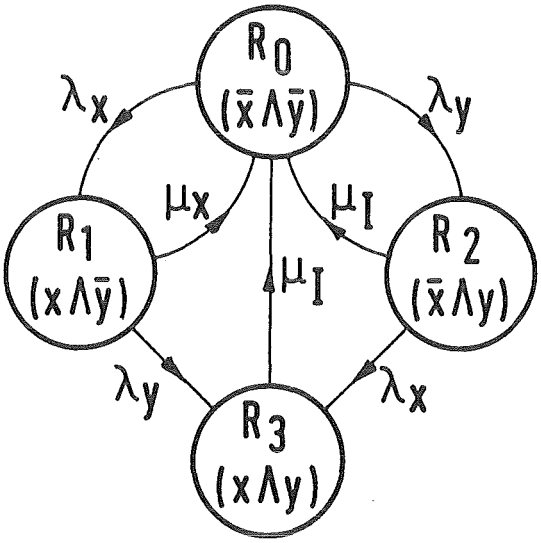
$$E\{F_{top}\} = \sum_{j=1}^{n} E\{F_{top} \wedge P_j\} \cdot S_j$$

Mit den Zuständen P_j werden aus F_{top} diejenigen Teile der Topfunktion ausgefiltert bei denen sich die Eintrittswahrscheinlichkeit nach bestimmten aus Vorrechnungen ermittelten Faktoren ändert, ($S_j \neq 1$).

Caldarola /15/ hat dieses Verfahren auf ein Untersystem angewendet, in dem ein bestimmter Ausfall, der in den meisten Fällen nach Warnung sofort beseitigt wurde, bei einer bestimmten Konstellation unentdeckt blieb, und erst behoben werden konnte, wenn



λ_a, λ_b = Ausfallrate der Komponente a bzw. b
μ_a, μ_b = Reparaturrate der Komponente a bzw. b
λ_c = Ausfallrate des Common-mode-Ausfalls



λ_x, λ_y = Ausfallrate der Komponente x bzw. y
μ_x = Reparaturrate der Komponente x
μ_I = 1/Inspektionszeit

Abb. 3: Zustandsdiagramme bei Vorhandensein statistischer Abhängigkeiten.

- entweder bei Reparatur des die Warnung auslösenden Teils auch die betreffende Komponente inspiziert wurde

- oder, wenn das ganze Untersystem nach Ablauf eines Inspektionsintervalls geprüft wurde.

Der dimensionslose Faktor wurde in einem speziellen Zustandsanalyse-Programm errechnet. Der Faktor wurde gebildet aus dem Verhältnis zwischen idealem Zustand (=Unverfügbarkeit bei sofortiger Reparatur aller Fehler) und zeitabhängiger Unverfügbarkeit bei Auftreten unentdeckter Fehler.

4.5 Übergänge von einem Systemzustand zu einem anderen

Im Programm MUSTAFA können Übergänge von einem Systemzustand zu einem anderen gerechnet werden, wenn

- jeder Systemzustand durch einen Fehlerbaum dargestellt ist,

- die Boole'schen Funktionen der Fehlerbäume sich gegenseitig ausschließen:

$$TOP_j \wedge TOP_k = 0 \quad \text{für alle } j = k$$

- alle Zustände des Systems sich zu 1 ergänzen:

$$TOP_1 \vee TOP_2 \vee \ldots TOP_n = 1$$

Bei dieser Option des Programms werden die Boole'schen Funktionen auf externe Speichereinheiten geschrieben, von wo sie zur Berechnung der sog. Übergangsfunktionen wieder gelesen werden.

Die Übergangsfunktionen von TOP_j zu TOP_K wird berechnet aus der Konjunktion der zu beiden Systemzuständen gehörenden Funktionen. Bei der Konjunktion der einzelnen Monome dieser Funktionen $m_{i,j} \wedge m_{l,K}$ entsteht ein neues Monom m_n verknüpft mit einem Übergang von einem Zustand einer Komponente zu einem anderen. Enthält z.B. $m_{i,j}$ als Bestandteil C_1 und enthält $m_{l,K}$ als Bestandteil C_o und ist dies das einzige gegenseitig sich ausschließende Paar von Variablen, so ist $m_n = 0$ und wird zur Berechnung der gesamten Übergangsrate herangezogen.

Die Übergangsrate von TOP_j zu TOP_K wird gebildet aus der Summe aller dieser Komponentenübergänge dividiert durch die Eintrittswahrscheinlichkeit von TOP_j.

5. Vollständige Übersicht über die Anwendung der Methoden und Verwendungszweck in den beiden Programmen MUSTAFA und MUSTAMO

Die folgende Tabelle 1 gibt eine vollständige Übersicht über den Einbau der Möglichkeiten und Methoden in die beiden Programme MUSTAFA und MUSTAMO. Das Fehlen einzelner Optionen im einen oder im anderen Programm erklärt sich aus den Zielrichtungen, die die Entwicklung der beiden Programme bestimmt haben.

Das Programm MUSTAFA wurde zunächst entwickelt, um die von Caldarola eingeführte BAWRV in einem analytischen Fehlerbaumprogramm anzuwenden. Im Laufe der Entwicklung konnten, da von Anfang an auf eine sorgfältige Trennung von Boole'scher Rechnung und angeschlossener Wahrscheinlichkeitsrechnung Wert gelegt wurde, Möglichkeiten einer vielseitigen und genauen Analyse von Systemen,deren Zustände sich mit Fehlerbäumen beschreiben lassen, eingebaut werden. (Fehlerbaumanalyse kombiniert mit Zustandsanalyse).

Gegen Ende der Entwicklung des Programms MUSTAFA sind uns einige Fehlerbäume bekannt geworden, deren Umfang und Grad der Vermaschung es ratsam erscheinen ließ, das Programm MUSTAFA durch ein weiteres Programm zu ergänzen, um ein Werkzeug zu haben, mit dem die Probleme der starken Vermaschung gelöst werden konnten. Das zunächst als Hilfsprogramm geplante Programm MUSTAMO hat sich in der Folgezeit als selbstständiges Programm entwickelt, das zwar nur Fehlerbäume kohärenter Systeme bearbeiten kann, aber Gewähr bietet, daß umfangreiche und stark vermaschte Systeme bearbeitet werden können. Dabei hat sich die BAWRV auch bei der Entmaschung als geeignetes Werkzeug erwiesen.

Tabelle 1

Übersicht über den Einbau der Methoden in den Programmen MUSTAFA und MUSTAMO

Zweck	Methoden und Möglichkeiten	
	MUSTAFA	MUSTAMO
1. Eingabe der Komponenten Eingabe der Gatter eines Fehlerbaums	Komponenten mit n Zuständen (2 n 8) Und-Gatter Oder-Gatter m aus n-Systeme: m/n, mindestens 4 aus n, = m/n, exakt m aus n, m/n, nicht mehr als m aus n, Vorgänger des Gatters.	Komponenten mit n Zuständen (2 n 16) Und-Gatter Oder-Gatter m aus n Systeme: m/n, mindestens m aus n, Vorgänger des Gatters.
2. Eingabe der Wahrscheinlichkeitsdaten der Komponenten	Wahlweise als Übergangsraten von einem Zustand der Komponente zu einem anderen, oder als Eintrittswahrscheinlichkeiten der einzelnen Zustände.	wie bei MUSTAFA
3. Auflösung des Fehlerbaums bzw. seiner Teile	TOP-down-Algorithmus mit Prioritätsregel -	TOP-down-Algorithmus mit Prioritätsregel. Zweistufige Anwendung nach Definition , sog. Stop-Gatter.
4. Modellierung statistischer Abhängigkeiten	Eingabe von Superkomponenten als Komponenten mit n Zuständen (s. unter 1.). Deklarierung einer Komponente als nicht-priviligierte abhängige Komponente, Eingabe einer Inhibitorfunktion, die priviligierte Komponenten enthält. Methode der Teilungskomponente.	Eingabe von Superkomponenten als Komponenten mit n Zuständen (s. unter 1.) - -

Zweck	Methoden und Möglichkeiten	
	MUSTAFA	MUSTAMO
5. Reduktion und Erweiterung der booleschen Strukturfunktion	Feststellung der Kohärenz Berechnung der vollständigen Basis Feststellung der obligatorischen Primobligationen Reduzierung zu einer kürzeren oder der kürzesten Basis Erweiterung durch Bennett-Algorithmus Erweiterung durch Reystone-Algorithmus	- - - - - -
6. Entmaschung und Modularisierung	Automatische Abtrennung von log. unabhängigen Superkomponenten (nicht immer möglich). - - - -	Abtrennung von log. unabhängigen Superkomponenten mit mehr als 2 Zuständen. Abtrennung von Superkomponenten mit mehr als 2 Zuständen gebildet aus Unterbäumen, die gemeinsam log. unabhängig sind. Modulbildung durch Ausklammern von Schlüsselkomponenten. Bildung vorläufiger Superkomponenten zur besseren Auflösung von vermaschten Fehlerbäumen. Splittingverfahren - durch Deklarierung einzelner Komponenten als deterministisch (1. Art). - durch automatische Bildung aller Kombinationen (0, 1) von deterministischen Komponenten (2. Art).
7. Übergänge von einem Systemzustand zu einem andern	Berechnung der Übergangsfunktionen, daraus Berechnung der Übergangsraten.	-

6. Die Optionen der Programme MUSTAFA und MUSTAMO, Ein- und Ausgabebeschreibung

Für jedes der beiden Programme folgt die Auflistung aller Anwendungsmöglichkeiten mit Angabe der dabei zu verwendenden Eingabeparameter, die Beschreibung der besonderen Eingabe; am Ende folgt die Beschreibung der (mit wenigen Einschränkungen) für beide Programme gültige Eingabe der Fehlerbaumdaten.

6.1 Die Optionen des Programms MUSTAFA

Die für die Festlegung der gewünschten Rechnung wichtige Eingabeparameter sind NOPT (Nummer der gewünschten Option), NTR (Parameter für die Berechnung der Übergangsfunktionen und NUN(J), J = 1,8, Eingabe der Nummer externer Speichereinheiten.

Bei allen Optionen muß NUN(1) bis NUN(4) die Nummer einer Speichereinheit angeben, auf der während der Rechnung Boole'sche Funktionen zwischengespeichert werden (temporäre Datensätze). NUN(5) bis NUN(8) wird nur benötigt, wenn NTR > 0; werden die Fehlerbäume der Systemzustände in ein und demselben Job gerechnet, können auch diese Speichereinheiten als temporäre Datensätze deklariert werden. Alle Datensätze haben variable Record-Länge (LRECL = X), und variables Record-Format (RECFM = VBS).

NTR ist nur dann > 0, wenn Übergangsraten von Systemzustand zu Systemzustand berechnet werden soll.

Jede der angegebenen Optionen ist die Anweisung, einmal das Programm zu durchlaufen und verlangt die Eingabe eines Fehlerbaums. In ein und demselben Job kann das Programm mehrmals nacheinander für je einen Fehlerbaum durchlaufen werden.

