第39卷第5期 原子能科学技术 Vol. 39, No. 5
2005年9月 Atomic Energy Science and Technology Sep. 2005

适用于动态概率安全评价的故障树逻辑简化方法

杨 宇[1,2],刘晓平[1,2],刘 萍[1],吴宜灿[1]
(1. 中国科学院 等离子体物理研究所,安徽 合肥 230031;
2. 合肥工业大学 计算机与信息学院,安徽 合肥 230009)

摘要:对故障树进行逻辑简化将有效提高分析计算的速度。根据故障树结构特点,提出了基于贪心算法的故障树逻辑简化方法。该方法已编程实现,并采用实际系统的故障树进行了测试。实践证明,该方法可大幅度提高分析求解速度,同时,该方法所采取的贪心策略又可运用在故障树分析的其他方面。

关键词:概率安全评价;故障树;贪心算法
中图分类号:TL364.5 **文献标识码**:A **文章编号**:1000-6931(2005)05-0433-05

Fault Tree Logical Reduction Strategy for Living Probabilistic Safety Assessment

YANG Yu[1,2], LIU Xiao-ping[1,2], LIU Ping[1], WU Yi-can[1]
(1. *Institute of Plasma Physics, Chinese Academy of Sciences, Hefei* 230031, *China*;
2. *School of Computer & Information, Hefei University of Technology, Hefei* 230009, *China*)

Abstract: Logical reduction can speed up fault tree analysis effectively. A fault tree logical reduction strategy based on greedy algorithm is presented. The strategy was implemented and tested by analyzing practical system fault trees. Facts prove that the strategy can speed up fault tree analysis greatly, meanwhile, the greedy thinking of the strategy can be applied to other aspects of fault tree analysis.

Key words: probabilistic safety assessment; fault tree; greedy algorithm

近年来,动态概率安全评价(Living PSA)成为核电站安全分析与评价中的热点之一,其关键是速度问题。在核电站的PSA分析中,均使用故障树分析法来计算系统的无效度。一般故障树求解的运算规模随着故障树中的结点(门和基本事件)数的增加而呈指数增长;同时,大型故障树在实际的系统安全分析中并不鲜见[1]。因此,开发快速准确的故障树求解方法是核电站Living PSA的一个重要课题。对故障树进行逻辑简化,减少其结点数,可以有效地降低求解的规模,大大提高求解的速度,从而满足Living PSA的要求。

收稿日期:2004-03-11;**修回日期**:2004-05-01
基金项目:中国科学院百人计划资助项目;国家自然科学基金资助项目(60273044)
作者简介:杨 宇(1978—),男,安徽池州人,硕士研究生,软件技术与理论专业

故障树的逻辑简化还没有一个系统的论述[2~4]。本文针对故障树的逻辑简化的特点，应用贪心算法的思想，提出基于贪心算法的故障树逻辑简化方法。

1 贪心算法简介

贪心算法是一种常用的求解最优化问题简单、迅速的方法。该算法将全局优化问题分解成一系列的子问题，按照构建步骤，依据局部优化准则，对这些子问题求局部最优解，逐步向符合全局优化准则的全局最优解靠近。由于贪心算法原理简单，效果良好，因此，在最优化领域得到广泛的使用。

适于使用贪心算法解决的问题一般有两个特点[5]。

1) 贪心选择性质

1个全局最优解可通过做局部最优选择(即贪心策略)来达到。贪心算法中，每一步所做的总是当前最佳的选择，然后再解决做了该选择之后所出现的子问题。贪心算法所做的当前选择可能要依赖于已经做出的所有选择，但不依赖于未做出的选择和子问题的解。因此，贪心算法通常是自顶向下执行，一个一个地做出贪心选择，不断地将给定的问题归约为更小的问题。

