一种故障树向 BDD 的转化方法

段 珊[1],张修如[2],刘树锟[1],王金娟[1]
DUAN Shan[1],ZHANG Xiu-ru[2],LIU Shu-kun[1],WANG Jin-juan[1]

1.湖南涉外经济学院 计算机学部,长沙 410013
2.中南大学 信息科学与工程学院,长沙 410083
1.College of Computer Science and Technology,Hunan International Economics University,Changsha 410013,China
2.School of Information Science and Engineering,Central South University,Changsha 410083,China

DUAN Shan,ZHANG Xiu-ru,LIU Shu-kun,et al.Transformation method of fault tree based on BDD.Computer Engineering and Applications,2009,45(21):51-54.

Abstract: Aiming at the key technology of fault tree analysis,sort and replacement.An effective method,LNPC is provided.This method uses the sort and the replacement strategy to complete the replacement of incident and the sort,and transform the fault tree to BDD.This method can enhance the chance of obtaining smallest size of the BDD,and reduce the storage space of BDD while the fault tree Boolean function doesn't need to write in advance.The analysis and experimental results show that this method is effective to defferent fault tree transform.

Key words: fault tree;Binary Decision Diagram(BDD);sort;replacement

摘 要:针对故障树分析的关键技术—排序和置换,提出一种基于 BDD 的快速有效的(LNPC)方法。该方法采用制定的排序和置换策略直接完成子事件的排序与门事件的置换,一次性完成故障树到 BDD 的转化和优化,增加了获取最小规模 BDD 的排序机会,同时降低了 BDD 的存储空间且不需要先写出故障树的布尔函数。算法分析与实验结果表明该方法对不同的故障树转化是有效的。

关键词:故障树;二元决策树;排序;置换
DOI:10.3778/j.issn.1002-8331.2009.21.013 **文章编号:**1002-8331(2009)21-0051-04 **文献标识码:**A **中图分类号:**TP301

故障树分析(Fault Tree Analysis,FTA)是目前国内外大型系统进行概率安全评价和可靠性分析时广泛使用的一种系统建模方法,其顶事件实质为一布尔函数。二元决策图(Binary Decision Diagram,BDD)则是布尔函数的一种高效表示[1]。1996 年 Andrews 提出了基于 BDD 的故障分析法[2],针对传统的利用最小割集来完成大型故障树分析所面临的组合爆炸问题指出了新方向。基于 BDD 的故障树分析方法是将故障树通过 Shannon 分解原理转化为 BDD,然后通过遍历 BDD 获得割集来完成对故障树定性和定量分析[3]。随着 BDD 理论研究的深入和应用领域的扩展,BDD 技术在故障树分析中显示了其计算效率高、计算精确、便于计算机编程实现的特点,近年来在系统可靠性分析中日益受到重视并取得不少应用。

现行 BDD 的故障树分析方法的缺陷:故障树的 BDD 的存储空间依赖底事件的排序,而较优底事件的排序是无法在故障树中直接取得;为了得到故障树的 BDD,必须先构造故障树的布尔函数,然后再构造相应的 BDD,无法一次性完成故障树到 BDD 的转化和优化,算法复杂度高。

针对故障树分析的关键技术—排序和置换,提出一种快速有效的(LNPC)方法。LNPC 方法采用制定的排序和置换策略直接完成子事件的排序与门事件的置换,一次性完成故障树到 BDD 的转化和优化。运用 LNPC 方法得到故障树的 BDD 是具有非固定变量排序的 FBDD,这种结构的 BDD 由于取消了统一变量序的约束,不仅增加了获取最小规模 BDD 的排序机会,同时保证了其具有与 OBDD 相同特点的性质;利用 LNPC 转化方法可以直接得到节点规模优化的 BDD 而不需要先写出故障树的布尔函数;边排序、边置换、边化简有效的减少了 BDD 的存储空间,使 BDD 一直以紧密的结构存储在计算机中,减少了存储空间的复杂度。算法分析与实验结果表明该方法对不同的故障树转化是有效的。

