第34卷 第13期 计 算 机 工 程 2008年7月
Vol.34 No.13 Computer Engineering July 2008
·软件技术与数据库· 文章编号：1000—3428(2008)13—0067—03 文献标识码：A 中图分类号：TP202.1

一种新的故障树定性分析方法

陈越洲[1]，谭 琳[1]，邢维艳[2]，刘 东[3]
(1. 中南林业科技大学计算机学院，长沙 410004；2. 中国华阴兵器试验中心，华阴 714200；3. 国防科技大学计算机学院，长沙 410073)

摘 要：提出基于割序集的分析方法以研究故障树顶事件发生时基本事件的动态行为。利用顺序失效符表示事件的顺序失效关系，并将静态门和动态门转化为顺序失效表达式来描述故障树中各种门的动态行为，利用顺序失效表达式构建故障树的割序集。结合实例阐述故障树割序集生成算法的流程。该算法将失效行为表示为长度小于系统中部件个数的有序部件序列，为研究故障树提供了一种新的定性分析方法。
关键词：故障树；割序集；定性分析

Qualitative Analysis Approach for Fault Tree

CHEN Yue-zhou[1], TAN Lin[1], XING Wei-yan[2], LIU Dong[3]
(1. Computer Science College, Central South University of Forestry and Technology, Changsha 410004;
2. China Huayin Ordnance Test Center, Huayin 714200;
3. School of Computer, National University of Defense Technology, Changsha 410073)

【Abstract】 In order to investigate the dynamic behaviors of basic events in fault trees, a new method based on Cut Sequence Set(CSS) is presented. Cut sequence includes a set of basic events that induce their top events, and the basic events in a cut sequence fail in a fixed order. CSS is the set of all cut sequences of a fault tree. The paper defines Sequential Failure Symbol(SFS) to describe the sequential failure of events. SFS is used to translate static gates and dynamic gates into Sequential Failure Expressions(SFE), which represents the sequential behaviors of the events in these gates. The failure of top event is described by some SFEs, namely cut sequences, which constitute CSS. A CSS generation algorithm is put forward, and an example illustrates the application and advantages of the method. The approach represents system failure behaviors by using a sequence with some components, which provides a new qualitative analysis approach to study fault trees.
【Key words】 fault tree; Cut Sequence Set(CSS); qualitative analysis

1 概述

故障树分析方法始于20世纪60年代[1]，由于具有直观、易于理解等优点，逐渐被可靠性分析工程师和研究人员采用，并对故障树开展了多种理论和应用研究。随着技术的发展，原有的静态故障树因为无法分析带有顺序失效、冗余、共因失效等特性的系统而不能适应新的情况。

在此基础上，文献[2]提出了动态故障树的概念，引入FDEP, CSP, PAND, SEQ等新的动态门来分析带有功能相关、储备、优先失效和顺序失效等特性的系统。对于动态故障树，描述顶事件发生的不仅是多个基本事件的积之和(即割集)，还包含基本事件的顺序失效关系。这种顺序失效的动态行为给动态故障树的分析造成了困难。对于由动态故障树描述的动态系统，其部件失效的动态行为是系统可靠性分析的研究热点。文献[3]将动态系统中部件的顺序失效引入到传统的最小割集概念中，提出了动态故障树最小割序(minimum cut sequence)的概念，但没有给出生成割序的具体过程。文献[4]提出了顺序失效逻辑的概率模型SFLM，利用多重积分定量分析可维修动态系统的可靠性；SFLM在列出系统的多重积分公式时，需要事先生成系统的顺序失效逻辑，其本质是故障树的割序，虽然体现了将动态系统转换为顺序失效逻辑的思想，但没有给出求解一般动态故障树的完整方法。文献[5]借助动态系统的演化方程(evolution equations)，利用Petri网模型的计数器技术分析了动态系统的顺序失效。该方法只针对PAND门的顺序失效关系，没有考虑冷储备、温储备等非工作冗余系统的情况。此外，Markov模型中的失效链是从系统初始状态到失效状态的状态链，可以用于确定动态系统的全部割序，但随着基本事件的增多，Markov模型将面临组合爆炸的问题。