NOPT = 0, NTR = 0, NUN(5) ff = 0
Berechnung der sich bei Auflösung des Fehlerbaums ergebende Minimalschnitte.

NOPT = 1, NTR = 0, NUN(5) ff = 0
Prüfung des Fehlerbams auf Kohärenz; bei Kohärenz evtl. notwendige Reduzierung auf Primimplikanten, bei nicht-kohärenten Systemen Reduzierung auf

kürzere Basis wird durchgeführt. Die Berechnung der Eintrittswahrscheinlichkeit wird bei NOPT = 0,1 durchgeführt, wenn für alle Komponenten die Wahrscheinlichkeiten der Komponentenzustände vorhanden sind. Alle folgenden Optionen verlangen Vollständigkeit der Eintrittswahrscheinlichkeiten, alle Optionen mit NTR > 0 Vollständigkeit der Übergangsraten.

NOPT = 2, NTR = 0:
Berechnung wie NOPT = 1; Erweiterung der Funktion durch Keystone-Algorithmus; Berechnung des exakten Zahlenwertes der Eintrittswahrscheinlichkeit des TOP.

NOPT = 3, NTR = 0:
Berechnung wie 0 oder 1, Erweiterung der Funktion durch Bennet-Algorithmus, Berechnung desexakten Zahlenwertes der Eintrittswahrscheinlichkeit.

NOPT = 2, NTR = 1, NUN(5) > 0:
Eingabe eines Fehlerbaums, der einem Systemzustand entspricht.
Berechnung wie bei NOPT = 2; Speicherung der Boole'schen Strukturfunktion auf Speichereinheit NUN(5).

NOPT = 2, NTR = 2, NUN(5) > 0:
Mindestens NUN(6) > 0, maximal NUN(6), NUN(7) und NUN(8) > 0.
Eingabe eines Fehlerbaums, der einem weiteren Systemzustand entspricht, Speicherung der Funktion auf NUN(5); Berechnung der Übergangsfunktion und der daraus resultierenden Übergangsraten zwischen der auf NUN(5) neu gespeicherten und auf NUN(6) bis NUN(8) vorhandenen Funktionen der verschiedenen Systemzustände. (Die in NUN(6) ff. angegebenen Nummern müssen denen entsprechen, die in vorangegangenen Durchläufen auf NUN(5) angegeben waren).

NOPT = 2, NTR = 3, NUN(5) > 0; NUN(6) > 0.
Eingabe eines Fehlerbaums, der einem von beiden Systemzuständen eines binären Systems entspricht. Berechnung des eingegebenen Systemzustandes wie bei NOPT = 2, Berechnung der Übergangsraten zwischen den beiden Systemzuständen. In diesem Falle muß der Komplementzustand des binären Systems nicht als Fehlerbaum eingegeben werden, da die Komplementfunktion als Zwischenergebnis des Nelson-Algorithmus vorliegt und gespeichert wird.

NOPT = 12, NTR = 0:
Zusätzliche Eingabe einer Teilungskomponente. Berechnung für jeden Zustand der Teilungskomponente (multipliziert mit Funktion des Fehlerbaums) wie bei NOPT = 2.

6.2 Die Eingabe für das Programm MUSTAFA

Die Eingabe des Programms MUSTAFA gliedert sich in

- zwei Eingabekarten zur Festlegung der Option, MA1 und MA2
- die Eingabe der Fehlerbaumdaten (s. 6.7)
- die Eingabe der Inhibitorfunktionen (wenn notwendig), Karten MA3, MA4, MA5, MA6
- die Trennkarte MA7
- die Eingabe einer Teilungskomponente (wenn notwendig), Karten MA8, MA9 und MA10.

Karte MA1: Format (8I2)

(NUN(J), J = 1,8)

(NUN(J) Nummern einer externen Speichereinheit (s. 6.1)

Karte MA2: Format (8I2, 4E10.4)

NOPT, NTR, NSE, KPR1, KPR2, KPR3, KPR4, EPS, ALPH, BET, GAM.

NOPT, NTR Erläuterung s. 6.1

NSE Steuerung des Abschneidens von Minimalschnitten mit geringem Gewicht (s. EPS) bein Anwendung der Optionen NOPT = 2 oder 3. Wenn NSE = 0 eingegeben wird, führt das Programm die "truncation" vor der Berechnung der vollständigen Basis durch, bei NSE = 0 erst vor Anwendung des Keystone-Algorithmus. Es wird empfohlen bei umfangreicheren Fehlerbäumen, insbesondere bei Berechnungen mit NTR = 3 NSE = 0 zu setzen.

NPRP Steuerung des Unterprogramms PREPAR zur Feststellung der Kohärenz.
NPRP = 0, PREPAR wird übersprungen.

KPR1 bis KPR4 Steuerung der Druckausgabe. Abweichungen von der Standardausgabe, wenn diese Parameter = 0 eingegeben werden, s. 6.3.

EPS Schranke bei der Vernachlässigung von Minimalschnitten mit geringem Gewicht, p_m EPS SUM
p_m = Wahrscheinlichkeit eines Minimalschnitts.
SUM = Summe der Wahrscheinlichkeiten der bis dato vorliegenden Minimalschnitte.
Default option (bei Eingabe 0, EPS = 1 10^{-7}).

ALPH, BET, GAM — Steuerungsgrößen bei der Reduzierung der vollständigen Basis.
Defaultoption (bei Eingabe $\alpha, \beta, \gamma = 0$)
$\alpha = 0{,}166$; $\beta = 0{,}75$; $\gamma = 0{,}125$.
Bei Eingabe $\gamma = 1.0$ wird Suche nach der kürzesten Basis übersprungen, bei Eingabe $\beta = 0$ wird Algorithmus zur Suche der kleinsten Basis in jedem Falle durchgeführt.

Auf die Karte MA2 folgen die Fehlerbaumdaten nach den Anweisungen in Abschnitt 6.6, gültig für beide Programme.

Enthält die Eingabe für eine oder mehrere Komponenten in Karte FT2 die Buchstabenkombination 3 ∝ DTR oder 3 ∝ DET, wodurch diese Komponenten als abhängige Komponenten deklariert sind, so müssen für jede dieser abhängigen Komponenten die Zustände einer Inhibitorfunktion im Ausschluß an die Fehlerbaumdaten eingegeben werden; enthält der Fehlerbaum keine abhängigen Komponenten, so wird die Eingabe vom MA3 bis MA6 übersprungen.

Karte MA3: Format (I4, 4X,A4)

NST, NCMP

NST — Nummer des Zustandes der Inhibitorfunktion
NST muß mit 1 beim ersten eingegebenen Zustand beginnen, die folgenden müssen fortlaufend numeriert werden.

NCMP — 4stellige ∝-numerische Bezeichnung der Komponente, deren Inhibitorfunktionen dieser Stelle eingegeben wird.
NCMP muß mit Bezeichnung auf FT2 übereinstimmen.

Karte MA4: Format (I4, 4X, 8 (A4, I2, 2X)

LG, (NO(L), N2(L), L =1, LG)

Format Fortsetzungskarte (8X, 8 (A4, I2, 2X))

LG — Anzahl der Primärvariablen im Monom

NO(L), N2(L) — Identifikation und Nummer des Zustandes eines Zustandes einer Komponente (Primärvariable).

Karte MA4 ist für jedes weitere Monom, das den Zustand J einer Inhibitorfunktion impliziert, zu wiederholen.

Karte MA5: Format wie MA3

LG, KENDS

LG = 1

KENDS = 4¢ENDS

Karte MA5 markiert das Ende eines Zustandes einer Inhibitorfunktion.

Karte MA6: Format wie MA3

LG, KEND

LG = 0

KEND = 4¢END b

Karte MA6 ist am Ende der Eingabe der Inhibitorfunktionen einzufügen.

Karte MA7: Format (A8, 8A4)

DECK, (LISTC(j), J = 1,8)

DECK = 8¢ ---------- Kennzeichnung als Trennkarte

LISTC = blanc, es wird zusätzliche gegliederte Liste der Boole'schen Funktion verlangt.

LISTC(J) Identifikation einer Komponente, die bei Aufgliederung in Module als Schlüsselkomponente ausgeklammert werden soll.
Die Anzahl der Schlüsselkomponenten wird vom Programm (LISTC(J) = blanc) ermittelt.

Die Trennkarte MA7 schließt die Eingabe für eine Durchlauf durch das Programm ab. Es folgt entweder die Eingabe einer weiteren Option (folgender Durchlauf) oder, wenn NOPT = 12 gewählt wurde, die Eingabe der Teilungskomponente, MA8 - MA10.

Karte MA8: Format (A4, 4X, F8.4)

NMFC, RAT

NMFC 4stellige alphanumerische Bezeichnung des Zustandes der Teilungskomponente.

RAT Dimensionsloser Faktor bei der Berechnung der Wahrscheinlichkeit (s. 4.4).

Karte MA9: Format und Inhalt wie MA4.
Karte MA9 wird für ale Monome, die den Zustand NMFC der Teilungskomponente implizieren, eingegeben.

Karte MA10: Format und Inhalt wie MA5.
Endmarkierung der den Zustand NMFC definierenden Funktion.

Karte MA11: Format (A4)
KEND
KEND = 4␣END b
Endmarkierung der Eingabe der gesamten Teilungskomponente.

6.3 Die Druckausgabe des Programms MUSTAFA

Die Druckausgabe des Programms enthält

- Fehlermeldungen
- Informationen über durchgeführte Operationen mit Zeitangabe
- Listen
- Ergebnis der Wahrscheinlichkeitsrechnung.

Zur Standardausgabe gehören (je nach Option) folgende Listen:

Liste 1	Wiedergabe der Eingabekarten
Liste 2a	Liste der Fehlerbaumknoten (Primärvariable und Gatter) mit Vorgängern und Nachfolgern
Liste 3	Liste der Komponenten mit Wahrscheinlichkeitsdaten
Liste 4	Liste der vollständigen Basis
Liste 5a	Liste der Minimalschnitte, (bzw. Primimplikanten) der Funktion mit ihrer Wahrscheinlichkeit
Liste 5b	Liste der (NOPT = 3) erweiterten Terme, berechnet mit Bennet-Algorithmus.