1 OBDD 与 FBDD

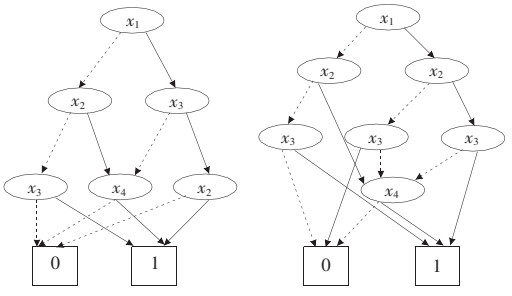
二元决策图是 boolean 函数的图形表示,可以直观地反映函数的逻辑结构。1959 年 Lee 提出来用 BDD 来表示二元函数的数据结构,随后在这种基本模型的基础上,发展了具有不同特征的 BDD 成员。OBDD(ordered bdd)和 FBDD(free bdd)就在其中。给 BDD 加上相应的约束条件,就能有效地处理布尔函数,OBDD 就是一种具有固定变量排序的 BDD,由于 OBDD 是 BOOLEAN 函数的规范表示,有效地完成相应操作,广泛成功

作者简介:段珊(1969-),女,高级工程师,研究方向:算法理论,体系结构;张修如(1957-),男,副教授,硕士导师,研究方向:软件工程,算法;刘树锟(1979-),男,硕士,主要研究方向:软件工程、数据库技术;王金娟(1981-),女,讲师,研究方向:数据挖掘,算法理论。

收稿日期:2008-10-10 **修回日期:**2009-01-04

地运用到数字系统各领域。自由排序二元决策图(Free Binary Decision Diagrams,FBDD)[4] 是针对受约束的 OBDD 而言,实质上是一种非固定排序的 BDD,但在任一分支上每个变量只出现一次。OBDD 并非一种完善的图结构,它的缺陷主要是 BDD 对排序的敏感;特别是 BDD 采用统一的固定排序,这种约束在一些情况下,约束了 BDD 最优结构的产生。比如整个 BDD 表示的函数是由多个不相关联的系统组成,不同的子系统采用不同的排序时,得到的规模更小。FBDD 虽然在全局的 BDD 中并不存在一个固定的排序,但在一条路径上,它的每个变量只出现一次;它采用了类似 OBDD 化简规则,也保留了类似 OBDD 的许多属性:精简性、唯一性等。所以在不相关的多子系统中,采用它来获得最优化的结构的几率要大,并同时保留了所需要的属性。

函数 $f=\bar{x}_1\bar{x}_2x_3+\bar{x}_1x_2x_4+x_1\bar{x}_3x_4+x_1x_2x_3$,图 1(a)对应函数 F 的 FBDD,图 1(b)对应函数 F 的 OBDD。



(a)F的 OBDD (b)F 的 FBDD

图 1 OBDD 与 FBDD

2 故障树向 BDD 的转化

利用 BDD 来完成故障树分析,关键处实现故障树向 BDD 的转化,这一个过程有两个关键点,一是如何根据故障树的特点,实现用底事件置换门事件,另一是如何确定底事件的排序。在以往的方法中,往往分成两步来完成。

Rauzy 提出来的 ITE 构造法[2]:这种技术基于递归思想使用 if_then_else 结构,如果 x 成立则 F1 成立;否则 F2 成立,数学表达式:

$$ite(x,f_1,f_2)=x\cdot f_1+\bar{x}\cdot f_2 \quad (1)$$

从故障树的最底层门事件开始用底事件替换门事件,逐层向上,每置换一步同时按 ITE 结构对置换进行编码。如此类推将所有的门事件均用底事件替换,便可得到顶事件的 BDD。采用这种思想得到 OBDD 首先要先确立底事件的顺序。

由于底事件的排序对应 BDD 中变量排序,因此它严重影响 BDD 的规模。底事件的排序与其在故障树的位置密切相关,由此联系到故障树中底事件的结构重要度。然而底事件的结构重要度必须先求故障树,这和利用 BDD 进行故障树分析相矛盾[6]。