综合割序、顺序失效逻辑的研究结果，本文认为割序描述了系统在发生失效时部件的动态行为，而动态故障树模型能够直观地描述系统和部件的失效关系，并且具有较小的计算复杂度，因此，利用动态故障树进行有关割序的研究是一种有效的定性分析方法。由于目前并没有完整的生成动态故障树割序表示的方法，因此本文研究了动态故障树的割序生成过程。

2 故障树割序集的生成算法

2.1 顺序失效符与故障树的割序集

由于动态故障树中部件的失效顺序决定了顶事件的发生，因此动态故障树的顶事件可以由多个包含顺序失效关系的基本事件描述，由此定义了顺序失效符(Sequential Failure Symbol, SFS)来表示事件(不要求必须是基本事件)的失效顺序，记为“→”。

SFS用于连接2个事件，它表明符号左边的事件在符号

基金项目：国家自然科学基金资助项目(60573103)
作者简介：陈越洲(1968－)，男，讲师、硕士，主研方向：分布式计算，数据库容灾技术；谭 琳，副教授；邢维艳，硕士；刘 东，博士研究生
收稿日期：2007-07-30 **E-mail：**yueweichen@tom.com

右边的事件之前失效。SFS 及其连接的 2 个事件共同构成顺序失效表达式(Sequential Failure Expression, SFE)，例如“$A\rightarrow B$”。多个事件可以用 SFS 串接，如“$A\rightarrow B\rightarrow C$”表明 A, B, C 的失效顺序依次是 A, B, C。

动态故障树的一个割序描述了导致顶事件发生的一种可能情况，而所有割序的集合构成割序集(Cut Sequence Set, CSS)。由于动态故障树一般具有多个割序，因此其割序集最终表示为多个 SFE 的逻辑或。事实上，静态故障树的割集也可以转换为含有失效顺序符的割序集，例如割集$\{AB\}$对应的割序集为$\{(A\rightarrow B)\cup(B\rightarrow A)\}$。因此，本文综合动态故障树的割序集与静态故障树的割集表示法，统一用割序集描述一般故障树(静态和动态)顶事件发生时基本事件的动态行为。

要用SFE表示顶事件的发生，需要分析静态门和动态门中输入/输出事件的动态行为，并将各种门转化为相应的SFE，本文称这个过程为SFS转化[2]。

2.2 静态门和动态门的 SFS 转化

2.2.1 AND 门

AND 门包含一个输出事件和多个输入事件，只有当输入事件全部发生时，才会产生输出。具有 n 个输入的 AND 门可以转化为 $n!$个 SFE 的逻辑或。例如，结构函数为 $\Phi=ABC$ 的故障树转化为 SFE 形式的割序集共有 3! = 6 个 SFE，即

$$CSS=\{(A\rightarrow B\rightarrow C)\cup(A\rightarrow C\rightarrow B)\cup(B\rightarrow A\rightarrow C)\cup(B\rightarrow C\rightarrow A)\cup (C\rightarrow A\rightarrow B)\cup(C\rightarrow B\rightarrow A)\}$$

故障树的割序集中一般具有多个割序，表示为多个 SFE 的逻辑或，因此，本文不考虑 OR 门的 SFS 转化。此外，由于 k/n 门可以等价为 AND/OR 门的结构，因此也不再分析 k/n 门的 SFS 转化问题。

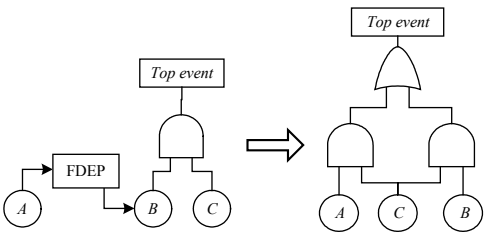
2.2.2 PAND 门

PAND 门是 AND 门的拓展，它在 AND 门的基础上增加一个附加条件，这个条件规定了输入事件的发生次序。例如，对于具有一个输出事件和 2 个输入事件(A 和 B)的 PAND 门，当 A 先于 B 发生时，输出事件发生；如果 A 和 B 都没有发生或 B 在 A 之前发生，则输出事件不发生。在用 SFS 转化 PAND 门时，仅须将 PAND 门的输入事件作为 SFE 的一个事件逐一列写。例如对于结构函数为 $\Phi=A$ PAND B PAND C 的故障树，用 SFE 表示的割序集为 $CSS=\{A\rightarrow B\rightarrow C\}$。