Mit den Steuerungsparametern KPR1 bis KPR können folgende Änderungen vorgenommen werden:

KPR1 = -1 keine Liste der Fehlerbaumknoten wird gedruckt
= 0 nur Liste 2a wird gedruckt (Standard)
= 1 nur Liste 2b wird gedruckt
Liste 2b enthält die vom Programm nach Vorgängern und Nachfolgern geordnete Liste des Fehlerbaums. Die in der Eingabe vorhandenen Mehrfach-Gatter sind zu Gattern mit nur je zwei Vorgängern aufgespalten.
= 2 beide Listen werden gedruckt

KPR2 = 0 Liste 4a bzw. 4b wird nicht gedruckt

KPR3 = 0 Liste 3 (vollst. Basis) wird nicht gedruckt

KPR4 > 0 Zusätzliche Liste der Komponenten und der Nummern der Minimalschnitte, in denen sie enthalten sind, wird ausgedruckt

KPR4 = 2 die Übergangsfunktionen (NTR 0) werden ausgedruckt

Die Fehlermeldungen im Programm MUSTAFA enthalten eine Nummer ERROR i zur leichteren Feststellung des Fehlers auf den Eingabekarten.

i	Art des Fehlers
0	Obligatorische Karte zur Fallidentifikation nicht vorhanden
1	Zu viele Kommentarkarten
2	Identifikation einer Komponente mehrfach verwendet
3	Fehler bei Bezeichnung eines Zustands einer deterministischen Komponente
4	Identifikation eines Gattes mehrfach verwendet
5	Ungültige Typenbezeichnung eines Knotens oder m (bei m aus n-Systemen) außerhalb der Grenzen
7,8	Nicht identifizierte Komponente in einer Inhibitorfunktion
9,10	Komponente bzw. deren Zustand konnte nicht identifiziert werden
11	Fehler bei Eingabe der Wahrscheinlichkeiten bzw. Übergangsraten.

6.4 Die Eingabe für das Programm MUSTAMO

Die Eingabe des Programms MUSTAMO beginnt mit den Fehlerbaumdaten (s. 6.6). Es folgen die Eingabekarten für eine vorläufige Rechnung a (und wenn notwendig b), danach die Karten MO1, MO2 und MO3 jeweils für eine vom Benutzer vorgesehene horizontale Schnittstelle des Programms in aufsteigender Reihenfolge, d.h. eine unterhalb eines als Schnittstelle gewählten Gatters liegende weitere Schnittstelle muß zuvor eingegeben und berechnet sein. Das Programm führt nach Eingabe der Daten für eine Schnittstelle deren Berechnung sofort durch, der Programmfluß kehrt danach zur Eingabeanweisung wieder zurück. Die Eingabe für die abschließende Berechnung der TOP-Funktion muß daher am Ende erfolgen.

Die Eingabe für die Fehlerbaumdaten ist in 6.6 beschrieben. Sollten externe Speichereinheiten benötigt werden, so sind deren Nummern auf den Positionen 1-8 der obersten Karte (Fallidentifikation) anzugeben. Die Eingabe für eine vorläufige Rechnung zur Prüfung der log. Unabhängigkeit und zur Ermittlung von Schlüsselkomponenten enthält die Karten Va und Vb.

Karte Va: Format (I4)
NPRF Anzahl der Karten Vb.
Wenn vorläufige Rechnung nicht mehr notwendig und nicht gewünscht wird, ist NPRF = 0 oder blanc einzugeben und Karte Vb entfällt.

Karte Vb: Format (I4, 6XX, 12 (A4, 1X))
(NSCM, (NO(J), J = 1, NSCM)

NSCM = Anzahl der Gatter NO(J)

NO(J) = Name der Gatter, deren Territorium (= Liste der in der Funktion der Gatter enthaltenen Komponenten) ausgedruckt werden soll.
Für Gatter, die zur Lösung mit NOPT = 1,2 oder 12 vorgesehen sind, wird NSCM = 1 und NO(1) = Name des Gatters, eingegeben.
Gatter, die mit NOPT = 3,4,13 oder 14 als eine Superkomponente gelöst werden sollen, sind auf einer Karte V_b gemeinsam einzugeben.
Der Output enthält alle Komponenten, die logische Abhängigkeiten verursachen in einer gesonderten Liste unter der Überschrift KEY COMPONENTS.

Es folgt für jede gewünschte Schnittstelle die Karte MO1, MO2 und MO3.

Karte MO1: NOPT, NGT, NKE, NHALT, KF, (IGT(J), J = 1, NGT)
Format (512, 4 (A4, 1X))

NOPT = Nummer der Option
NGT = Anzahl der Gatter, die eine Schnittstelle bilden
NKE = Anzahl der Schlüsselkomponenten
NHALT = Anzahl der Stop-Gatter
KF = Steuerparameter für Modulbildung bei vorläufigen Superkomponenten
IGT(J) = Identifikationen von ausgewählten Gattern

Karte MO2: Format (14 (A4, 1X))
(KEY(J), J = 1, NKE)
KEY(J) = Identifikationen von Schlüsselkomponenten

Karte MO3: Format wie MO2
(NH(J), J = 1, NHALT)
NH(J) = Identifikationen von Gattern bis zu denen herab die erste Stufe des Top-down-Algorithmus durchgeführt werden soll.

Die Karten MO2 und MO3 entfallen, wenn NKE = 0 bzw. NHALT = 0 eingegeben wurde.

6.5 Die Optionen des Programms MUSTAMO

Es folgt eine Auflistung aller Möglichkeiten mit Angabe der einzugebenden Parameter. Weggelassen werden die Parameter, die für betr. Option keine Bedeutung haben. Die Auswahl von Stop-Gattern (NHALT 0) bleibt in jedem Falle dem Benutzer überlassen, wenn es wegen der besonderen Vermaschung des Fehlerbaums geboten erscheint.

NOPT = 1, NGT = 1, NKE = 0, IGT (1):
Das Gatter IGT(1) hat unter sich einen logisch unabhängigen Fehlerbaum. Nach Ausführung diser Option wird IGT(1) in der folgenden Berechnung als (Super)-Komponente mit zwei Zuständen behandelt und kann, wenn notwendig, mit seiner eigenen Identifikation als Schlüsselkomponente innerhalb der Funktionen von Gattern oberhalb benutzt werden.

NOPT = 2, NGT = 1, NKE > 0, IGT(1) (KEY(J), J = 1, NKE):
Die für das Gatter IGT(1) berechnete Boole'sche Funktion wird aufgeschlüsselt in Kombinationen von obligatorischen Schlüsselkomponeten und Modulen. Der Fehlerbaum unterhalb IGT(1) wird vereinfacht und endet bei den Schlüsselkomponenten und den in der Folgerechnung wie binäre Komponenten behandelten Modulen.

NOPT = 3, NGT (1), NKE = 0, (IGT(J), J = 1, NGT):
Aus den Gattern (IGT(J), J = 1, NGT) wird eine Superkomponente mit 2^{NGT} Zuständen gebildet. Superkomponenten wird (in der Reihenfolge der Eingabe) die Bezeichnung SC nn, nn = 01,02 ff. gegeben, sie sind mit dieser Bezeichnung SC01, SC02.... als Schlüsselkomponenten in der Rechnung oberhalb, wenn notwendig, anzugeben.

NOPT = 4, NGT > 1, NKE = 0, (IGT(J), J = 1, NGT):
Die Gatter IGT(J), J = 1, NGT enthalten die vom Benutzer durch vorherige Umformung des Fehlerbaums definieten NGT Zustände einer Superkomponente. Für Bezeichnung SC nn etc. gilt das gleiche wie bei NOPT = 3.

NOPT = 12, NGT = 1, NKE > 0, KF, IGT(1), NUN(1) und NUN(2) 0:
Die Berechnung erfolgt wie bei NOPT = 2 mit dem Unterschied, daß das Gatter IGT(1) in der Folgerechnung vorläufig (zeitweilig) als Superkomponente mit 2 Zuständen betrachtet wird. Bei Berechnung der TOP-Funktion wird die auf eine externe Einheit abgespeicherte Funktion von IGT(1) abgerufen. Die Speicherung erfolgt bei KF = 0 Speicherung der unveränderten Funktion ohne Bildung von Modulen bei KF = 1 weden Kombinationen von Schlüsselkomponenten und Modulen gebildet, die am Ende nicht weiter aufgelöst werden. IGT(1) ist bei der nachfolgenden Berechnung der TOP-Funktion als obligatorische Schlüsselkomponente anzugeben.

NOPT = 13 oder 14, NGT > 1, NKE > 0, KF, (IGT(J), J = 1, NGT), NUN(1), NUN(2) 0:
Die Berechnung von NOPT = 13 erfolgt wie bei NOPT = 3, bei NOPT = 14 wie bei NOPT = 4 mit dem Unterschied, daß die dabei gebildete Superkomponente SC nn eine vorläufige ist. Für die Speicherung gilt das gleiche wie bei NOPT = 12.
SC nn ist bei Berechnung der TOP-Funktion als obligatorische Schlüsselkomponente anzugeben.

Berechnung der TOP-Funktion:

Die Berechnung der TOP-Funktion eines Fehlerbaums geschieht mit

- NOPT = 2, NGT = 1, NKE > 0, IGT(1); IGT(1) = Identifikation des Top-Gatters, wenn alle zuvor verlangten Optionen 10 waren (d.h., wenn keine vorläufigen Superkomponenten gebildet wurden).

- NOPT = 10 oder 20, NGT = 1, NKE > 0, IGT(1) = wenn zuvor mindestens eine vorläufige Superkomponente gebildet wurde.
  IGT(1) ist in allen Fällen die Identifikation des Gatters an der Spitze des Fehlerbaums.

NOPT = 10 und NOPT = 20 unterscheiden sich dadurch, daß bei NOPT = 20 vor Einsetzung der Funktionen der vorläufigen Superkomponenten Modularisierung der Funktion durch Ausklammern der Schlüsselkomponenten vorgenommen wird. Bei NOPT = 10 entfällt diese Modularisierung. In beiden Fällen wird die endgültige Top-Funktion nach Auflösung der vorläufigen Schlüsselkomponenten in Kombinationen von Schlüsselkomponenten und Modulen aufgegliedert.

Für die Aufgliederung in Schlüsselkomponenten und Modulen können auch nicht-obligatorische Schlüsselkomponenten eingegeben werden, wenn aus äußeren Gründen eine diesbezügliche Aufgliederung gewünscht wird.

Komponenten, die auf die eine oder andere Art als deterministische Komponenten deklariert werden, bewirken eine Umformung des Fehlerbaums; sie verschwinden im umgeformten Fehlerbaum und müssen daher nicht als obligatorische Schlüsselkomponenten angegeben werden. Weite Teile des Fehlerbaums können bei günstiger Wahl von deterministischen Komponenten als logisch unabhängig betrachtet und mit den Optionen 1-4 behandelt werden, wodurch erheblich Rechenzeit eingespart werden kann.

Die Deklarierung deterministischer Komponenten erfolgt

- durch Vorschalten einer Karte, die auf den Positionen 1-4 4 0&*DET enthält vor die betreffende Komponentenkarte FT2 (für diese Komponente sind Wahrscheinlichkeitsdaten, d.h. Übergangsraten bzw. Eintrittswahrscheinlichkeiten einzugeben)

- durch Eingabe von N2 < 0 auf Karte FT2 und Nummer des auf 1 gesetzten Zustandes in Parameter N4.

Dieses in 3.5 beschriebene automatische Splittingverfahren wird für alle Kombinationen der mit *DET deklarierten Komponenten durchgeführt.