为了高效快速地得到故障树的 BDD,在对已采用的排序和置换方法进行分析研究的基础上,提出了逻辑相邻优先组合(Logic Neighbor Priority Connect,LNPC)方法。该方法按制定规则对故障树从顶事件开始向下遍历,遍历的顺序确定底事件的顺序,并在遍历的过程中完成门事件的编码置换,直到整个故障树转化为只含底事件的 BDD。这种处理手段可以一次性地完成故障树到 BDD 的转化与优化。执行它得到的 BDD 为 FBDD 结构,它不仅保留 OBDD 的属性,而且增加了获取最小 BDD 结构的灵活性。

2.1 LNPC 方法的概念

为了便于描述 LNPC 方法,对出现在 LNPC 方法中用到的基本概念进行说明(见图 2 故障树图解):

底事件:无法再分解的元事件 A、B、C、D、E;

门事件:可以用底事件或其他门事件来组合置换的事件 G1、G2、G3;

顶事件:故障树中的最高处的门事件 G1;

子事件:用来替代门事件的所有成分都称为子事件,包括该门下的所有底事件和门事件。G1 的子事件为 G2 A,G3;G2 的子事件为 B,C;G3 的子事件为 D,E;G1 的子事件 G2、A、G3;G2 的子事件为 B、C;

兄弟底事件:同一门下的底事件,也称兄弟底事件 B 和 C、D 和 E;

兄弟门事件:同一门下的门事件;G2 和 G3 为 G1 门下的兄弟门事件。

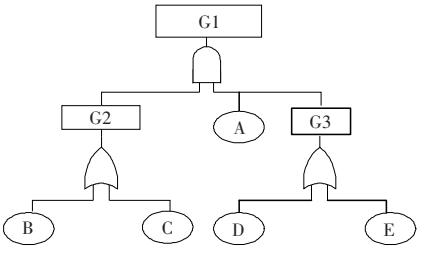


图 2 故障树图解

2.2 LNPC 排序策略

LNPC 方法的基本思想是高效得到故障树最少节点规模的 BDD,而影响 BDD 规模的因素就是底事件的排序。

为了得到最优化的底事件排序,提出以下几种排序策略:

(1)门事件、底事件均参与排序。故障树中,有两种事件:底事件(底事件)和门事件,除顶门事件外,在排序过程中所有子事件同等对待。

(2)采用非全局的固定排序。为了得到规范的 BDD 范式,Bryant 给 BDD 加上了固定的变量序约束。固定的变量序可使 BDD 有唯一的表示,便于 BDD 的操作,但它可能会导致最小规模 BDD 的丢失。选择一个故障树中的底事件,它的两个分支对应不同系统的状态,这两个状态可能是完全不同的。这种情况下采用统一的底事件顺序来约束不同的系统状态不是必须的,底事件的固定顺序可能限制了 BDD 最小规模的产生。非全局的固定排序允许不同路径上有不同的排序,增加最小规模 BDD 产生的几率。

(3)逻辑相邻关系的优先排序。目前运用于故障树排序的启发性方式可分为两大类:结构重要度和概率重要度[5]。结构重要度主要依赖底事件在故障树中的物理位置,所以这样方法得到的 BDD 取决于向下搜索的方法;概率重要度考虑底事件的为关键事件的概率,它虽然也考虑结构的限制,但它忽略了底事件之间的关系;新排序的思想不仅考虑门之间的拓扑结构,也充分考虑了相邻底事件及门事件之间的逻辑关系。所以对于逻辑相邻的事件在排序时应尽量相邻排序,因为从分析可以看到:当故障树中底事件或门事件失效时最先受其影响的是其逻

