基于时序描述逻辑的故障树分析方法研究

司 佳,朱羿全,马 琳
(南京航空航天大学 计算机科学与技术学院,江苏 南京 210016)

摘 要:故障树分析法是工业界常用的安全分析方法之一。然而由于其非形式化方法的局限性,难以对软件故障进行形式化验证,更难以描述嵌入式实时系统中事件之间的时序逻辑关系。因此,提出了一种基于时序描述逻辑的故障树分析方法,以解决故障树难以对时序关系进行描述以及难以形式化验证的问题。首先,通过时序描述逻辑对故障树进行时序特征的扩充与规约;其次抽取出用描述逻辑表示的软件安全属性;最后对软件系统进行安全属性建模并通过模型检测工具 SPIN 形式化验证软件系统是否满足这些属性。以某一机载控制系统环境输入模块为案例,对该案例进行故障树分析和建模并给出该案例的待验证安全属性以及实验分析结果。结果表明,提出的方法是有效的和可行的。

关键词:故障树分析;时序描述逻辑;安全属性;形式化验证
中图分类号:TP311 **文献标识码**:A **文章编号**:1673-629X(2017)12-0089-04
doi:10.3969/j.issn.1673-629X.2017.12.020

Research on Fault Tree Analysis Based on Temporal Description Logic

SI Jia,ZHU Yi-quan,MA Lin
(School of Computer Science and Technology,Nanjing University of Aeronautics and Astronautics,
Nanjing 210016,China)

Abstract:Fault Tree Analysis (FTA) is one of safety analysis methods which is commonly used in industry. However,as the limitation of its non-formal method,it is difficult to be formal verification of software fault and even to describe the temporal logic relation between events in embedded real-time system. Therefore,in order to solve the problem,a formal fault tree analysis based on Temporal Description Logic (TDL) is proposed. Firstly,the fault tree is extended and constrained in temporal sequence characteristic by TDL. Secondly,safety attributes of software are extracted in the representation of TDL. At last,the safety attributes modeling is carried out in software system which is verified whether to meet these attributes or not by SPIN,a model checking tool. A case of environment input module of airborne control system is given where the analysis and modeling of fault tree is conducted,and its security attributes to be checked and experimental results are achieved. It is showed that the proposed method is effective and feasible.

Key words:fault tree analysis;temporal description logic;safety attributes;formal verification

0 引 言

目前故障树分析法(Fault Tree Analysis,FTA)是学术界和工业界应用最为广泛的安全性分析方法之一[1]。它首先分析了造成故障结果的所有原因,并给出了失效结果与失效原因之间的因果关系链,进而协助检测系统的设计错误、安全缺陷以及薄弱环节[2]。

由于传统的故障树分析法一般采用非形式化方法进行描述,缺乏精确语义,因此难以对软件故障进行形式化验证。另一方面,由于其非形式化方法的局限性,使传统的故障树缺乏对具有时序特征的安全关键系统精确的描述方法[3]。

时序描述逻辑(Temporal Description Logic,TDL)是描述逻辑的时序扩展,既具有描述逻辑很强的表达和可判断推理能力,又包含大量时序算子,可以对动态领域的知识进行描述[4]。

文中提出一种基于时序描述逻辑的故障树分析方法,旨在将时序描述逻辑与传统故障树分析方法相结合。首先利用时序描述逻辑对故障树进行时序扩充,而后从中抽取出描述软件安全属性的时序逻辑公式,最终实现软件系统安全性的形式化验证。

收稿日期:2016-11-30 **修回日期**:2017-04-05 **网络出版时间**:2017-08-01
基金项目:国家自然科学基金资助项目(61272083,61100034,61170043);中央高校基本科研业务费专项资金(NS2014099);江苏省自然科学基金青年基金项目(BK20130812)
作者简介:司 佳(1992-),男,硕士研究生,研究方向为需求工程、安全工程。
网络出版地址:http://kns.cnki.net/kcms/detail/61.1450.TP.20170801.1554.054.html

1 相关工作

如何将传统的故障树分析技术与形式化分析验证方法相互结合,已成为国内外学者研究的热点之一。文献[5]采用 PLTLP 对故障树的语义进行形式化,然而 PLTLP 的表达能力有限,无法描述复杂的时序性质;文献[6]通过 Z 语言对故障树进行了形式化规约,但由于 Z 语言的局限性,其缺乏对故障树中安全属性提取和正确性验证的方法;文献[7]为了能够表达更多的故障,已经对软件系统进行规约,其对现有的时序故障树进行了 TCTL 扩充;文献[8]提出一种用线性时序逻辑规约的时序故障树 LTFT,从中提取出描述系统时序故障的安全属性,用于软件安全性验证。