6.6 Die Eingabe der Fehlerbaumdaten

Die Fehlerbaumdaten können in beiden Programmen nach den hier beschriebenen Anweisungen eingelesen werden; die einzige Ausnahme bildet die Deklarierung abhängiger Komponenten, für die im Programm MUSTAFA eine Inhibitorfunktion eingegeben werden muß. Diese Möglichkeit besteht nur im Programm MUSTAFA.

a) Die Karte FT1 enthält einen Kommentar zur Fallidentifikation; dieser Kommentar wird als Überschrift beim Ausdruck der Ergebnisse verwendet.

Karte FT1: Format (4I2, 8A8)

IU1, IU2, IU3, IU4, IDCASE

IU1, IU2, IU3, IU4 = Nummern externer Einheiten, in MUSTAMO benötigt.

IU1, IU2 = Speichereinheiten, die benötigt werden, wenn mindestens einmal eine Option > 10 verlangt wird.

IU3 = Speichereinheit, von der bei Restart die Daten der Vorrechnung gelesen werden. Wenn 0 < IU3 < 50 entfällt die Fehlerbaumeingabe.

IU3 > 50 = Speichereinheit, die benötigt wird, wenn mindestens eine Komponente mit DET deklariert wird.

IU4 = Speichereinheit auf die die Daten des Fehlerbaums und Ergebnisse einer Rechnung gespeichert werden sollen, die bei einem Restart benötigt werden, sonst IU4 = 0.
In MUSTAFA sind IU1 bis IU4 ohne Bedeutung.

IDCASE = maximal 64 Zeichen umfassender Kommentar.

b) Der Block der Komponentendaten umfaßt die Karten FT2 bis FT4.
Für jede Komponente wird eine Karte FT2, und wenn verlangt, eine Karte FT3 eingegeben.

Karte FT2: Format (A4, 2I4, I3, I1)

NI, N1, N2, N3, N4

NI = Identifikation der Komponente, 4 Zeichen.

N1 = Anzahl der Zustände einer Komponente
in MUSTAFA: 2 < = N1 < = 8,
in MUSTAMO: 2 < = N1 < = 16.

Für jeden Zustand der Komponente generieren die Programme eine Variable, bestehend aus 6 Zeichen: Position 1-4 (NI) bezeichnet die Komponente, Position 5-6 bezeichnet den Zustand; die Reihenfolge ist 0 = Intakt-Zustand, j ($1 \leqslant j \prec N1$) die Ausfallzustände. Zu beachten ist, daß in MUSTAFA die nicht benötigte Dezimalstelle für Zahlen > = 10 immer mit blank eingegeben werden muß.

Die Eingabe von Komponentenzuständen als unmittelbare Vorgänger von Gattern muß dieser Konvention entsprechen, um vom Programm erkannt zu werden.

N2 = Anzahl der auf der folgenden Karte enthaltenen Wahrscheinlichkeitsdaten (N2 ≥ 0). Wenn N2 = 0, entfällt die folgende Karte FT3. Mit N2 < 0 wird die Komponente als deterministisch deklariert, Karte FT3 entfällt.

N3 = Im Normalfall 3b bbb. Ausnahme ist nur die Deklaration abhängiger Komponenten im Program MUSTAFA:

N3 = blank bei abhängiger Komponenten (nicht privilegiert, Inhibitorfunktion mit ausschließlich privilegierten und unabhäng. Komponenten wird verlangt)

N3 = 3b DTR bei privilegierten abhängigen Komponenten. (Inhibitorfunktion wrd verlangt, wird aber nur bei Berechnung der Übergangsfunktionen von Systemzustand zu Systemzustad berücksichtigt)

N4 = Dieser Parameter ist nur bei deterministischen oder abhängigen Komponenten von Bedeutung.
Wenn N2 < 0 (deterministische Komponente):
N4 = Nummer des Zustandes, der 1 gesetzt werden soll.
Wenn N3 ≠ b (abhängige Komponente):
N4 = Anzahl der Zustände der Inhibitorfunktion.
Beide Eigenschaften (determ. und abhäng.) können nicht in ein und derselben Komponente vereint auftreten, da determimistische Komponenten nur in MUSTAMO und abhängige Komponenten nur in MUSTAFA an dieser Stelle deklariert werden.

Karte FT3: Format (7(2I2, E10,4)),
(IDS(1,J), IDS(2,J), B(J),J=1,N2)
Die Wahrscheinlichkeitsdaten können entweder als Übergangsraten oder als Eintrittswahrscheinlichkeiten der Zustände gegeben werden.

Eingabe der Übergangsraten:
IDS(1,J) = Zustand von dem
IDS(1,J) = Zustand zu dem Übergangsrate B (J) gilt
B(J) = Übergangsrate [1/t].

Die nicht benötigte 1. Stelle (10^1) ist auch hier immer mit blanc einzugeben.

Eingabe der Eintrittswahrscheinlichkeiten

IDS (1,j), IDS (2,J) = 2 bb.

B(J) = Eintrittswahrscheinlichkeit des Zustandes j-1.

Es können für ein und dieselbe Komponente Übergangsraten und Eintrittswahrscheinlichkeiten zusammen eingegeben werden. Bei den Eintrittswahrscheinlichkeiten muß auf die Reihenfolge der Zustände (0,1...) geachtet werden.

Karte FT4: Format (A4), Endkarte

KEND

KEND = 4% ENDb

Die Endkarte schließt den Block der Komponentendaten ab.

c) Die Eingabe der Karten für die Gatter des Fehlerbaums:
Für jedes Gatter ist eine Karte FT5 enzugeben. Die Reihenfolge der Gatter ist beliebig.

Karte FT5:

Format (2A4, 2I1, 8(A4, I2, A1))

NI, NTP, M, N, (INP(J), IST(J), NG(J), J = 1, N)

NI = 4stellige -numerische Identifikation des Gatters.

NTP = Typ des Gatters; die folgenden Typen sindmöglich:

	4% ANDb	für Und-Gatter
	4% ORbb	für Oder-Gatter
(*)	4% NOTb	für Negation (nur ein Vorgänger möglich)
	4% MAJb	System m aus n (mindestens m aus n Vorgängern)
(*)	4% EXCb	System m aus n (exakt m aus n Vorgängern)
(*)	4% NMRb	System m aus n (nicht mehr als m aus n Vorgängern)

Die mit (*) bezeichneten Typen können nur im Programm MUSTAFA eingegeben werden.

M = Angabe wieviele aus N Vorgängern (M N) bei Systemen, sonst 0

N = Anzahl der unmittelbaren Vorgänger des Gatters.

INP(J) = 4stellige -numerische Identifikation der Vorgänger.

IST(J) = Identifikation des Zustandes des j-ten Vorgängers.
IST(J) muß bei Komponenten korrekt angegeben werden, wenn INP(J) eine Komponente bezeichnet. Wenn INP(J) ein Gatter bezeichnet, ist IST(J) ohne Bedeutung.

NG(J) = Markierung der Negation: NG(J) = b bewirkt Negation des Vorgängers INP(J).

Der Block der Karten für die Gatter muß ebenfalls mit einer Endkarte FT3 abgeschlossen werden.

7. Anwendungsbeispiele

7.1 Berechnung eines Systems mit mehreren Betriebszuständen im Programm MUSTAFA

Das zur Demonstration des Programms MUSTAFA ausgewählte Beispiel ist ein Energieversorgungssystem, das in dieser, sowie in einer noch weiter vereinfachten Form Caldarola zur Darlegung der Theorie in seinen Veröffentlichungen verwendet hat (Abb. 4).
Die Komponenten dieses Systems sind:

L, F:
Transformatoren mit 3 für den Fehlerbaum relevanten Zuständen:
- intakt
- ausgefallen ohne Stromverbindung
- ausgefallen mit Stromverbindung

I: Transformator, binäres Ausfallverhalten (intakt, ausgefallen)

H, C, D:
Hochspannungsleitung und zwei Versorgungsschienen für die interne Stromversorgung (binäres Ausfallverhalten).

SP, SQ, SR, SU, ST:
Schalter mit 3 für den Fehlerbaum relevanten Zuständen:
- intakt
- geschlossen ausgefallen und öffnet nicht bei Anforderung
- geöffnet ausgefallen und schließt nicht bei Anforderung.

SC:
Eine Superkomponente deren 5 Zustände den möglichen Konstellationen von Netzausfall und Ausfall des Generators entsprechen.

Der Ausfallzustand ist:
TOP1:
Kein Strom für den internen Bedarf des Kraftwerkes.

Weitere Betriebszustände sind:

TOPO:
Intakt-Zustand, Generator liefert Strom an das Netz und für den Eigenbedarf.

TOP2:
Kein Strom an das Netz, Generator arbeitet nur für den Eigenbedarf.

TOP3:
Generator ausgefallen, Fremdeinspeisung aus dem Netz.
Das Gatter GA01 (s. Abb. 4) entspricht TOP1, die übrigen Betriebszustände wurden berechnet mit:

$$\text{TOP } 0 = \overline{\overline{GA01} \wedge \overline{M} \wedge \overline{V}}$$
$$\text{TOP } 2 = \overline{\overline{GA01} \wedge \overline{M} \wedge V}$$
$$\text{TOP } 3 = \overline{\overline{GA01} \wedge M \wedge \overline{V}}$$

und schließen sich daher alle aus; alle Funktionen zusammen ergeben 1.
M und V sind kleine Unterbäume, deren Minimalschnitte wie folgt zusammengesetzt sind:

M = SC1 + SC3 + SC4 + F1 + L1 + F2 L2
V = SC1 + SC2 + SC4

Auf folgender Tabelle (Tab. Nr. 2) sind die Eingabekarten aufgelistet. Der Übersicht wegen wurden die für die Fehlerbäume TOPO, TOP2, TOP3 einzugebenden Duplikate der ersten Eingabe weggelassen und nur die geänderten bzw. zusätzlichen Karten hier abgedruckt.