辑上的相邻因子(门事件和底事件)。

(4)先遍历先排序,边排序边置换。按制定的规则遍历故障树,依据遍历的次序对故障树中的子事件(包含门事件和底事件)进行排序,边排序边编码置换为对应的 BDD。

2.3　LNPC 转换置换规则

转换置换就是将故障树中的子事件转换为相应的 BDD 的节点,门事件对应的 BDD 节点称为门节点,底事件对应 BDD 的底事件节点;并对 BDD 中存在的门节点所对应的故障树中门事件继续按规则遍历排序、置换,直到 BDD 节点均为底事件节点。故障树转化为 BDD 的置换过程中,遵循下面规则进行。

(1)规范后的故障树中含有两种事件:门事件和底事件,门事件和底事件一样均为二态,可以将门事件作为等价底事件来对待,即门事件和底事件一起参与置换。

(2)故障树经过化简规范后门事件就两种"与"门和"或"门,"与"门"或"门事件的置换按下面方法[7]:

与门:"与"门下各输入子事件(包含底事件和门事件)沿 1 支路连接;

或门:"或"门下各输入子事件(包含底事件和门事件)沿 0 支路连接

(3)故障树中同一门下的子事件在排序后完成门事件的置换。在门事件的置换中,若遇到下列情况需进行化简处理。

第一种情况:一条路径上出现重复事件。事件在路径上的第一次出现就决定了第二次出现时的状态,因此就用第二次出现时节点输出取代第一次状态时下面的节点。

第二种情况:冗余节点的出现。当 BDD 中某节点的 1 边和 0 边指向的节点结构一致时,则此节点为冗余节点,删除它并将指向此节点的边直接指向此节点的后续节点。

第三种情况:同构的节点。当多个节点同构时,保留一个节点,并将指向同构节点的边引向保留的节点,其他节点删除。

2.4　LNPC 算法

2.4.1　故障树冗余化简

很多情况下原始的故障树中可能存在一些不相干的因素,即冗余。这些因素可以是底事件,也可以是门事件(子树)。为此必须要将这些不相干的因素在保证逻辑不变的情况下进行化简。化简的依据是吸收律:

$$A+(A*B)=A \tag{2}$$

$$A*(A+B)=A \tag{3}$$

A、B 可以是门事件也可是底(子)事件。遍历故障树若发现在兄弟节点或兄弟树中存在相同的事件(因子)可根据上述的规则进行化简重复或删除多余因子。

2.4.2　LNPC 转化规则

LNPC 实现故障树向 BDD 的转化实质是一个从顶向下按下列实现规则完成子事件的排序置换过程:

规则 1(底事件排序置换规则)　底事件位置越高,优先级高;底事件重复出现次数越多,优先级越高,次数相同则选择第二次出现的位置高低确定优先级;左边位置的优先级高。底事件按其父门的类别进行置换化简。

规则 2(门事件排序置换规则)　位置越高优先级高;同等优先位置则选择含有最近已排序的优先级最高底事件的门事件;底事件少的门事件优先级高;深度小的门事件优先;左边门事件优先。门事件按父门的类别进行置换化简。

规则 3(混合子事件的排序置换规则)　若门事件中同时存在底事件和子门事件,按下列情况排序置换:

(1)若本门中所有底事件均未排序过(相当于顶门事件)则先按底事件排序规则对底事件排序,再按门事件排序规则对门事件排序。该门子事件排序完毕后则根据门的类别对门事件进行置换法化简。

(2)若门事件中的底事件中有最近已经排序的,则选择含有优先级最高的已排序事件的子事件优先排序;若子事件为该门下的底事件,该底事件和其兄弟底事件先排序,再对兄弟门事件排序;若子事件为门事件,对余下子事件排序时,将刚排序的门事件下的子事件视同为最近已排序事件按规则 3(2)处理余下子事件。该门子事件排序完毕后则根据父门的类别对所有子事件按组合方法组合化简。