2 时序描述逻辑

描述逻辑(Description Logic,DL)是一种基于对象的知识表示形式化方法[9]。它建立在概念(concept)和角色(role)之上。概念是一类事物的抽象,通常用 $A,B,C,D\cdots$ 表示;而角色则用来刻画事物之间的各种联系和关系,通常用 $P,Q,R,S\cdots$ P 表示。基础描述逻辑知识详见文献[9]。

时序描述逻辑主要是在基础的描述逻辑上扩展了 4 个时序逻辑算子[10-11]:○、□、◇和 ∪,详细说明如表 1 所示。

表 1 时序描述逻辑扩展的时序算子

符号	说明
(at the next moment)	下一时刻
□(always in the future)	所有未来时刻
◇(eventually)	某个或某些未来时刻
∪ (until)	直到

2.1 TDL 扩展的语法和语义

(1)TDL 扩展的语法。

$C,D::=\square C\mid \square C\mid \Diamond C\mid C\cup D$

$R_1,R_2::=\square R_1\mid \square R_1\mid \Diamond R_1\mid R_1\cup R_2$

$\varphi,\psi::=\square\varphi\mid \square\varphi\mid \Diamond\varphi\mid \varphi\cup\psi$

其中,C 和 D 表示为描述逻辑中的概念;R_1 和 R_2 表示角色;φ 和 ψ 表示公式; 、 、◇为一目算子;∪为二目算子,用来表示左式一直为真,直到右式为真。

(2)TDL 扩展的语义。

假设时间流 $\zeta=<T,<>$,其中 T 表示时间点的非空集合,$<$ 表示 T 中时间点之间严格的线性时序二元关系。TDL 扩展部分的语义可描述如下[12]:

$$(\ \ C)^I=\{<t,a>\mid \exists t_1.\ t\wedge<t_1,a>\in C^I\wedge \neg\exists t_2.\ t<t_2<t_1\}$$

$$(\Diamond C)^I=\{<t,a>\mid \exists t_1.\ t\leqslant t_1\wedge<t_1,a>\in C^I\}$$

$$(\square C)^I=\{<t,a>\mid \forall t_1.\ t\leqslant t_1\wedge<t_1,a>\in C^I\}$$

$$(C\cup D)^I=\{<t,a>\mid \exists t_1.\ t\leqslant t_1\wedge<t_1,a>\in D^I\wedge\forall t_2.\ t<t_2<t_1\rightarrow<t_2,a>\in C^I\}$$

2.2 TDL 扩展后的可满足性

由于文献[13]已证明,描述逻辑的推理问题均可以规约为可满足性问题,可满足性的定义如下:

定义 1(可满足性):时序描述逻辑模型 $M=<T,<,I>$,其中 T 表示时间点的集合;$<$表示 T 中时间点之间的时序关系;I 表示一个解释函数。对任意 $i\in T$,均有 $I(i)=<\Delta,R_0^{I(i)},\cdots,C_0^{I(i)},\cdots,a_0^{I(i)}\cdots>$,其中非空集合 Δ 是模型 M 的论域,$R_0^{I(i)}$,$C_0^{I(i)}$,$a_0^{I(i)}$ 分别解释为论域的二元关系、子集和一个元素。对任意概念 C,D 和公式 φ,ψ 均有:

(1)$(M,i)\models C=D\Leftrightarrow C^{I(i)}=D^{I(i)}$

(2)$(M,i)\models a:C\Leftrightarrow a^{I(i)}\in C^{I(i)}$

(3)$(M,i)\models aRb\Leftrightarrow a^{I(i)}Rb^{I(i)}$

(4)$(M,i)\models\varphi\cup\psi\Leftrightarrow\exists j>i,\ (M,i)\models\psi\&\&\forall k.\ (i\leqslant k<j\rightarrow(M,k)\models\varphi)$