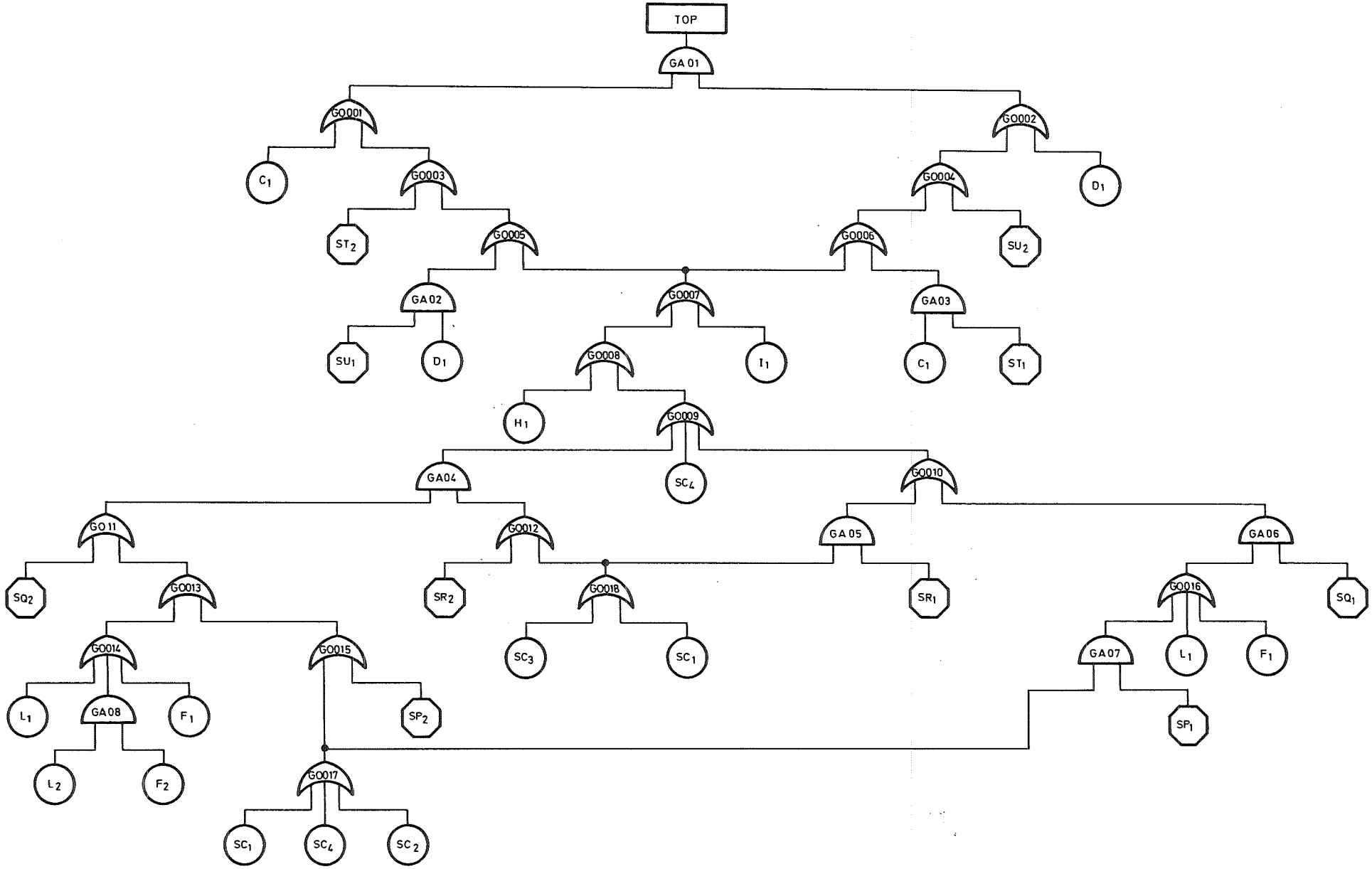


Abb. 4: Fehlerbaum eines Energieversorgungssystems (vereinfacht).

```
MA1    1011121314
MA2     2 0 1 0
FT1           ,E P P S, TOP1, AUSFALL DER INTERN.STROMVERSORGUNG
FT2       F   3   3
FT3     0 1     3.0E-5 2 0     6.7E-3 1 2     0.1
FT3       L   3   3
FT3     0 1     3.0E-5 2 0     6.7E-3 1 2     0.1
FT3      SC   5  10
FT3     1 3      .01     1 4    6.7E-4 2 1     1.1E-5 2 0     .01     3 1     5.0E-5
FT3     3 0      6.7E-4  4 0    .01     0 1    7.0E-7 0 2     5.0E-5  0 3     1.1E-5
FT3       H   2   2
FT3     1 0      5.0E-3  0 1    5.0E-7
FT3       I   2   2
FT3     1 0      2.0E-3  0 1    6.0E-7
FT3       C   2   2
FT3     0 1      2.0E-6  1 0    5.0E-3
FT3       D   2   2
FT3     0 1      2.0E-6  1 0    5.0E-3
FT3      SP   3   4DEP2
FT3     1 0      6.7E-3  2 0    6.7E-3  0 1    5.0E-6  0 2    8.0E-4
FT3     1 0      6.7E-3  2 0    6.7E-3  0 1    1.5E-5  0 2    2.0E-6
FT3      SQ   3   4DEP2
FT3     1 0      6.7E-3  2 0    6.7E-3  0 1    5.0E-6  0 2    8.0E-4
FT3     1 0      6.7E-3  2 0    6.7E-3  0 1    5.0E-6  0 2    2.0E-6
FT3      SR   3   4DEP2
FT3     1 0      6.7E-3  2 0    6.7E-3  0 1    5.0E-6  0 2    5.0E-4
FT3     1 0      6.7E-3  2 0    6.7E-3  0 1    2.0E-6  0 2    2.0E-6
FT3      ST   3   4DEP2
FT3     1 0      6.7E-3  2 0    6.7E-3  0 1    5.0E-6  0 2    1.0E-4
FT3     1 0      6.7E-3  2 0    6.7E-3  0 1    5.0E-6  0 2    2.0E-6
FT3      SU   3   4DEP2
FT3     1 0      6.7E-3  2 0    6.7E-3  0 1    5.0E-6  0 2    1.0E-4
FT3     1 0      6.7E-3  2 0    6.7E-3  0 1    5.0E-6  0 2    2.0E-6
FT4    END
FT5    GA01AND  2G001   G002
FT5    GA02AND  2  SU 1     D 1
FT5    GA03AND  2  ST 1     C 1
FT5    GA04AND  2G011   G012
FT5    GA05AND  2G018      SR 1
FT5    GA06AND  2  SQ 1 G016
FT5    GA07AND  2  SP 1 G017
FT5    GA08AND  2   L 2     F 2
FT5    G001OR   2   C 1 G003
FT5    G002OR   2   D 1 G004
FT5    G003OR   2  ST 2 G005
FT5    G004OR   2  SU 2 G006
FT5    G005OR   2GA02   G007
FT5    G006OR   2GA03   G007
FT5    G007OR   2   I 1 G008
FT5    G008OR   2   H 1 G009
FT5    G009OR   3  SC 4 GA04   G010
FT5    G010OR   2GA05   GA06
FT5    G011OR   2  SQ 2 G013
FT5    G012OR   2  SR 2 G018
FT5    G013OR   2G014   G015
FT5    G014OR   3   L 1     F 1 GA08
FT5    G016OR   3   L 1     F 1 GA07
FT5    G015OR   2  SP 2 G017
FT5    G017OR   3  SC 1   SC 2   SC 4
FT5    G018OR   2  SC 1   SC 3
FT4    END
```

Tabelle 2: Eingabedaten für MUSTAFA, Fehlerbaumdaten

```
MA3     1       SP
MA4     1       SC 1
MA4     1       SC 2
MA4     1       SC 4
MA4     1        L 1
MA4     1        F 1
MA4     2        F 2     L 2
MA5     1     ENDS
MA3     2       SP
MA4     3       SC 0     F 0     L 0
MA4     3       SC 0     F 0     L 2
MA4     3       SC 0     F 2     L 0
MA4     3       SC 3     F 0     L 0
MA4     3       SC 3     F 2     L 0
MA4     3       SC 3     F 0     L 2
MA5     1     ENDS
MA3     1       SQ
MA4     1       SC 1
MA4     1       SC 2
MA4     1       SC 4
MA4     1        L 1
MA4     1        F 1
MA4     2        F 2     L 2
MA5     1     ENDS
MA3     2       SQ
MA4     3       SC 0     F 0     L 0
MA4     3       SC 0     F 0     L 2
MA4     3       SC 0     F 2     L 0
MA4     3       SC 3     F 2     L 0
MA4     3       SC 3     F 0     L 2
MA4     3       SC 3     F 0     L 0
MA5     1     ENDS
MA3     1       SR
MA4     1       SC 1
MA4     1       SC 3
MA4     1       SC 4
MA5     1     ENDS
MA3     2       SR
MA4     1       SC 0
MA4     1       SC 2
MA5     1     ENDS
MA3     1       ST
MA4     1        C 1
MA5     1     ENDS
MA3     2       ST
MA4     1        C 0
MA5     1     ENDS
MA3     1       SU
MA4     1        D 1
MA5     1     ENDS
MA3     2       SU
MA4     1        D 0
MA5     1     ENDS
MA6     0     END
MA7   --------
```

Tabelle 2 a: Fortsetzung der Eingabe, Inhibitorfunktionen

```
MA1   1011121315 14
MA2    2 2 1 1
FT1           ,E P P S,  TOPO , SYSTEM INTAKT,STROM VÓN GEN. AN NETZ U.EIGEN
FT1   .
FT1   .
FT1   .   (DUPLIKAT DER EINGABE VON TOP1)
FT1   .
FT1   .
FT4      VOR    3  SC 1   SC 3   SC 4
FT4   MOO1AND   2   F 2    L 2
FT4      MOR    6  SC 1   SC 2   SC 4    L 1    F 1 MOO1
FT4   TOPOAND   3GA31  N   M  N   V  N
FT4   .
FT4   .
FT7   --------
      10111213161415
      10111213161514
       2 2 1 1
             ,E P P S,  TOP2 , INSELBETRIEB,STROM VON GEN. NUR F.EIGENBED.
      .
      .
      .   (DUPLIKAT DER EINGABE VON TOP1)
      .
      .
         VOR    3  SC 1   SC 3   SC 4
      MOO1AND   2   F 2    L 2
         MOR    6  SC 1   SC 2   SC 4    L 1    F 1 MOO1
      TOP2AND   3GA31  N   M  N   V
      .
      .
      --------
      10111213171415 16
       2 2 1 1
             ,E P P S,  TOP3 , FREMDEINSPEISUNG VON NETZ,GEN.AUSGEFALLEN
      .
      .
      .   (DUPLIKAT DER EINGABE VON TOP1)
      .
         VOR    3  SC 1   SC 3   SC 4
      MOO1AND   2   F 2    L 2
         MOR    6  SC 1   SC 2   SC 4    L 1    F 1 MOO1
      TOP3AND   3GA31  N       N   N
      .
      .
```

Tabelle 2 b: Fortsetzung der Eingabedaten für MUSTAFA, Eingabe der Fehlerbäume der Systemzustände TOPØ, TOP2, TOP 3.

Zu beachten ist:

- Die Fehlerbäume TOPO, TOP2, TOP3 enthalten zusätzlich die Unterbäume M und V, die entweder mit oder ohne Negationsmarkierung mit GA01 verknüpft sind.

- Die Übergänge von einem Systemzustand zu anderen konnten vollständig in einem Job gerechnet werden. Die Funktion vom TOP1 wurde auf Unit Nr. 14 die folgenden auf Unit Nr. 15, 16 und 17 gespeichert und die Übergänge zu den bis dahin bereits vorhandenen TOP-Funktionen sollten berechnet werden, wie aus der jeweils 1. Karte hervorgeht.

7.2 Fehlerbaumrechnung im Programm MUSTAMO mit horizontaler Unterteilung

Dieser und der in 7.3 gezeigte Fehlerbaum enthält ausschließlich binäre Komponenten, deren Bezeichnungen bei der Eingabe mit C001, C002, C003 ff. geschrieben wurden. Die beiden Zustände sind Cnnn 0 (intakt) und Cnnn 1 (ausgefallen). Der besseren Lesbarkeit wegen sind diese Bezeichnungen in den Abbildungen der Fehlerbäume nur verkürzt wiedergegeben: nnn in der Abbildung bedeutet Cnnn 1 (Komponente Cnnn ausgefallen).

