基于粗糙集的因果图方法简化研究

张永强,刘彦瑞
(河北工程学院,河北 邯郸 056038)

摘 要:因果图方法是一种有效的软件测试方法,它适合于描述对于多种条件的组合、相应产生多个动作形式的测试用例设计,因果图最终被转换为判定表。在判定表中,测试用例的数目随输入数据数目的增加而线性地增加,当输入数据数目较大时,最终生成的判定表的规模会很大,相应的生成的测试用例会很多,并且,程序的规格说明中给出的有些条件及条件值并不是必需的,因此,文中采用粗糙集理论对因果图方法进行简化,并用一个实例说明了该简化方法的合理性和实用性。经简化,由因果图生成的判定表的条件和条件值都不再有冗余,所生成的测试用例数目大大减少,测试变得更加简洁有效。

关键词:软件测试;因果图;粗糙集;判定表
中图分类号:TP311.56 **文献标识码**:A **文章编号**:1673-629X(2006)07-0042-03

Simplification Research on Graph of Cause and Effect Method Based on Rough Set

ZHANG Yong-qiang, LIU Yan-rui
(Hebei Engineering University, Handan 056038, China)

Abstract: Graph of cause and effect is an effective software testing method, which is suitable to design the testing cases that the combination of multiple conditions generated multiple actions. The graph of cause and effect is ultimately transferred into decision table. In decision table, with the increasing number of the input data, the testing cases grow linearly. When the number of input data is lager, the corresponding produced decision table must has great scale and the testing cases will be excessive. Moreover, some conditions and their values given from specification introduction of the project is unnecessary. Thus, in this paper, the graph of cause and effect is simplified using rough set. And an instance is used to illustrate the reasonableness and applicability of the simplified method. After simplified, conditions and their values have no more redundancies in the decision table derived from the graph of cause and effect. That lessens the number of the test cases far and away. So the testing is more brief and effective.

Key words: software testing; graph of cause and effect; rough set; decision table

0 引 言

软件测试作为软件质量保证的重要手段,已经成为每个软件公司生产流程必不可少的组成部分。加强软件测试的力度和提高软件测试的效率对保证软件产品最终质量有很大帮助。软件测试的方法有很多,但不外乎两类:黑盒测试和白盒测试。

因果图方法是一种黑盒测试方法,它是一种适合于描述对于多种条件的组合、相应产生多个动作形式的测试用例设计方法[1]。它最终被转换成判定表,判定表适合于检查程序输入条件的各种组合情况。它生成的测试用例包括了所有输入数据的取 TRUE 与取 FALSE 的情况,测试用例的数目随输入数据数目的增加而线性地增加。所以,当输入数据数目较大时,最终生成的判定表的规模就会很大,相应的生成的测试用例就会很多,并且,程序的规格说明中给出的有些条件及条件值并不是必需的,这样就有必要用一种方法对其进行简化,消除它的冗余条件及条件值[2]。

收稿日期:2005-09-16
基金项目:国家自然科学基金资助项目(60573088)
作者简介:张永强(1966-),男,河北南宫人,教授,研究方向为软件可靠性工程。

1 粗糙集理论方法

粗糙集方法(Rough Set)[3]是一种处理不确定、不完全、不一致数据的数学方法,是数据挖掘的方法和工具[4]。粗糙集方法基于一个机构(或一组机构)关于一些现实的大量数据信息,从中发现、推理知识和分辨系统的某些特点、过程、对象等[5]。

1.1 粗糙集的基本概念

设 U:对象集,也称为论域;R:属性集,也称为知识或关系;U/R:等价类族。

1.1.1 粗集

若 U 上的关系 R 同时具有自反、对称、传递性,则 R 是 U 上的等价关系。U 在 R 下被划分成若干等价类,所有等价类构成等价类族,记为 U/R,每一个等价类里的元素

在 R 下不可区分,任意两个等价类之间交集为 $\emptyset$,所有等价类的并集为整个论域。$[x]_R$ 表示等价关系 R 对 U 划分成的等价类中包含 x 的那些等价类。

U 中子集 X 的下逼近 $\underline{R}(X) = \{X \mid [x]_R \subset X, x \in U\}$,所以 $\underline{R}(X)$ 是由 U 中完全包含在 X 中的等价类构成的集合。

集合 X 的上逼近 $\overline{R}(X) = \{X \mid [x]_R \cap X \neq \emptyset\}$,所以 $\overline{R}(X)$ 是由与 X 相交非空的等价类构成的集合。

当 $\overline{R}(X) - \underline{R}(X) \neq \emptyset$ 时,称 X 为 R 粗集。

1.1.2　约简与核

给定决策系统 $T = (U, A = C \cup D)$,其中,U 为论域,C 为条件属性集,D 为决策属性集,C 中所有等价关系的交集记作 IND(C),D 中所有关系的交集记作 IND(D),则 IND(C) 把论域 U 划分的等价类关系如下:

$U/\mathrm{IND}(C) = \{X_1, X_2, \cdots, X_m\}$　$U/\mathrm{IND}(D) = \{Y_1, Y_2, \cdots, Y_n\}$

$C(Y_i) = \mathrm{POS}_C(Y_i) = \{x \mid [X]_C \subseteq Y_i, x \in U\}$

定义 1　令 $\mathrm{POS}_C(D) = \cup C(Y_i)$,表示包含在决策等价类中的元素。

定义 2　给定 $T = (U, A = C \cup D)$,若 $R \subseteq C$,且 $\mathrm{POS}_{C-R}(D) = \mathrm{POS}_C(D)$,则称 R 是 C 上 D 可约去的,否则称 R 为 C 上 D 不可约去的。

定义 3　若等价关系 $P \in C$,且 P 上的每个等价关系 V 都是 P 上 D 不可约去的,则称 P 关于 D 是独立的。

定义 4　等价关系 $P \in C$ 是 C 的 D 约简,当且仅当 P 是 C 的 D 独立子族,且 $\mathrm{POS}_P(D) = \mathrm{POS}_C(D)$。

定义 5　所有 C 中 D 不可约去的等价关系的集合被称为 C 的 D 核,记作 CORD_D,若令 RED(A) 表示 A 的所有约简构成的集合,则

$\mathrm{CORE}_D(C) = \cap \mathrm{RED}_D(P)$。

1.2　粗糙集理论方法对决策表数据的约简步骤

(1)一致性检验。

在决策表中,检验当条件属性值相同时,决策属性值是否相同,若相同,说明该决策表是一致的(协调的)。

(2)属性约简,先求出核属性。

前提:决策表是一致的。

做法:去掉一列条件属性,若该决策表出现不一致则该条件属性是核属性,不能去掉,否则可以去掉,然后再合并相同的规则,通过往核中每次增加 1 个,2 个……非核外属性的方法,确定决策表的约简集。

(3)属性值约简(值约简,也称为决策规则约简),得出核值表和值约简表。

第一步:求出核值表。依次在约简集的属性表的每一条规则中分别去掉每一个条件属性值,若决策表出现不一致,则该条件属性值不能去掉,否则可以去掉,然后再合并相同的规则。

第二步:根据核值表和属性约简表,求出该决策表的值约简表。

(4) 在已求得的约简中任取一种约简作为最小算法。

2　粗集在因果图法中的应用

例　机器人控制系统(智能体自适应系统)。

机器人抓棒的过程中,所处环境有如下 6 种状态:

(1)机器人所处位置是否在中心点,若是则 a 为 1,否则为 0;

(2)机器人是否正前方对棒,若是则 b 为 1,否则为 0;

(3)机器人是否在棒的中心线上,若是则 c 为 1,否则为 0;

(4)机器人正前方面对中心线,若是则 d 为 1,否则为 0;

(5)机器人是否已抓棒,若是则 e 为 1,否则为 0;

(6)行为:A 旋转;B 前移;C 抓棒;D 停止。

2.1　根据描述明确原因和结果

原因:a:机器人所处位置在中心点;

b:机器人正前方对棒;

c:机器人在棒的中心线上;

d:机器人正前方面对中心线;

e:机器人已抓棒。

结果 f:1 旋转;2 前移;3 抓棒;4 停止。

2.2　画出因果图

分析软件规格说明描述中的语义,找出原因与结果之间、原因与原因之间对应的关系,根据这些关系,画出因果图。如图 1 所示。

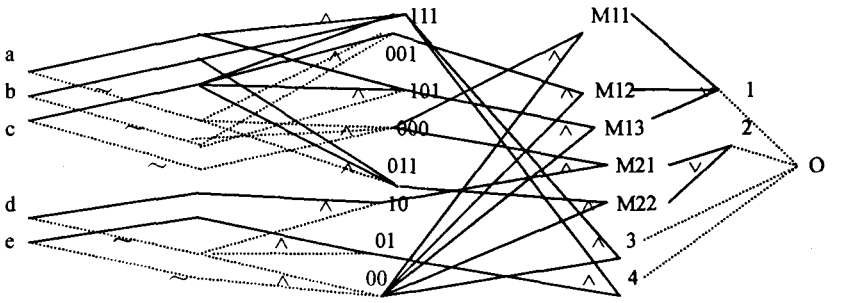

图 1　机器人控制系统的因果图

2.3　根据因果图写出判定表

根据所画出的因果图得到判定表[1](见表 1)。

表 1　机器人控制系统的判定表

		1	2	3	4	5	6	7
原因	a	0	0	1	0	0	1	1
	b	0	0	0	0	1	1	1
	c	0	1	1	0	1	1	1
	d	0	0	0	1	0	0	0
	e	0	0	0	0	0	0	1
结果	f	1	1	1	2	2	3	4