2) 最优子结构

如果某个最优化问题的一个最优解包含了其子问题最优解，则称该问题呈现出最优子结构。这个性质是对贪心算法的可应用性进行评价的关键一点。

需指出，贪心算法虽可给出最优化问题的最优解，但它只能找到近似的最优解[6]。而在工程实践中，近似最优解往往就足够了。

2 基于贪心算法的故障树逻辑简化方法

2.1 前提

贪心算法是解决组合优化问题很好的求解策略[6]，而故障树逻辑简化是1个典型的组合优化问题。

不失一般性，设 a 为故障树中的某个结点，N_a 表示以 a 为顶结点的子树中结点的个数(包括 a)，则 N_a 可以表示成如下递归形式：

$$N_a = 1 + \sum_{i=1}^{n} N_{b_i} \qquad (1)$$

式中：n 为 a 的扇出数(a 为基本事件结点时，$n=0$)；b_i 为 a 的第 i 个输入结点。

式(1)为1个和式，因此，可以得到如下两个推论。

1) 当 N_{b_i} 都取得最小值时，N_a 取得最小值。

针对故障树自下而上反复应用推论1，可得当故障树中的所有子树的结点数达到最小，则整个故障树的结点数必然达到最小，因此，故障树的逻辑简化问题具有贪心选择性质。

2) 当 N_a 取得最小值时，N_{b_i} 都取得最小值。

针对故障树自上而下反复应用推论2，可以得到，当整个故障树的结点数达到最小时，则故障树中的所有子树的结点数必然达到最小，因此，故障树逻辑简化问题满足最优子结构性质。

综上所述，故障树逻辑简化问题满足应用贪心算法的两个性质，因此，可以用贪心算法来解决该问题。

2.2 方法简述

对于故障树的逻辑简化，其全局优化准则是故障树中的结点数最少，局部优化准则是每个门下的子树结点数最少。方法的基本思想是，以某种顺序遍历故障树，依次对每个门下的子树进行局部结构优化，最终达到整个故障树的结构优化。

1) 局部结构优化

要求解局部最优解，就必须定义局部优化规则。基于文献[2~4]和自己的分析经验，提出了如下子树优化规则。

(1) 收缩规则

收缩规则1：若相邻两层门类型相同，则可合并。该规则对应的布尔运算规则为：

$a \cup (b \cup c) = a \cup b \cup c$，$a \cap (b \cap c) = a \cap b \cap c$。

收缩规则2：在同一个门的输入中，相同的底事件可以合并。该规则对应的布尔运算规则为：$a \cup a = a$，$a \cap a = a$。

收缩规则3：如果1个门只有1个输入，则这个门可以删除，其输入上移。

(2) 删除规则

应用前提是故障树中的门的类型是隔层相同(相邻层门的类型相异，这可以通过对故障树应用收缩规则来实现)。

删除规则 1:如果在故障树的某个门 G1 下有基本事件 B,同时,在以门 G1 为顶的子树的偶数层上的某个门 G2 下也有基本事件 B,则以门 G2 为顶的子树可以删去。该规则对应的布尔运算规则为:

$a \cup (a \cap b) \cup c = a \cup c, a \cap (a \cup b) \cap c = a \cap c$。

删除规则 2:如果在故障树的某个门 G1 下有基本事件 B,同时,在以门 G1 为顶的子树的奇数层上的某个门 G2 下也有基本事件 B,则可以删去 G2 下面的基本事件 B。该规则对应的布尔运算规则为:

$$a \cup (b \cap (a \cup c \cup d)) = a \cup (b \cap (c \cup d)),$$
$$a \cap (b \cup (a \cap c \cap d)) = a \cap (b \cup (c \cap d))。$$

(3) 提取规则

提取规则即相同底事件处在一层的若干个门中,可将该事件提取出来,其应用前提是故障树中的门的类型隔层相同。该规则对应的布尔运算规则为:

$$(a \cap b \cap c) \cup (a \cap d \cap e) = a \cap ((b \cap c) \cup (d \cap e)),$$
$$(a \cup b \cup c) \cap (a \cup d \cup e) = a \cap ((b \cup c) \cap (d \cup e))。$$

值得注意的是,对于某个子树,使用上面的优化规则,不一定能达到所谓的最优局部结构(Best-Local)[4],只能说是较好的局部结构(Better-Local)[4],但是,可以认为这一结构是现有条件下所能作出的最优解,这在贪心算法中是允许的。