Der Fehlerbaum in Abb. 5 ist wegen seiner starken Vermaschung mit herkömmlichen Methoden kaum zu lösen; die Anzahl aller möglichen Auflösungswege beträgt $1.62 \quad 10^{16}$, von denen jedoch nur 5630 Minimalschnitte übrig bleiben. Die übersichtliche Abbildung macht deutlich sichtbar, daß die Gatter 10X, 10Y und 10Z ein gemeinsames Territorium besitzen und daher zum Bau einer Superkomponente geeignet sind. Einige Vorüberlegungen ergaben, daß

- die Kombination $10X \wedge 10Y \wedge 10Z$, die dem Zustand Nr. 7 einer Superkomponente entspricht in der TOP-Funktion nicht vorhanden ist

- statt dessen alle Kombinationen 2 aus 3 (aus den Gattern 10X, 10Y und 10Z) Minimalschnitte der TOP-Funktion sind.

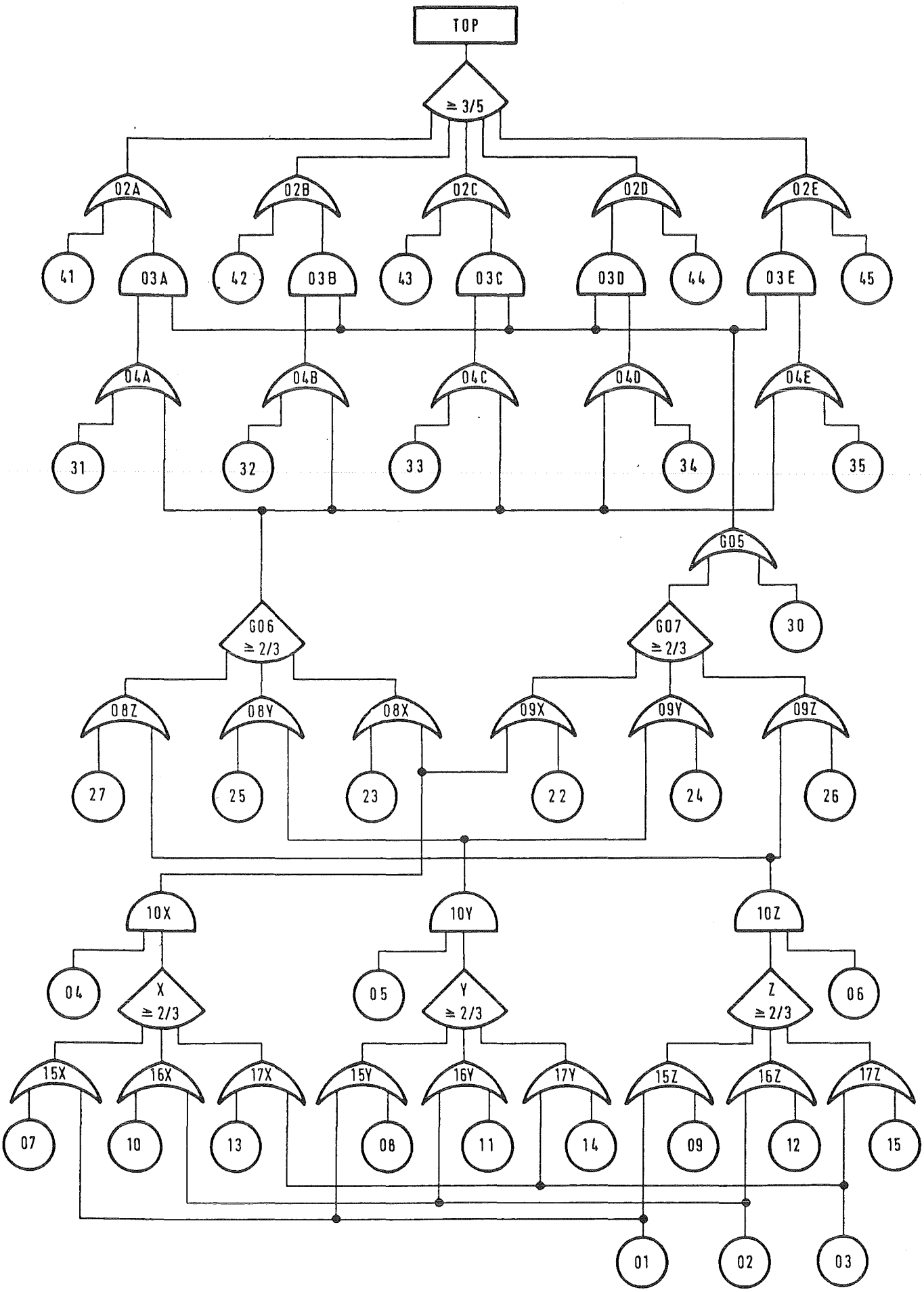


Abb. 5: Testfehlerbaum B, Original

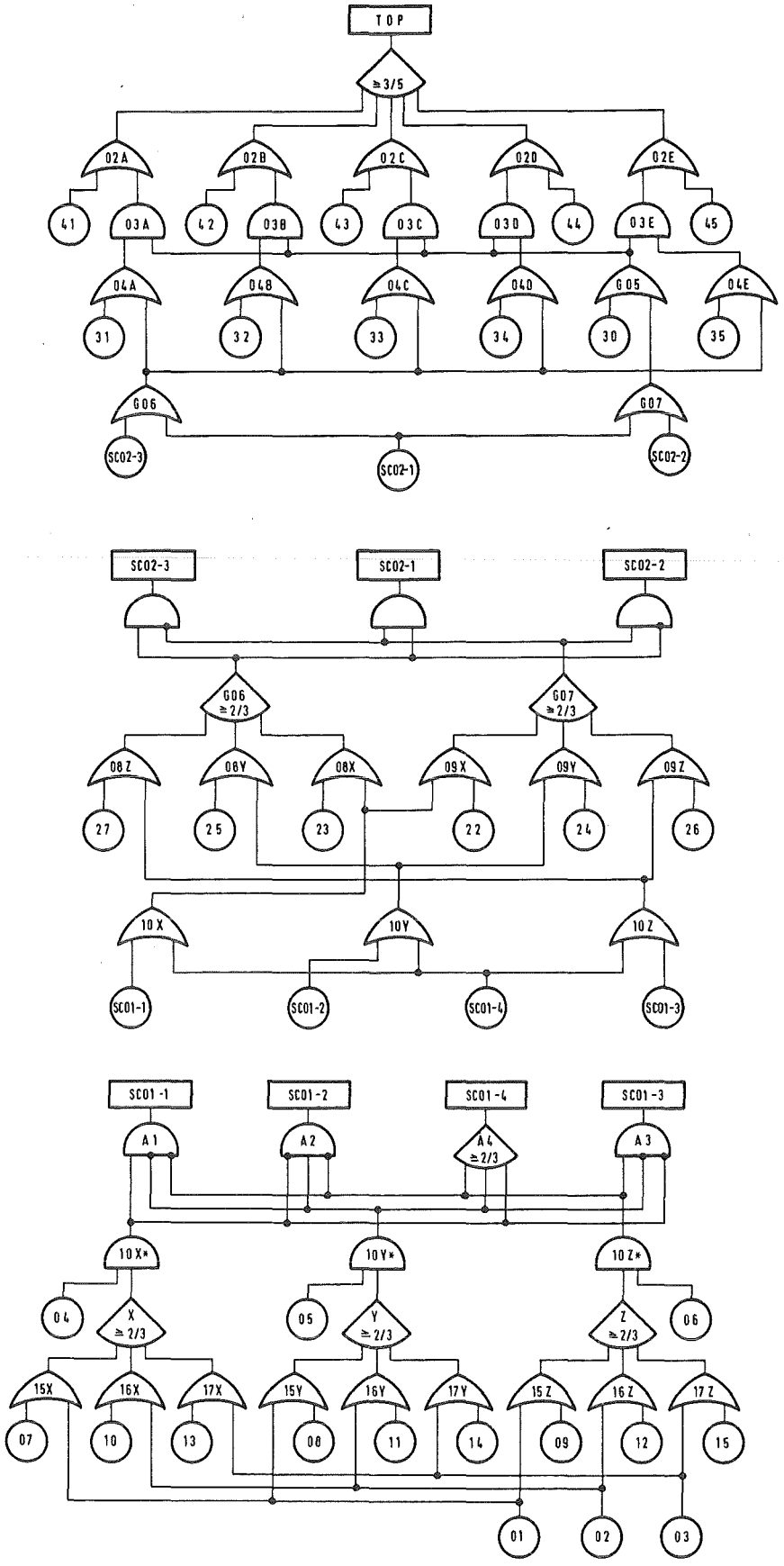


Abb. 5a: Testfehlerbaum B, zweimal horizontal geschnitten

So lag es nahe, die Zustände der Superkomponente durch Umformung des Fehlerbaums selbst zu definieren. Dabei sind die Gatter

$$A1 = 10X \cdot \overline{10Y} \cdot \overline{10Z}$$
$$A2 = \overline{10X} \cdot 10Y \cdot \overline{10Z}$$
$$A3 = \overline{10X} \cdot \overline{10Y} \cdot 10Z$$
$$A4 = (10X \cdot 10Y) + (10X \cdot 10Z) + (10Y \cdot 10Z)$$

Die Umformung ist abgebildet in Abb. 5a.
Ein zweiter Schnitt kann bei Gatter GO6 und GO7 gemacht werden. Die dabei entstehende Superkomponente hat 4 Zustände.
Tabelle 3 enthält die Eingabedaten für die Anwendung der entsprechenden Optionen des Programms MUSTAMO.

- Der erste Schnitt wird mit NOPT = 4 durchgeführt die NGT (Ausfall-) Zustände der zu bildenden Superkomponente findet das Programm in den Funktionen der Gatter A1, A2, A3 und A4.

- Der zweite Schnitt erfordert die Eingabe NOPT = 3 für eine automatisch gebaute Superkomponente. Mit NGT wird die Anzahl der Gatter angegeben aus denen die 2 NGT Zustände der Superkomponente gebildet werden. (Das Programm numeriert automatisch die Superkomponenten in der Reihenfolge der Eingabe mit SCO1 und SCO2).