2.4　根据判定表构造决策表

根据表 1 得到决策表(见表 2)。

表 2 机器人控制系统的粗糙集决策表

U \ A	a	b	c	d	e	f
1	0	0	0	0	0	1
2	0	0	1	0	0	1
3	1	0	1	0	0	1
4	0	0	0	1	0	2
5	0	1	1	0	0	2
6	1	1	1	0	0	3
7	1	1	1	0	1	4

2.5 对决策表进行简化

2.5.1 一致性检验

经检验,在该决策表中,所有规则的条件属性值各不相同,所以该决策表是一致的。

2.5.2 属性约简

依次去掉表中的一个条件属性所在列,若决策表出现不一致,则该属性是核属性,不能删去,否则,该条件属性可以删去。

属性约简并合并相同规则后,得表 3。

表 3 属性约简表

U \ A	a	b	d	e	f
1	0	0	0	0	1
3	1	0	0	0	1
4	0	0	1	0	2
5	0	1	0	0	2
6	1	1	0	0	3
7	1	1	0	1	4

由于表中只有一种非核属性,其余全是核属性,所以这些属性构成该决策表的唯一约简。

2.5.3 属性值约简(决策规则约简)

属性值约简就是去掉规则中的某些冗余属性值,且不影响决策,并将去掉冗余属性值后某些规则变得相同时,就进行规则合并以减少规则数。

①确定核值表。

依次在核属性表的每一条规则中分别去掉每一个条件属性值,若决策表出现不一致,则该条件属性值不可去掉,否则可以去掉,然后再合并相同的规则,得表 4。

表 4 核值表

U \ A	a	b	d	e	f
1	–	0	0	–	1
3	–	0	–	–	1
4	–	–	1	–	2
5	0	1	–	–	2
6	1	1	–	0	3
7	–	–	–	1	4

②根据核值表求约简规则[4]。

规则 1:[1]f1={1,3}

[1]a0={1,4,5} [1]b0={1,3,4}

[1]d0={1,3,5,6,7} [1]e0={1,3,4,5,6}

所以得 b0∧d0→f1。

同理得 规则 3:(b0∧d0)∨(a0∧b0)→f1

规则 4:d1→f2

规则 5:a0∧b1→f2

规则 6:a1∧b1∧e0→f3

规则 7:c1→f4

所以得值约简表(见表 5)。

表 5 属性值约简表

U \ A	a	b	d	e	f
1	–	0	0	–	1
3	1	0	–	–	1
3′	–	0	0	–	1
4	–	–	1	–	2
5	0	1	–	–	2
6	1	1	–	0	3
7	–	–	–	1	4

注:"–"表示可去掉的属性值,指去掉任意一个"–"后,决策规则不受影响。

2.5.4 取出一种最小规则

在值约简表中任意取出一种最小规则,得表 6。

表 6 约简后的一种最小规则

U \ A	a	b	d	e	f
1	–	0	0	–	1
4	–	–	1	–	2
6	1	1	–	0	3
7	–	–	–	1	4

2.5.5 将简化后的决策表转换成判定表,得表 7。

表 7 约简后的判定表

		1	4	6	7
原因	a	–	–	1	–
	b	0	–	1	–
	d	0	1	–	–
	e	–	–	0	1
结果	f	1	2	3	4

2.5.6 由判定表写出测试用例

输入	输出
(1)机器人正前方不对棒且不在中心线上	旋转
(2)机器人正前方对中心线	前移
(3)机器人在棒的中心线上且正前方对棒,尚未抓棒	抓棒
(4)机器人已抓棒	停止

可以看到,简化后的测试用例可以对该规格说明进行有效的测试,并且测试用例的数目明显减少了,说明了此方法的可行性。

3 结束语

文中运用粗糙集的理论对因果图软件测试方法进行了简化,在保证测试质量的前提下,减少了测试用例的数目,提高了软件测试的效率。

(下转第 48 页)

s3cExcVecSet(void)将其与异常处理函数的关联起来。

```
void s3cExcVecSet(void)
{
int i;
i = (int)&excEnterUndef;
*((volatile int*)(S3C_EXC_BASE + 0x0)) = i;
i = (int)&excEnterSwi;
*((volatile int*)(S3C_EXC_BASE + 0x4)) = i;
i = (int)&excEnterPrefetchAbort;
*((volatile int*)(S3C_EXC_BASE + 0x8)) = i;
i = (int)&excEnterDataAbort;
*((volatile int*)(S3C_EXC_BASE + 0xc)) = i;
i = (int)&intEnt;
*((volatile int*)(S3C_EXC_BASE + 0x14)) = i;
return;
}
```