(5)$(M,i)\models\ \ \varphi\Leftrightarrow(M,i+1)\models\varphi$

(6)$(M,i)\models\Diamond\varphi\Leftrightarrow\exists j>i,(M,j)\models\varphi$

(7)$(M,i)\models\square\varphi\Leftrightarrow\forall j>i,(M,j)\models\varphi$

(8)$(M,i)\models\varphi\rightarrow\psi\Leftrightarrow(M,i)\models\varphi\rightarrow(M,i)\models\psi$

3 基于 TDL 的形式化故障树构建

文中工作的核心是构建带有形式化信息的故障树,并从中抽取支持形式化验证的软件安全属性。因此,首先利用时序描述逻辑对传统故障树进行扩充,生成支持安全属性规约的能够形式化表示的时序描述逻辑故障树。

时序描述逻辑故障树保留了传统故障树中的全部逻辑门和事件[14],在此基础上引入与 TDL 时序算子相对应的○、□、◇和 ∪ 时序逻辑门:

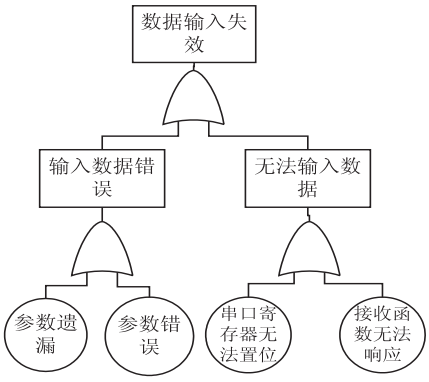
(1)“○”门分别有一个输出事件以及一个输入事件。当“○”门的输入故障事件在系统运行的下一个时刻为永真,“○”门的输出故障事件发生。

(2)“□”门分别有一个输出事件和一个输入事件。若“□”的输入故障事件在整个系统运行时间内为永真,则“□”门的输出故障事件发生。

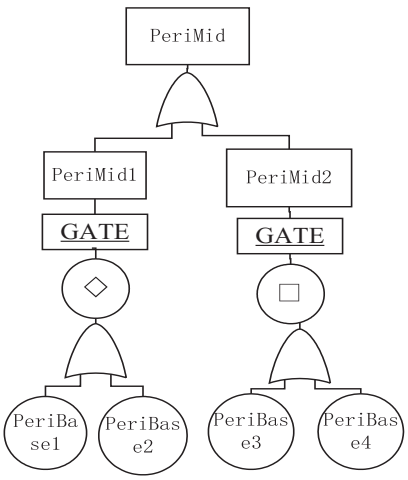
(3)“◇”门分别有一个输出事件和一个输入事件。若“◇”门下的输入故障事件在系统运行时间内的某一时刻为真,则“◇”门的输出故障事件发生。

(4)对于条件“$A\cup B$”而言,此“∪”门有两个输入事件 A 和 B 以及一个输出事件 C,只有满足条件“$A\cup B$(即事件 A 为真,直到事件 B 为真)”时,输出事件 C 才发生。

引入以上时序逻辑门后,基于 TDL 的故障树构建原则仍遵循传统故障树的基本原则[14]。在传统的故障树构建过程中,不允许逻辑门之间直接相连。而由于引进了时序逻辑门,为了正确地构建故障树,文中允许时序逻辑门与普通逻辑门以及时序逻辑门之间的直接相连。举例说明如图 1 所示。



(a)传统的故障树构建



(b)基于 TDL 的形式化故障树构建

图 1　传统故障树与形式化故障树对比

4　软件安全属性的提取及验证

4.1　安全属性的提取

通过对传统故障树的扩展和形式化规约构建出能够描述时序关系的形式化故障树。在此基础上,研究形式化故障树安全属性的提取过程,给出一个基于“上行法”的故障树安全属性提取算法,为软件系统模型的形式化验证提供证据。“上行法”是传统故障树分析方法中一种常规分析法,其基本思想是从故障树的基本事件开始,由下到上进行规约。基于“上行法”提取形式化故障树安全属性的流程如下:

步骤一:自底向上遍历 TDL 扩充的故障树,根据下层的 TDL 公式以及事件之间的逻辑门关系,给出新的 TDL 公式来描述上一层逻辑门的输出或输出事件;

步骤二:判断当前 TDL 公式是否是最简形式,若不是,则执行步骤四;

步骤三:判断当前事件是否为顶事件,若是,则执行步骤五,否则执行步骤一;

步骤四:对 TDL 公式进行约简,并执行步骤三;