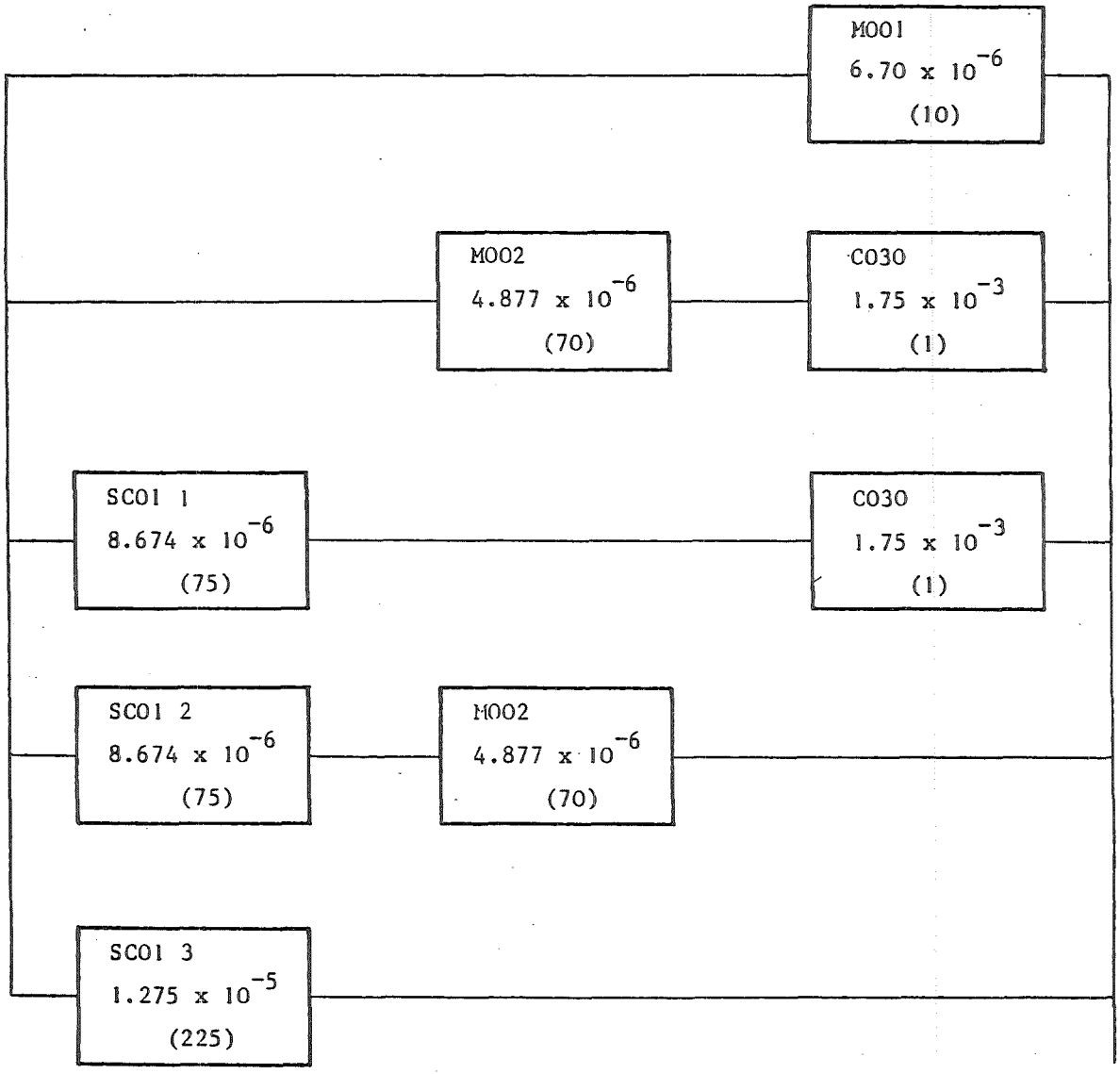
- Die Berechnung der TOP-Funktion muß in diesem Falle mit NOPT=2 erfolgen, zur Aufgliederung der TOP-Funktion wurde SCO2 und die Komponente CO30 als Schlüsselkomponente eingegeben.

Die Eingabe ist in Tabelle 3 wiedergegeben.
Das Endergebnis ist nach Ausklammerung der in der dritten Eingabekarte angegebenen Schlüsselkomponenten modularisiert. Tab. 4 zeigt das aus dieser Aufgliederung resultierende Blockdiagramm. Die Rechenzeit betrug 7,4 Sekunden. Zum Vergleich wurde der Fehlerbaum mit nur einem horizontalen Schnitt gerechnet. Die Rechnung mit nur einem Schnitt bei den Gattern GO6 und GO7 benötigte 48 Sek.

NOPT	NGT	NKE	NHLT	KF	Gatter (IGT (I), I = 1, NGT)	Schlüsselkomponente (KEY(I), I = 1, NKE)	Stop-Gatter (WH(I), I = 1, WHLT)
4	3	0	0	0	10X, 10Y, 10Z	entfällt	entfällt
3	2	0	0	0	G06, G07	"	"
2	1	2	0	0	TOP	SC02, C030	

Tabelle 3: Eingabe für die Berechnung des Fehlerbaums Abb. 4

Tabelle 4: Blockdiagramm des Ergebnisses der Berechnung des Fehlerbaums aus Abb. 5



	6.70×10^{-6} (10)
	8.535×10^{-9} (70)
	1.518×10^{-8} (75)
	4.23×10^{-11} (5250)
	1.275×10^{-5} (225)
Summe der Eintrittswahrscheinlichkeit:	1.947×10^{-5}
Summe der Minimalschnitte:	(5630)

7.3 Fehlerbaumberechnung mit dem Splitting-Verfahren

Der Fehlerbaum Abb. 6, wurde graphisch so dargestellt, daß die Wege der Komponenten C001 bis C009, die in alle Unterbäume eingehen, sowie die starken Vermaschungen im Mittelteil des Fehlerbaums deutlich sichtbar werden. Abb. 7 zeigt den Originalfehlerbaum, Abb. 6 eine Umformung, die mit einer kleinen Handrechnung für die Gatter G125, G126, G127 sowie durch Aufteilung der Gatter G41 bis G44 leicht bewerkstelligt werden konnte. Dieser Fehlerbaum wurde den Berechnungen zugrunde gelegt. Die Komponente C001 nimmt eine Sonderstellung ein. Da sie in je einem Und-Gatter mit allen übrigen Schlüsselkomponenten C002 bis C009 verknüpft ist, kann und muß sie nur in denjenigen Minimalschnitten anwesend sen, in denen mindestens eine der genannten Schlüsselkomponente anwesend ist.

Es wurde zunächst versucht lediglich mit horizontalen Schnitten zu unterteilen und berechnen (Rechnung Nr. 1 und Nr. 2) danach wurde das vertikale Splitting-Verfahren getestet (Rechnung 3 und 4); die Tabellen 5 bis 8 enthalten die Eingabekarten der verschiedenen Rechnungen. Als Schnittstelle für die horizontale Unterteilung bieten sich die Gatter G106, G125 und G127 an, sowie die Paare G88 und G89, G90 und G91, G98 und G99. Die für diese Gatter berechneten Module sowie Superkomponenten können nur vorläufige sein (Option 12, 13, 14). Die Umformung bei den Gattern G98 und G99 zur Eingabe einer selbst-definierten Superkomponente ist in als Beispiel erwähnt.

Rechnung Nr. 1 und Nr. 2 unterscheiden sich durch den Grad der Auflösung, wobei in Rechnung Nr. 1 mit KF=1 die Teilmodularisierung der Zustände der Superkomponente und mitNOPT = 10 Teilmodularisierung bei der Berechnung der Top-Funktion ausgeschaltet wurde. Recnung Nr. 1 benötigte 101 Minuten, Rechnung Nr. 2 38 Minuten. Rechnung Nr. 3 und Nr. 4 wurden mit dem Splittingverfahren durchgeführt. Bei der Rechnung wurde durch Deklarierung der Komponenten C006 bis C009 G127, G88 und G89, G90 und G91 zu logisch unabhängigen Modulen und Superkomponenten. Die Komponente C001 wurde als deterministische Komponente 1. Art mit 1 für den Ausfallzustand eingegeben, mit den oben genannten vier Komponenten (deterministisch 2. Art) wurde der Fehlerbaum in 16 einander ausschließende Bäume gespalten.

In Rechnung Nr. 4 wurden alle 8 Schlüsselkomponenten C002 bis C009 als deterministisch 2. Art gekennzeichnet. Dies ergab 256 Bäume durch Splitting, wobei einzelne Kombinationen Minimalschnitte waren, bzw. redundante Terme.

Die Komponente C001 wurde in Rechnung Nr. 3 und Nr. 4 als deterministisch 1. Art mit ihrem Ausfallzustand C001 1 eins gesetzt, weswegen das Ergebnis der Rechnung mit der Eintrittswahrscheinlichkeit von C001 1 multipliziert werden mußte, dazu mußte die Eintrittswahrscheinlichkeit des einen Moduls, der mit allen Intaktzuständen der Schlüsselkomponenten verbunden war addiert werden.

Die Rechenzeiten der einzelnen Rechnungen betrugen:

Rechnung Nr. 1	103 Min.	20000 Zeilen	Ausgabe
Rechnung Nr. 2	38 Min.	16000 "	"
Rechnung Nr. 3	11 Min.	30000 "	"
Rechnung Nr. 4	1 Min.	48000 "	"