(3)若本门事件中的底事件均已排序则按排序规则完成组合化简。

2.4.3　LNPC 方法的执行步骤

步骤 1　从顶事件开始向下遍历依据 LNPC 的实现规则 3(1),得到最初 BDD。

步骤 2　对已得到的 BDD 中门事件依顺序按实现规则 1、2、3 进行排序置换,直到 BDD 中不再出现门事件为止。

利用 LNPC 方法实现故障树的 BDD 转化过程中可以发现:在没有完全转化为对应的 BDD 之前 BDD 的节点实际上包含底事件和门事件;组合过程中一条路径上同一子事件可能出现多次,化简后可以保证一条路径出现一次,但在整个 BDD 中没有一个固定的子事件排序。

2.4.4　LNPC 算法描述

LNPC 方法的核心有两个。一个是子事件的排序,另一个是故障树向 BDD 的转化置换。这两者均在遍历故障树的过程中实现。具体实施时依据实现规则 1,规则 2,规则 3 来完成遍历顺序的选择和门事件的置换(或者子事件的组合),算法流程图描述如图 3。

3　算法分析与实验

将上面的算法用 C 语言编程,分别用 ITE 算法和 LNPC 算法在几个故障树上运行,得到时间对比和空间对比图(图 4)。

LNPC 方法总体上采用从上至下逐级分解,从底事件和门事件看 BDD 中的节点,然后逐级对节点中门事件完成分解,直到整个 BDD 完全由底事件组合替代。

将 LNPC 的方法与其他故障树的构造法进行比较分析,具有以下特点:

(1)LNPC 算法在对故障树进行遍历的过程中,通过排序置换一次性得到优化的 BDD,因此不需要求取顶事件的逻辑函数;其他的方法(如 ITE 方法)大多采用两步来完成 BDD 的优化,先运用某种策略得到较合适的排序,然后求取顶事件的逻辑函数构造 BDD。

(2)LNPC 算法按制定的策略遍历故障树的过程是一个隐式确定底事件排序的过程。遍历的顺序非单纯依赖位置关系,

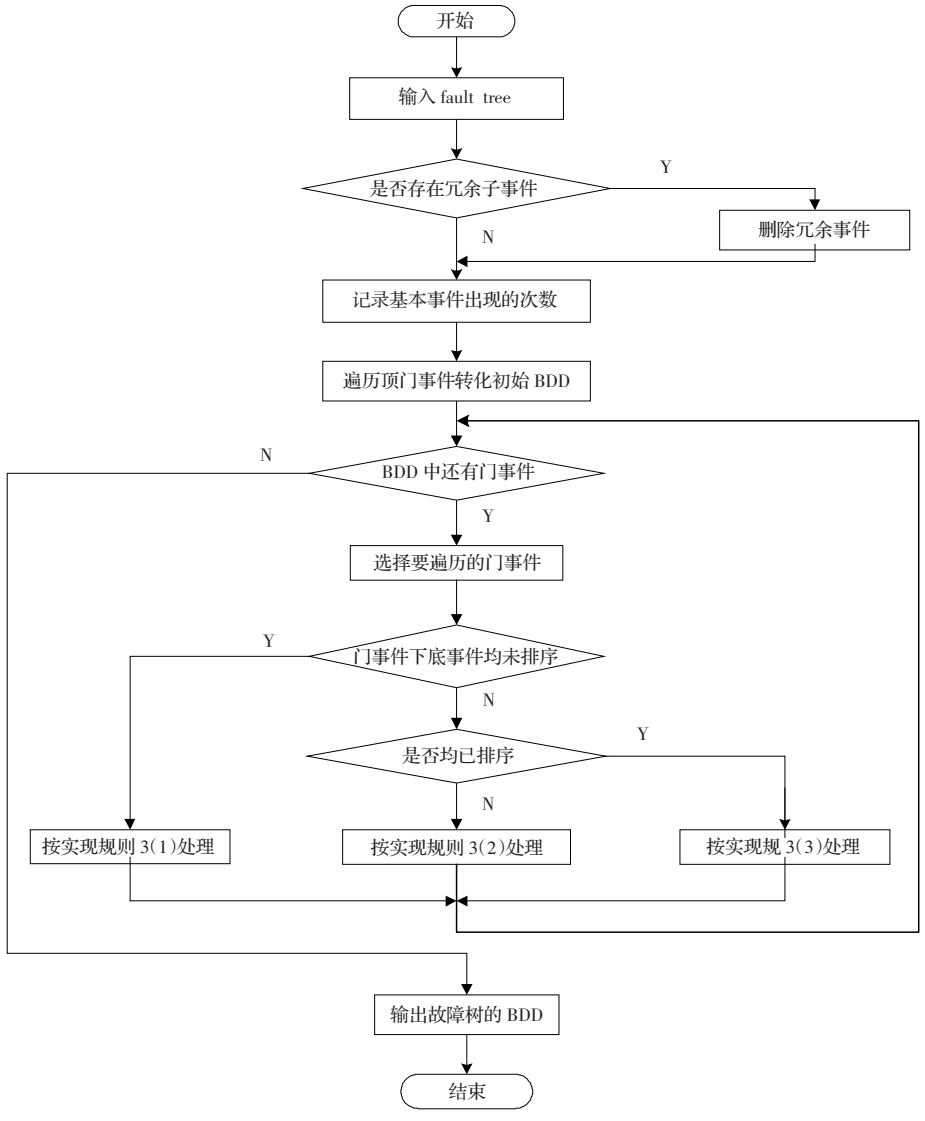
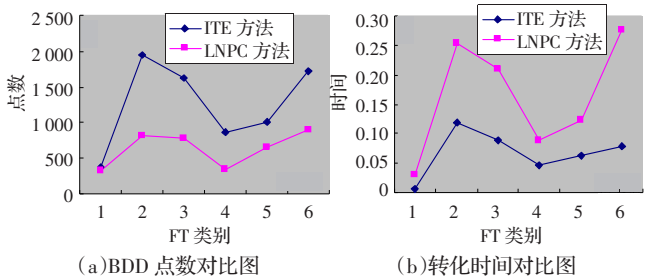