步骤五:得到描述顶事件的描述逻辑公式,即为系统需要被验证的安全属性。

4.2　安全属性的形式化验证

形式化验证方法的主要思路是使用数学的公式、定理和系统来验证一个系统的正确性[15]。目前常用的形式化验证方法主要包括模型检测和定理证明。文中采用模型检测方法,运用 Promela 语言对软件系统形式化建模,验证所提取的安全属性。

方法的具体实施流程如图 2 所示。首先,用时序描述逻辑对传统故障树进行扩充和形式化规约,形成能够形式化描述系统时序故障的形式化故障树。根据 3.1 节安全属性提取算法,从形式化故障树中提取出描述系统安全属性的 TDL 公式;其次,根据 TDL 公式,利用 Promela 进行系统建模,得到待验证的系统模型;最后,将待验证的安全属性和系统模型导入 SPIN 中进行验证,并根据 SPIN 提供的验证结果对系统进行安全性分析。

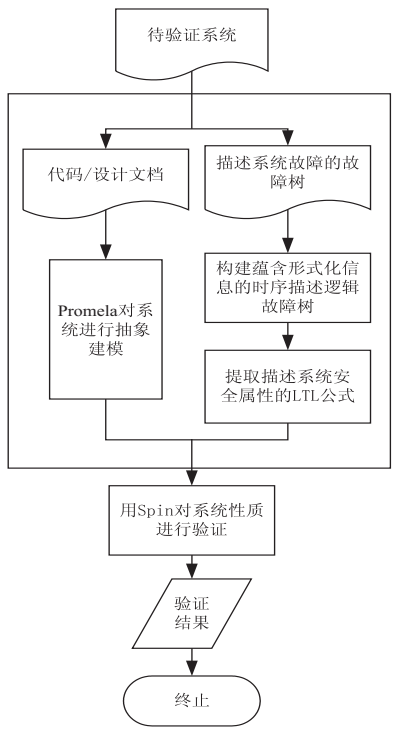


图 2　安全属性形式化验证流程

5　实例分析

为了说明文中方法的有效性,更清晰地演示方法的运用过程,给出某一机载控制系统环境输入模块代码的安全性验证与分析案例。

环境输入模块主要负责采集不断变化的外界环境参数,其失效模式的初始化故障树描述如图 3 所示。

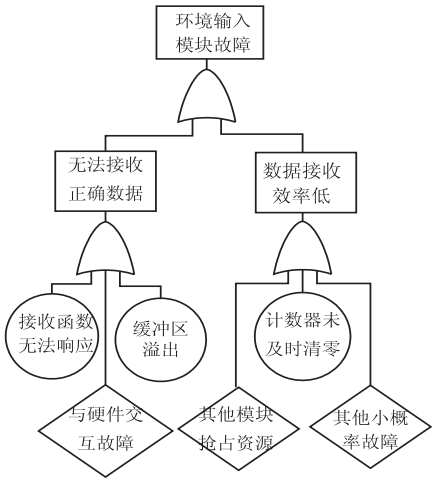


图 3 环境输入模块故障树

根据文中方法对该故障树进行形式化扩充,如图 4 所示。

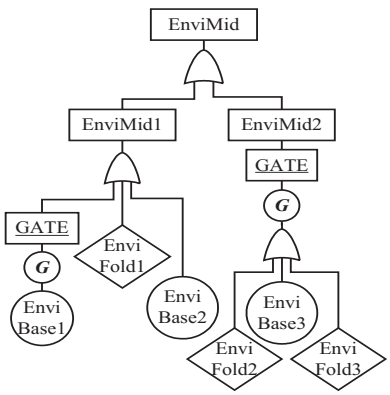


图 4 基于 TDL 扩展后的形式化故障树

再利用 3.1 节提出的 calTDL 算法对扩展后的形式化故障树进行安全属性提取,得出 3 条通过 TDL 表达式描述的待验证的安全属性:

(1)割集为{Sub-Tree1},表示该模块接受函数始终无法响应,其对应的 TDL 表达式为:□¬(g_SciaRX_Ready == 1);

(2)割集为{EnviBase2},表示为缓冲区溢出,其对应 TDL 表达式为:◇(g_dRegLength > MAX_REG);

(3)割集为{Sub-Tree2},表示计数器始终无法及时清零,其对应的 TDL 表达式为:□¬(SCIRXCounter == 0)。

再利用模型检测工具 SPIN 对环境输入模块进行形式化验证。SPIN 是基于 Promela 语言的主流模型检测工具之一[16]。

输入一:Promela 语言描述的环境输入模块;