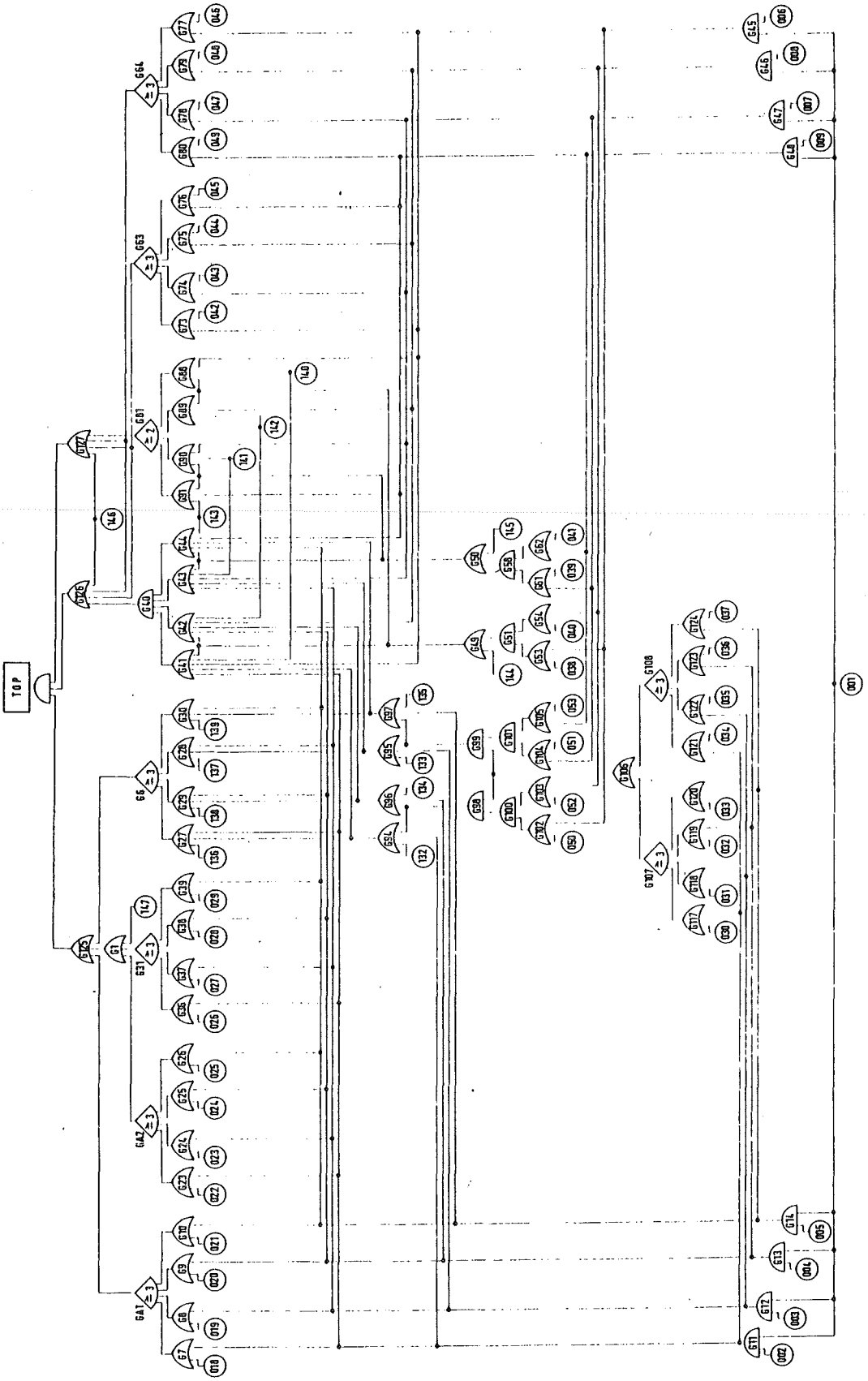


Abb. 6: Testfehlerbaum A, Original

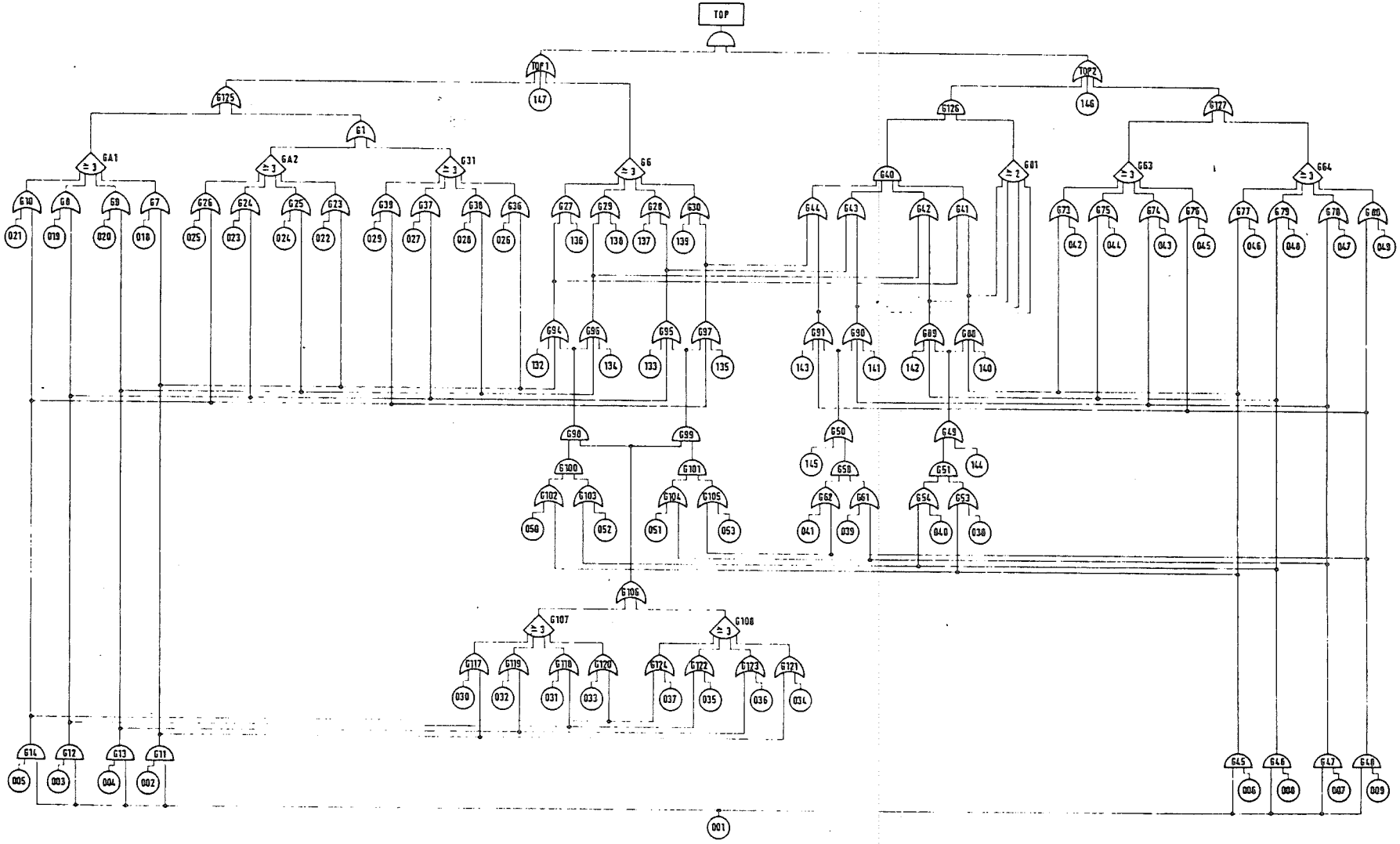


Abb. 7: Testfehlerbaum A, umgeformt

Tabelle 5: Fehlerbaum A, Rechnung Nr. 1

Karte M01						Karte M02	Karte M03
NOPT	NGT	NKE	NHLT	KF	(IGT(I), I = 1, NGT)	Schlüsselkomp. (KEY(I), I = 1, NKE)	Stop-Gatter (NH(I), I = 1, NHLT)
12	1	5	0	0	G127	C001, C006, C007, C008, C009	entfällt
12	1	5	0	0	G125	C001, C002, C003, C004, C005	"
2	1	5	0	0	G106	C001, C002, C003, C004, C005	"
14	3	9	0	1	X, Y, Z	C001, C002, C003, C004, C005, C006, C007, C008, C009	"
13	2	5	0	1	G88, G89	C001, C006, C007, C008, C009	"
13	2	5	0	1	G90, G91	C001, C006, C007, C008, C009	"
10	1	14	4	0	TOP	C001, C002, C003, C004, C005, C006, C007, C008, C009, G125, G127, SC01, SC02, SC03	G94, G95, G96, G97

Tabelle 6: Fehlerbaum A, Rechnung Nr. 2

Karte M01						Karte M02	Karte M03
NOPT	NGT	NKE	NHLT	KF	(IGT (I), I = 1, NGT)	Schlüsselkomponenten (KEY (I), I = 1, NKE)	Stop-Gatter (NH (I), I = 1 NHLT)
12	1	5	0	0	G127	C001, C006, C007, C008, C009	entfällt
12	1	5	0	0	G125	C001, C002, C003, C004, C005	"
2	1	5	0	0	G106	C001, C002, C003, C004, C005	"
14	3	9	0	0	X, Y, Z	C001, C002, C003, C004, C005, C006, C007, C007, C008, C009	"
13	2	5	0	0	G88, G89	C001, C006, C007, C008, C009	"
13	2	5	0	0	G90, G91	C001, C006, C007, C008, C009	"
20	1	14	4	0	TOP	C001, C002, C003, C004, C005, C006, C007, C008, C009, G125, G127, SC01, SC02, SC03	G94, G95, G96, G97

Tabelle 7: Fehlerbaum, Rechnung 3

Deterministisch 1. Art: C001 1 = (Ausfallzustand der Komp. C001)

Mit '+DET' gekennzeichnete deterministische Komponenten 2. Art: C006, C007, C008, C009

Karte M01						Karte M02	Karte M03
NOPT	NGT	NKE	NHLT	KF	(IGT (I), I = 1, NKE)	Schlüsselkomponente (KEY (I), I = 1 NKE)	Stop-Gatter, (NH (I), I = 1 NHLT)
1	1	0	0	0	G127	entfällt	entfällt
2	1	4	0	0	G125	C002, C003, C004, C005	"
2	1	4	0	0	G106	C002, C003, C004, C005	"
3	2	0	0	0	G88, G89	entfällt	"
3	2	0	0	0	G90, G91		
2	1	4	4	0	TOP	C002, C003, C004, C005	G94, G95, G96, G97

Tabelle 8: Fehlerbaum A, Rechnung Nr. 4

Deterministisch 1. Art: C001 1 (=Ausfallzustand C001) = 1

Deterministisch 2. Art, mit '+DET' gekennzeichnet: C002, C003, C004, C005, C006, C007, C008, C009

Karte M01						Karte M02	Karte M03
NOPT	NGT	NKE	NHLT	KF	(IGT (I), I = 1, NGT)	Schlüsselkomponente (KEY (I), I = 1 NKE)	Stop-Gatter (NH (I), I = 1, NHLT)
1	1	0	0	0	G127	entfällt	entfällt
1	1	0	0	0	G125	"	"
1	1	0	0	0	G106	"	"
3	2	0	0	0	G88, G89	"	"
3	2	0	0	0	G90, G91	"	"
2	1	1	4	0	TOP	G106	G94, G95, G96, G97

Literaturverzeichnis:

/1/ L. Caldarola, A. Wickenhäuser:
"The Karlsruhe computer program for evaluation of the avialability and reliability of complex repairable systems".
Nucl. Engineering and Design 43 (1977) pp. 463-470

/2/ L. Caldarola, A. Wickenhäuser:
"Recent Advancements in Fault Tree Methodology at Karlsruhe".
Int. Conf. on Nucl. Systems Rel., Eng. and Risk Assessment, Gattinburg, 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"
Nucl. Engng. and Design 58 (1980) 127-139

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

/7/ L. Caldarola:
"Grundlagen der Boole'schen Algebra mit beschränkten Variablen"
KfK 2915, EUR 6405d, Febr. 1980

/8/ L. Caldarola, A. Wickenhäuser:
"The Boolean Algebra with Restricted Variables as a Tool for Fault Tree Modularization"
KfK 3190, EUR 7056e

/9/ J.B. Fussell u.a.:
"MOCUS - A computer program to obtain minimal sets from fault trees"
Aerojet Nucl. Company, March 1974

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

/11/ R.G. Benndt:
"On the analysis of Fault Trees"
IEEE Transctions on Reliability, Vol. R-24, No. 3, Aug.1975

/12/ G. Weber:
unveröffentlichter Bericht, 1981

/13/ A. Wickenhäuser:
unveröffentlichter Bericht, 1980

/14/ A. Wickenhäuser:
unveröffentlichter Bericht, 1982

/15/ L. Caldarola, H. Schnauder, A. Wickenhäuser:
unveröffentlichter Bericht, 1982