2) 构建步聚的选择

前面指出,对于故障树逻辑简化问题,其构建步骤是以某种顺序遍历故障树,依次对每个门下的子树求最优解。对于 1 个树形结构,其遍历的顺序有 2 个:深度遍历和广度遍历。深度遍历是一种带回溯的遍历方法,其本质是自下而上地求解问题,对于某个母结点,其求解依赖于子结点们的求解。而广度遍历本质上是一个自上而下的求解过程,对母结点求解完毕,再对子结点依次求解,母结点的求解依赖于母结点的母结点的求解,而与子结点的求解无关,这种特性非常符合贪心算法的要求,因此,本文采用广度遍历策略。

2.3 方法实现

方法的流程图示于图 1,其基本流程是依照广度遍历的顺序,对每个遍历到的门,采用局部最优求解规则,对以该门为顶的子树进行简化。在对故障树进行了一轮简化后,故障树中某些局部的优化又可能导致其他局部不满足贪心性质(局部最优),所以,要对故障树反复使用该方法,直到某一轮简化没有对任何局部实施简化动作为止。此外,如前所述,删除规则和提取规则的使用前提是树中门的类型隔层相异,因此,在使用这两个规则前后都要对整个故障树或是某个子树应用吸收规则。

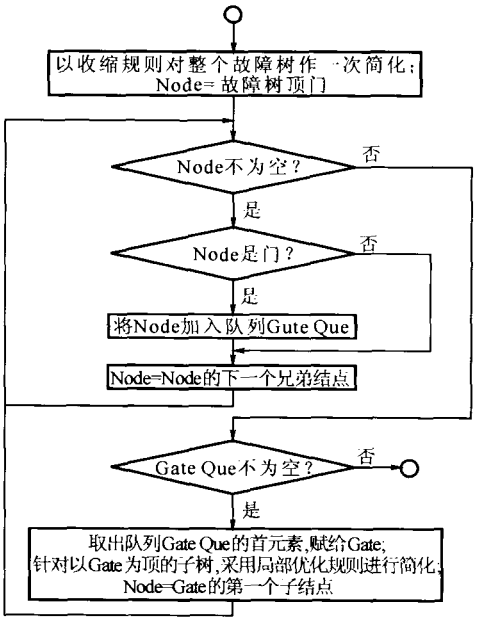

图 1　故障树简化流程图

Fig. 1　Flow chart of fault tree reduction

需指出,由于本文的方法对于局部优化只能达到较好的局部结构(Better-Local),因此,最后的全局解也只能是个较好的全局结构(Better-Global)[4],但对于工程应用来说已经足够了。

3　应用和分析

图 2 给出了一个应用本文方法的示例。图 2a 的结点数为 70,图 2b 的结点数为 42,结点数减少 40%。

作者在为中国科学院等离子体物理研究所开发的可靠性分析软件 LPSA 的故障树分析部分中已经实现了该方法,并使用大亚湾核电

站的3个子系统故障树对其进行了测试，表1给出了测试数据。在表1中，硬件平台：CPU为Pentium 4 2.0 G，内存为512M；软件平台：Windows 2000 Professional sp3。可以看出，该方法使用很小的代价大幅度提高了故障树的求解速度。

值得一提的是，通常在对故障树进行分析计算前还有一些预处理操作，如当故障树中含有如表决门(KN)和异或门(XOR)这样的复杂门时，需要用与门、或门和非门来替换这些门；对于故障树中的各个门上“逻辑非”操作，需要运用De. Morgan律下移到基本事件中；当故障树中含有房形事件时，还需要根据房形事件的值对故障树进行剪枝。这些预处理过程都可以采用类似本文提出的策略：根据问题特点，制定局部优化规则，对故障树进行广度遍历，对各个