输入二:待验证的 3 条安全属性;

输出结果:环境输入模块是否满足这 3 条安全属性。

在对 TDL 公式 □¬(SCIRXCounter == 0) 进行验证时,出现如图 5 所示的结果,即程序在执行到 8100 深度处才得以终止。

```
Warning:never claim+accept labels requires  a flag to fully ve
Warning:for p.o. reduction to be valid the never claim must be st
<never claims generated form TDL formulate are stutter-invariant>
Pan:1 end state in claim reached<at depth 8100>
Pan:wrote Envi.pml.trail
press any key to continue
```

图 5 SPIN 验证结果

由图中结果可以说明,由于环境输入模块的计数器无法及时清零,导致程序多次执行了空语句,浪费了宝贵的计算资源。

6 结束语

针对传统故障树分析法缺乏形式化语义、无法解决时序问题的缺陷,提出一种基于时序描述逻辑的故障树分析方法。将时序描述逻辑与故障树相结合,并给出了具体的形式化故障树扩展方法;在此基础上,给出了一种提取故障树安全属性的算法,并将安全属性用于系统的模型检测。下一步工作的重点是研究如何对故障树本身的正确性进行验证。

参考文献:

[1] Lee W S, Grosh D, Tillman F A, et al. Fault tree analysis, methods, and application: a review[J]. IEEE Transactions on Reliability, 1985, 34(3): 194-203.

[2] 马 琳, 黄志球, 徐丙凤, 等. 支持模型检测的故障树生成方法研究[J]. 计算机与数字工程, 2013, 41(2): 257-260.

[3] 黄志球, 徐丙凤, 阚双龙, 等. 嵌入式机载软件安全性分析标准、方法及工具研究综述[J]. 软件学报, 2014, 25(2): 200-218.

[4] 杨海波. 基于时序描述逻辑的 UML 状态图语义研究[D]. 兰州: 兰州理工大学, 2010.

[5] Palshikar G K. Temporal fault trees[J]. Information and Software Technology, 2002, 44(3): 137-150.

[6] Coppit D, Sullivan K J, Dugan J B. Formal semantics of models for computational engineering: a case study on dynamic fault trees[C]//Proceedings of 11th international symposium on software reliability engineering. [s. l.]: IEEE, 2000: 270-282.

[7] 刘 磊. 软件时序故障树建模与分析技术研究[D]. 长沙: 国防科学技术大学, 2011.

[8] 马 琳. 基于故障树的航电软件系统安全性验证方法研究[D]. 南京: 南京航空航天大学, 2012.

[9] Badder F, Nutt W. The description logic hand book: theory, implementation and application[M]. Cambridge: Cambridge University Press, 2002.

[10] Lutz C, Wolter F, Zakharyaschev M. Temporal description logic: a survey[C]//Proceedings of the 15th international symposium on temporal representation and reasoning. Washington, DC: IEEE, 2008: 3-14.

(下转第 97 页)

表 1　方案对比

方案	请求	响应	本地储存
Ateniese 等提出的方案[6]	$4\ Z_N^*$ (4 096 bit)	$2\ Z_N^*$ (2 048 bit)	$O(1)$
Sebe 提出的方案[8]	$2\ Z_N^*$ (2 048 bit)	$1\ Z_N^*$ (1 024 bit)	$O(n)$
ID-PPDP	$3\ Z_q^*$ (480 bit)	$1\ G + 1\ Z_q^*$ (480 bit)	$O(1)$

5　结束语

文中提出了身份基代理远程数据持有证明,给出了系统模型和安全模型,然后设计了基于双线性对的高效的协议,并且通过安全和性能分析证明了该协议的安全性和高效性。

参考文献:

[1] 冯登国,张　敏,张　妍,等. 云计算安全研究[J]. 软件学报,2011,22(1):71-83.

[2] 张玉清,王晓菲,刘雪峰,等. 云计算环境安全综述[J]. 软件学报,2016,27(6):1328-1348.

[3] 冯朝胜,秦志光,袁丁云. 数据安全存储技术[J]. 计算机学报,2015,38(1):150-163.

[4] 胡德敏,余　星. 云存储服务中支持动态数据完整性检测方案[J]. 计算机应用研究,2014,31(10):3056-3060.