2.2.3 FDEP 门

FDEP 门由触发输入事件、非相关输出事件和若干相关基本事件构成，当触发事件发生时，直接产生输出，而所有相关部件随即成为不可达或无法使用。FDEP 门可以分解为其他逻辑门的组合。

图 1(a)是某一系统的动态故障树，A 的失效将会导致 B 的失效，而 B, C 均失效时会导致顶事件发生。该故障树可以转化为图 1(b)所示的静态故障树，因此，该故障树的割序集为 $CSS=\{(A\rightarrow C)\cup(C\rightarrow A)\cup(C\rightarrow B)\cup(B\rightarrow C)\}$。



(a)某系统的动态故障树　　(b)某系统的静态故障树

图 1　FDEP 门的 SFS 转化

2.2.4 CSP 门

CSP 门具有一个初始输入和若干替补输入。初始输入是指在系统开始工作时就处于工作状态的部件，替补输入作为冷储备，在工作部件失效后，逐个依次替补。图 2 是一个将 CSP 门进行 SFS 转化的实例。在其中的故障树中，B 作为冷储备由 A 和 C 共享，则该故障树的割序集为 $CSS=\{(A\rightarrow(B\cup C))\cup(C\rightarrow(A\cup B))\}$。

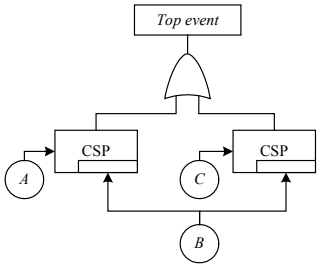


图 2　CSP门的SFS转化

2.2.5 WSP 门

WSP 门体现了实际系统中的温储备特性。WSP 门具有一个初始输入和若干个替补输入，只有当所有输入事件全部发生时，WSP 门才会产生输出，但由于 WSP 门允许替补输入在初始输入之前发生，因此 WSP 门的 SFS 转化过程与 AND 门相同。

2.2.6 SEQ 门

SEQ 门具有一个输出事件和若干个输入事件，并要求输入事件必须严格按照规定的顺序失效；直到所有输入事件发生后，输出事件才会发生。SEQ 门与 PAND 门的区别在于：SEQ 门的输入事件只能按照规定的顺序逐个发生，而 PAND 门中的输入事件可按任意次序发生。在对 SEQ 门进行 SFS 转化时，仅须将 SEQ 门的输入依次作为 SFE 的一个事件列写。

2.3 SFS 的推演规则

在对静态门和动态门进行 SFS 转化之后，故障树顶事件的发生表示为 SFE 的多种逻辑组合，如 $CSS=\{(A\rightarrow(B\cup C))\cup(C\rightarrow(A\cup B))\}$。本文称这种不规则的表达式为 CSS 的“初级形式”，而将只含有“$\neg$”、“$\rightarrow$”和基本事件符号的 CSS 形式称为“标准形式”或“标准割序集”。为了得到最终的标准割序集，本文给出如下的 SFS 推演规则，其中，◇表示∪或∩。

(1)分配律

1)$x\rightarrow(y\diamond z)\Leftrightarrow(x\rightarrow y)\diamond(x\rightarrow z)$

2)$(x\diamond y)\rightarrow z\Leftrightarrow(x\rightarrow z)\diamond(y\rightarrow z)$

3)$x\rightarrow(y\rightarrow z)\Leftrightarrow(x\rightarrow y)\rightarrow z\Leftrightarrow x\rightarrow y\rightarrow z$