图 3 算法流程图



(a)BDD 点数对比图 (b)转化时间对比图

图 4 空间对比和时间对比图

同时也考虑逻辑上近邻优先;对同一门下的子事件排序时依赖的是其事件的重复性和逻辑近邻关系。而其他方法不管是结构重要度方法还是概率重要度方法都只强调一个方面。所以相对其他算法 LNPC 算法排序更具备完备性。

(3)LNPC 算法边遍历边置换化简,保证每次得到的结构最为优化,所以在整个构造过程中不会产生很大的临时开销,减少了存储空间。

LNPC 算法的空间复杂度无论是理论上分析还是实例验证,都是简洁有效的,并可得到优化的故障树的 BDD 结构;在 6 个较简单的故障树上分别运行 LNPC 算法和 ITE 算法的实验结果表明,LNPC 算法是有效的。当然,该算法还需进一步完善。

参考文献:

[1] Akers B.Binary decision diagrams[J].IEEE Transactions on computers,1978,27(6):509-516.

[2] Sinnamon R M,Andrews J D.New approaches to evaluating fault trees[J].Reliability Engineering and System Safety,1997,58(3):89-96.

[3] Sinnamon R M,Andrews J D.Improved efficiency in qualitative fault tree analysis[C]//Quanlity and Reliability Engineering International,1997:293-298.

[4] Gergov J,Meinel C.Efficient analysis and manipulation of OBDDS can be extended to FBDDS[J].IEEE Transactions on Computers,1994,43(10):1197-1209.

[5] 金星,洪延姬.系统可靠性与可用性分析方法[M].北京:国防工业出版社,2007.

[6] 郭伟伟,马捷中,翟正军.故障树分析中底事件排序问题的研究[J].计算机工程与设计,2007,28(15):3557-3559.

[7] Remenyte R,Andrews J D.A simple component connection approach for fault tree conversion to binary decision diagram[C]//Proceeding of the First International Conference on Availability,Reliability and Security,2006:449-457.