[5] 沈文婷,于　佳,杨光洋,等. 具有私钥可恢复能力的云存储完整性检测方案[J]. 软件学报,2016,27(6):1451-1462.

[6] Ateniese G,Burns R,Curtmola R,et al. Provable data possession at untrusted stores[C]//Proceedings of 14th ACM conference on computer and communication security. [s. l.]:ACM,2007:598-609.

[7] Ateniese G,Dipietro R,Mancini L V,et al. Scalable and efficient provable data possession[C]//Proceedings of fourth international conference on security and privacy in communication networks. [s. l.]:[s. n.],2008.

[8] Sebe F,Domingo-Ferrer J,Martinez-Balleste A,et al. Efficient remote data integrity checking in critical information infrastructures[J]. IEEE Transactions on Knowledge and Data Engineering,2008,20(8):1034-1038.

[9] Erway C C,Kupcu A,Papamanthou C,et al. Dynamic provable data possession[C]//Proceedings of 16th ACM conference on computer and communication security. [s. l.]:ACM,2009:213-222.

[10] Wang C,Wang Q,Ren K,et al. Privacy-preserving public auditing for data storage security in cloud computing[C]//29th IEEE international conference on computer communications. [s. l.]:IEEE,2010:1-9.

[11] Zhu Y,Hu H,Ahn G J,et al. Cooperative provable data possession for integrity verification in multicloud storage[J]. IEEE Transactions on Parallel and Distributed Systems,2012,23(12):2231-2244.

[12] Wang H. Proxy provable data possession in public clouds[J]. IEEE Transactions on Services Computing,2013,6(4):551-559.

[13] Wang H. Identity-based distributed provable data possession in multi-cloud storage[J]. IEEE Transactions on Services Computing,2015,8(2):328-340.

[14] Mambo M,Usuda K,Okamoto E. Proxy signatures for delegating signing operation[C]//Proceedings of third ACM conference on computer and communications security. [s. l.]:ACM,1996:48-57.

[15] Hwang M,Lu J,Lin E. A practical (t, n) threshold proxy signature scheme based on the RSA cryptosystem[J]. IEEE Transactions on Knowledge and Data Engineering,2003,15(6):1552-1560.

[16] Libert B,Vergnaud D. Unidirectional chosen-ciphertext secure proxy re-encryption[J]. IEEE Transactions on Information Theory,2011,57(3):1786-1802.

[17] Pack S,Rutagemwa H,Shen X,et al. Proxy-based wireless data access algorithms in mobile hotspots[J]. IEEE Transactions on Vehicular Technology,2008,57(5):3165-3177.

[18] Boneh D,Franklin M. Identity-based encryption from the weil pairing[C]//Proceedings of 21st annual international cryptology conference. [s. l.]:[s. n.],2001:213-229.

[19] Miyaji A,Nakabayashi M,Takano S. New explicit conditions of elliptic curve traces for fr-reduction[J]. IEICE Transactions on Fundamentals Electronics Communications & Computer Sciences,2001,84(5):1234-1243.

[20] Kumanduri R. Number theory with computer applications[M]. [s. l.]:Prentice Hall,1998.

[21] Miller V. Uses of elliptic curves in cryptography[C]//Proceedings of fifth annual international cryptology conference. [s. l.]:[s. n.],1985:417-426.

[22] Vanstone S. Responses to NISTs proposal[J]. Communication of ACM,1992,35:50-52.

(上接第 92 页)

[11] 李　明,刘士仪,年福忠. 基于时序描述逻辑的 Web 服务本体语言过程模型语义[J]. 计算机应用,2013,33(1):266-269.

[12] 荣先球. 基于描述逻辑的 UML 行为图的形式化研究[D]. 兰州:兰州理工大学,2012.

[13] Baader F,Bauer A,Lippmann M. Runtime verification using a temporal description logic[C]//Proceedings of 7th international conference on frontiers of combining systems. Berlin:Springer-Verlag,2009:149-164.

[14] Vesely W E,Goldberg F F,Roberts N H,et al. Fault tree handbook[M]. United States:U. S. Nuclear Regulatory Commission,1981.

[15] Schamann J M. Automated theorem proving in software engineering[M]. Berlin:Springer-Verlag,2001.

[16] Huth M,Ryan M. Logic in computer science:modeling and reasoning about systems[M]. Cambridge,UK:Cambridge University Press,2004.