(2)结合律

$x\rightarrow x$ 或$(x\rightarrow y)\cup x\Leftrightarrow x$

(3)吸收律

1)$(x\rightarrow y)\cup y\Leftrightarrow y$(若 y 不是 x 的冷储备)

2)$(x\rightarrow y)\cup y\Leftrightarrow x\rightarrow y$(若 y 是 x 的冷储备)

3)$(x\rightarrow y)\cap x$ 或$(x\rightarrow y)\cap y\Leftrightarrow x\rightarrow y$

(4)与分配律

$(x_1\rightarrow x_2\rightarrow\cdots\rightarrow x_m)\cap(y_1\rightarrow y_2\rightarrow\cdots\rightarrow y_n)$, $m\le n$, $x_i\neq y_j$ $(1\le i\le m, 1\le j\le n)\Leftrightarrow(x_1\rightarrow x_2\rightarrow\cdots\rightarrow x_m\rightarrow y_1\rightarrow y_2\rightarrow\cdots\rightarrow y_n)\cup(x_1\rightarrow x_2\rightarrow\cdots\rightarrow x_{m-1}\rightarrow y_1\rightarrow x_m\rightarrow y_2\rightarrow\cdots\rightarrow y_n)\cup\cdots\cup(y_1\rightarrow y_2\rightarrow\cdots\rightarrow y_n\rightarrow x_1\rightarrow x_2\rightarrow\cdots\rightarrow x_m)$

(5)不存在

1)$x\rightarrow y_1\rightarrow y_2\rightarrow\cdots\rightarrow y_n\rightarrow x$

2)$x\rightarrow\neg x$

2.4 割序集生成算法

本文的故障树割序集生成算法流程如图 3 所示。

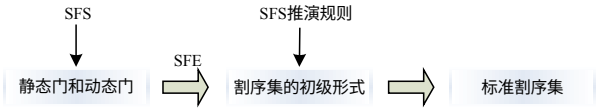


图 3　割序集生成算法的流程

故障树的割序类似于 Markov 模型中的一条失效链。随着基本事件的增多，Markov 模型将面临组合爆炸问题，而割序集生成算法也存在这样的隐患。但与 Markov 模型中的失效链不同，本文提出的割序不考虑割序包含的事件之外的其他事件(这些事件可能为正常或者失效)，从而降低了计算复杂度。

3　实例分析

本节用一个实例——HDS(Hypothetical Dynamic System)阐述割序集生成算法的详细过程。HDS 具有 4 个部件 A, B, C 和 S。C 作为 A 和 B 的冷储备，在 A 或 B 失效时替换首先失效的部件，但如果 S 在 A 或 B 之前失效，则 C 无法起到替换部件的作用，从而导致 C 失效。系统的正常工作要求 A, B, C 中至少有 2 个正常工作。

HDS 的故障树如图 4 所示。

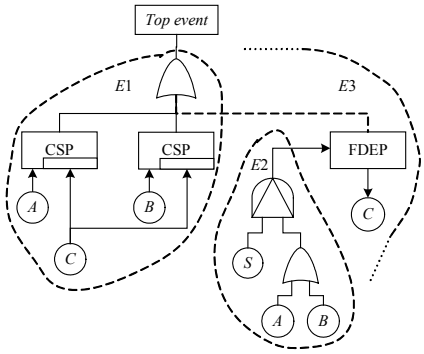


图 4　HDS 的故障树

(1)动态门的 SFS 转化

1)2 个 CSP 门可以转化为

$E1 = \{(A\rightarrow(B\cup C))\cup(B\rightarrow(C\cup A))\}$

2)PAND 门可转化为

$E2=\{S\rightarrow(A\cup B)\}$

3)FDEP 门可转化为

$E3=\{E2\cap(A\cup B)\}$

(2)生成割序集的初级形式

$CSS=\{E1\cup E3\}=$

$\{((A\rightarrow(B\cup C))\cup(B\rightarrow(C\cup A)))\cup((S\rightarrow(A\cup B))\cap(A\cup B))\}$