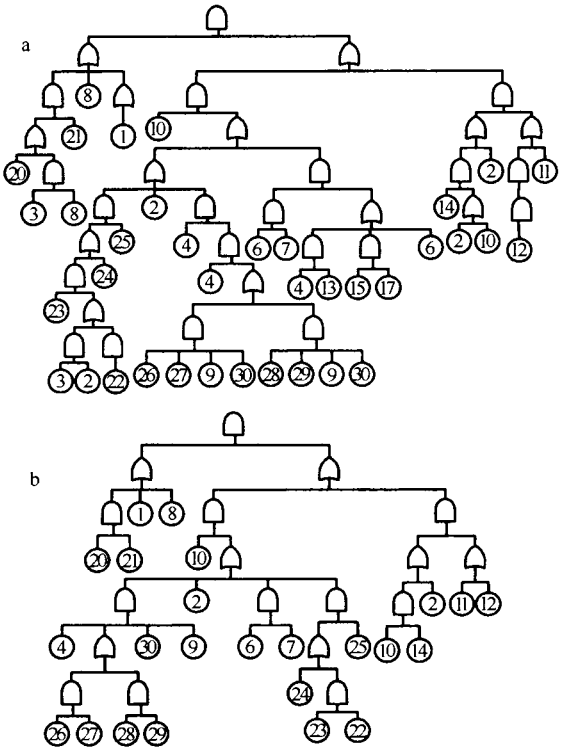

图2 方法应用示例

Fig. 2 Example of strategy

a——简化前的故障树；b——简化后的故障树

表1 方法实际应用效果

Table 1 Practical application effect of strategy

故障树名称	故障树规模(结点数)		故障树运算时间/s		逻辑简化耗费时间	
	简化前	简化后	运算前不进行简化	运算前进行简化(包括简化的时间)	简化的时间/ms	简化时间占运算时间的比例/%
高压安注系统直接注入阶段失效故障树	1 774	934	4.6	0.45	24	5.3
高压安注系统再循环冷端注入阶段失效故障树	1 219	536	超过2 h仍然没有结果	137.6	34	0.025
化学和容积控制系统维持上充失效故障树	842	580	超过2 h仍然没有结果	96.1	33	0.034

门下的子树运用局部优化规则,最终达到整个故障树的优化。另外,近几年来,针对具有动态随机性故障的容错系统,冗余(冷、热储备)可修复系统以及顺序相关性系统,研究者们又通过在传统故障树方法中加入一些新的符号(主要是表征动态行为的逻辑门)而引入了所谓的动态故障树方法[7]。对于动态故障树,仍然可以采用本文提出的贪心策略进行逻辑简化。

4　结论

本文给出的基于贪心算法的故障树逻辑简化方法,是一个基于树的广度遍历的非递归的线性方法,构造简单,简化效果也很明显。在故障树分析计算前,使用该方法对故障树进行预处理,可以有效降低故障树的结点数,从而大幅度地提高分析求解的速度,非常适合核电站Living PSA的要求。同时,该方法所采取的贪心策略也具有一定的通用性,可以运用在故障树分析的其他方面。

参考文献:

[1] 臧希年,闫　术,陈树明,等. 广东大亚湾核电站概率风险评估报告(附录)[R]. 北京:清华大学核能技术设计研究院,中国原子能科学研究院反应堆工程研究设计所,1997.

[2] Karen AR, John DA. A Fault Tree Analysis Strategy Using Binary Decision Diagrams[J]. Reliability Engineering and System Safety, 2002, 78(1):45～56.

[3] Hennings W, Kuznetsov N. FAMOCUTN and CUTQN: Programs for Fast Analysis of Large Fault Trees With Replicated and Negated Gates [J]. IEEE Transactions on Reliability, 1995, 44(3):368～376.

[4] Elliott MS. Computer-Assisted Fault-Tree Construction Using a Knowledge-Based Approach [J]. IEEE Transactions on Reliability, 1994, 44(1): 112～120.

[5] Cormen TH, Leiserson CE, Rivest RL, et al. Introduction to Algorithms[M]. Massachusetts: The MIT Press, 2001.326.

[6] Vince A. A Framework for the Greedy Algorithm[J]. Discrete Applied Mathematics, 2002, 121(1-3):247～260.

[7] Amari S, Dill G, Howald E. A New Approach to Solve Dynamic Fault Trees[J]. Reliablity and Maintainability Symposium, 2003, (27-30): 374～379.