同时,将函数 s3cExcVecSet()加到 sysHwInit()即可完成向量表的设置。

(7) 时器驱动。

在 sngks32cTimer. h 中,进行寄存器定义。在 sngks32cTimer. c 中,修改与寄存器、初始化、连接、使能和禁止函数有关。辅助时钟暂时没有使用,相关函数中的寄存器操作语句都注释掉。

(8) 串口驱动。

S3C44B0X 和 S3C4510 串口寄存器的地址和使用有很大差别。参考 CPU 手册,在 sngks32cSio. h 中,去掉无用寄存器定义,添加新寄存器定义。

串口的波特率设置值也不相同,S3C44B0X 波特率与系统主时钟有如下计算公式:

UBRDIVn = ((int)(fMCLK/16./baud + 0.5) - 1)

根据该公式,重新计算设置值,替换原来的定义。

串口驱动的实现在 sngks32cSio. c 中,需要根据寄存器定义变化作相应修改:

① 数据结构。

修改串口数据结构 devParas[],设置 UART 的接收发送的中断号、向量号、寄存器基地址等。

② 初始化。

串口初始化时用 devParas[]数据填充底层驱动的 S3C44B_CHAN 结构,每个 UART 都有各自的 S3C44B_CHAN 结构,然后调用底层初始化函数 s3c44bDevInit 进行实际的 UART 初始化。对 UART 的初始化分为 2 个阶段,第 1 阶段设置回调函数、初始化硬件、设置波特率、设置 UART 为轮询模式;第 2 阶段设置 UART 为中断模式,但不使能 UART 中断。

③ 发送(中断和轮询)。

中断方式下,由 s3c44bTxStartup 启动发送,往 UTXH 写入数据,并使能 UART 中断;发送完成后由回调函数 s3c44bIntTx 检查是否还有数据要发送,是则发送,否则关闭中断。轮询方式下,不断检查 UTRSTAT,若发送就绪往 UTXH 写入数据返回 OK,否则返回 EAGAIN。

④ 接收(中断和轮询)。

中断方式下,当接收到数据时,由函数 s3c44bIntRx 调用回调函数 pChan -> putRcvChar 接收数据。轮询方式下,不断检查 UTRSTAT,若接收就绪则返回接收到的数据。

至此,S3C44B0X 的 BSP 基本模块已开发完毕。使用命令或者在 Tornado 集成环境下用 Build Boot ROM 来创建各种类型的 BootROM[5]。

一旦 VxWorks 内核被成功激活,就可以进入 BSP 的调试和完善阶段了。

在此阶段,需要对 BSP 进行测试,验证系统是否正确、稳定运行;同时,还可添加一些驱动模块,如 END 驱动模块、TFFS 驱动模块和 LCD 驱动模块。通过这些步骤,一个比较完整的 S3C44B0X 的 BSP 包开发就完成了。

3 结束语

BSP 包是目标机系统上电后首先执行的代码,它的开发正确性将直接影响到 VxWorks 系统运行的稳定性和项目的后续开发工作。BSP 设计与开发涉及到硬件和软件两方面的知识,故对所设计的系统结构要有足够的认识。

参考文献:

[1] 周启平. VxWorks 下设备驱动程序及 BSP 开发指南[M]. 北京:中国电力出版社,2004.

[2] 王田苗. 嵌入式系统设计与实例开发[M]. 北京:清华大学出版社,2003.

[3] VxWorks Programmer's Guide(Version5.4 Edition 1)[Z]. Atlantic:Wind river systems,1999.

[4] Wind river systems. Tornado BSP Training Workshop(version1.0.2)[Z]. Atlantic:Wind river systems,1998.

[5] Wind river systems. Tornado User's Guide(Windows Version2.2)[Z]. Atlantic:Wind river systems,2002.

(上接第 44 页)

参考文献:

[1] 古 乐,史九林. 软件测试技术概论[M]. 北京:清华大学出版社,2004.70-76.

[2] 郑人杰. 计算机软件测试技术[M]. 北京:清华大学出版社,1994.72-85.

[3] Pawlakz, Grzymala - Busse J, Slowinski R. et al. Roughs sets [J]. Communications of the ACM,1995,38(11):89-95.

[4] 曾黄麟. 粗集理论及其应用[M]. 重庆:重庆大学出版社,1998.117-127.

[5] 庞彦军,栗文国,刘开第. 粗集理论与未确知系统理论的互补性研究[J]. 河北工程学院学报,2003,20(3):5-8.