(3)得到标准割序集

$$\begin{aligned}CSS=\ &\{((A\rightarrow(B\cup C))\cup(B\rightarrow(C\cup A)))\cup((S\rightarrow(A\cup B))\cap(A\cup B))\}=\\ &\{((A\rightarrow B)\cup(A\rightarrow C)\cup(B\rightarrow C)\cup(B\rightarrow A))\cup\\ &((S\rightarrow(A\cup B))\cap(A\cup B))\}= && \text{[分配律]}\\ &\{((A\rightarrow B)\cup(A\rightarrow C)\cup(B\rightarrow C)\cup(B\rightarrow A))\cup\\ &((S\rightarrow(A\cup B))\rightarrow(A\cup B))\cup\\ &((A\cup B)\rightarrow(S\rightarrow(A\cup B)))\}= && \text{[与分配律]}\\ &\{((A\rightarrow B)\cup(A\rightarrow C)\cup(B\rightarrow C)\cup(B\rightarrow A))\cup\\ &(S\rightarrow(A\cup B)\rightarrow(A\cup B))\}= && \text{[结合律]}\\ &\{((A\rightarrow B)\cup(A\rightarrow C)\cup(B\rightarrow C)\cup(B\rightarrow A))\cup\\ &(S\rightarrow(A\cup B))\}= && \text{[吸收律]}\\ &\{(A\rightarrow B)\cup(A\rightarrow C)\cup(B\rightarrow C)\cup(B\rightarrow A)\cup\\ &(S\rightarrow A)\cup(S\rightarrow B)\} && \text{[分配律]}\end{aligned}$$

由该标准割序集可以定性分析出 HDS 所有可能的失效模式：A 在 B 之前失效；A 在 C 之前失效；B 在 C 之前失效；B 在 A 之前失效；S 在 A 之前失效；S 在 B 之前失效。

4　结束语

本文系统地提出了故障树割序集的生成方法，并用实例阐述了割序集的生成过程。其中的割序集生成算法是故障树研究中的一种新方法，为系统可靠性分析提供了一种新的定性分析手段。

参考文献

[1] Watson H. Launch Control Safety Study[R]. Murray Hill, USA: Bell Telephone Laboratories, 1961.

[2] Dugan J B, Bavuso S, Boyd M. Dynamic Fault Tree Models for Fault Tolerant Computer Systems[J]. IEEE Transactions on Reliability, 1992, 41(3): 363-377.

[3] Tang Zhihua, Dugan J B. Minimal Cut Set/Sequence Generation for Dynamic Fault Trees[C]//Proceedings of Annual Reliability and Maintainability Symposium. [S. l.]: IEEE Press, 2004.

[4] Long W, Sato Y, Horigone M. Quantification of Sequential Failure Logic for Fault Tree Analysis[J]. Reliability Engineering & System Safety, 2000, 67(3): 269-274.

[5] Adamyan A, He David. Sequential Failure Analysis Using Counters of Petri Net Models[J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems and Humans, 2003, 33(1): 1-11.

（上接第 57 页）

参考文献

[1] Hair J F, Anderson R E, Tatham R C, et al. Multivariate Data Analysis[M]. 4th ed. New Jersey, USA: Prentice Hall Inc., 1998: 56-64.

[2] Han Jiawei. Data Mining: Concepts and Techniques[M]. San Francisco, USA: Morgan Kaufmann Publishers, 2001.

[3] Lakshminarayan K, Harp S, Goldman R, et al. Imputation of Missing Data Using Machine Learning Techniques[C]//Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining. Menlo Park, CA, USA: AAAI Press, 1996: 140-145.

[4] Lakshminarayan K, Harp S A, Goldman R, et al. Imputation of Missing Data Using Machine Learning Techniques[C]//Proc. of KDD'96. Portland, USA: [s. n.], 1996: 140-146.

[5] Ragel A, Cremilleux B. MVC——A Preprocessing Method to Deal with Missing Values[J]. Knowledge-based Systems, 1999, 12(5): 285-291.