文章编号:1001 - 9081(2007)04 - 0884 - 04

基于新的条件熵的决策树规则提取方法

孙 林,徐久成,马媛媛
(河南师范大学 计算机与信息技术学院,河南 新乡 453007)
(slinok@126.com)

摘 要:分析了知识约简过程中现有信息熵反映决策表"决策能力"的局限性,定义了一种新的条件熵,以弥补现有信息熵的不足;然后对传统启发式方法中选择属性的标准进行改进,由此给出了新的属性重要性定义;以新的属性重要性为启发式信息设计决策树规则提取方法。该方法的优点在于构造决策树及提取决策规则前不进行属性约简,计算直观,时间复杂度较低。应用实例分析的结果表明,该方法能提取更为简洁有效的决策规则。

关键词:粗糙集;数据挖掘;条件熵;规则;决策树
中图分类号: TP311.13 **文献标识码**:A

Rules extraction method of decision tree based on new conditional entropy

SUN Lin, XU Jiu-cheng, MA Yuan-yuan
(*College of Computer and Information Technology, Henan Normal University, Xinxiang Henan* 453007, *China*)

Abstract: The disadvantages of the current information entropy for estimating decision ability were analyzed deeply. To eliminate the limitations, a new conditional entropy was defined. The attribute selection metric of traditional heuristic algorithm was modified, so the new improved significance of an attribute was proposed. Finally, a heuristic algorithm for rules extraction of decision tree was designed. This reduction method does not need attribute reduction before extracting decision rules, and its computation is direct and efficient, and its time complexity is less than the others. The experiment and comparison show that the algorithm provides more precise and simple decision rules.

Key words: rough set; data mining; conditional entropy; rule; decision tree

0 引言

粗糙集理论是一种可以分析模糊和不确定知识的数学工具,已广泛应用于数据挖掘的各个阶段[1]。目前,许多学者通过不同的方法从不同的角度对决策规则获取(值约简)做了深入的研究。文献[2]提出一种基于分类一致性的规则获取算法。文献[3]提出的 RITIO 算法采用熵测度来度量决策表中条件属性和决策属性的相关性,通过逐步删除最不相关属性,从相容的对象中提取规则,该方法抗噪声能力强,但规则前件较复杂,有冗余且可能有些规则不一致[2]。文献[4]提出一种值约简算法,但该算法获取的规则存在属性冗余和规则冗余。

为了解决以上问题,本文采用目前 KDD 中最有效的决策树[5~6]分类规则学习方法,但构造最优决策树已被证明是 NP-Hard 问题[7],所以在属性的选择中,采用更优的启发式函数来构造决策树,并提取决策规则的方法具有潜在的实用意义。在决策表中,首先深入分析了现有信息熵反映决策表"决策能力"的局限性,给出一种新的条件熵定义,使约简结果能更客观地反映决策表"决策能力"的实质;然后从优化决策树方面考虑,对传统启发式方法中选择属性的标准进行改进,在此基础上定义新的属性重要性;并以新的属性重要性为启发式信息构造决策树;最后设计规则约简过程来简化决策规则。实例分析表明,该方法是有效的。

1 主要概念

定义 1[1] 五元组 $S = (U, C, D, V, f)$ 是一个决策表,其中 U 为论域;C 为条件属性集;D 为决策属性集且 $C \cap D = \varnothing$;$V = \cup \{V_a \mid a \in C \cup D\}$,$V_a$ 为属性 a 的值域;$f: U \times (C \cup D) \rightarrow V$ 是一个信息函数,它对一个对象的每一个属性赋予一个信息值,即 $\forall a \in C \cup D, x \in U$,有 $f(x,a) \in V_a$;每一个属性子集 $P \subseteq C \cup D$ 决定了一个二元不可区分关系 $IND(P)$:$IND(P) = \{(x,y) \in U \times U \mid f(x,a) = f(y,a), \forall a \in P\}$,关系 $IND(P)$ 可确定 U 的一个划分,用 $U/IND(P)$ 表示,简记为 U/P,U/P 中的任何元素 $[x]_P = \{y \mid f(x,a) = f(y,a), \forall a \in P\}$ 称为等价类(划分块)。

定义 2[1] 设 $X \subseteq U$ 为论域的一个子集,条件属性子集 $P \subseteq C$,$\underline{P}(X) = \cup \{[x]_P \mid [x]_P \subseteq X\}$ 表示 X 的 P - 下近似集,决策属性集 D 的 P - 正域记为 $Pos_P(D)$,定义为:

$$Pos_P(D) = \cup \{\underline{P}(X) \mid X \in U/D\}$$

定义 3 在决策表 $S = (U, C, D, V, f)$ 中,称 $Pos_C(D)$ 为决策表 S 的一致对象集,$U - Pos_C(D)$ 为决策表 S 的不一致对象集;若 $Pos_C(D) = U$,则称决策表 S 为一致决策表,否则称决策表 S 为不一致决策表[8]。

在决策表 $S = (U, C, D, V, f)$ 中,若 S 是一致决策表,则决

收稿日期:2006 - 10 - 23;**修订日期**:2006 - 12 - 17
基金项目:河南省自然科学基金资助项目(0511011500);河南省高校新世纪优秀人才支持计划资助项目(HANCET)
作者简介:孙林(1979 -),男,河南南阳人,硕士研究生,主要研究方向:粗糙集理论、数据挖掘; 徐久成(1964 -),男,河南洛阳人,教授,博士,主要研究方向:粗糙集理论、粒计算、数据挖掘.

策树(指用树形结构来表示决策集合,这些决策集合通过对数据集的分类产生决策规则)的各叶子节点只对应相同决策类的对象,即每个叶子节点对应的是确定性决策规则,其可信度(对象的条件概率分布)等于1;否则决策树的某些叶子节点对应不同决策类的对象,这样的叶子节点对应的是不确定性决策规则,其可信度小于1。

定义4　在决策表 $S=(U,C,D,V,f)$ 中,决策规则为隐含式,记为 $(C_1,c1)\wedge(C_2,c2)\wedge\cdots\wedge(C_k,ck)\rightarrow(D,d)$,其中 $ci\in V_{C_i},C_i\in C,i=1,2,\cdots,k,k\leqslant|C|,d\in V_D$。

2　新的条件熵和决策树规则提取方法

在决策应用中,决策规则的可信度和对象覆盖度都是衡量决策表"决策能力"的重要指标,但经典的知识信息熵并没有完全客观地反映决策表"决策能力"的变化情况。本节首先分析现有信息熵的局限性,定义新的条件熵;然后改进传统启发式属性选择的标准,在此基础上定义新的属性重要性,并以新的属性重要性为启发式信息,构造决策树;最后设计一个规则约简过程,简化所提取的决策规则。

2.1　现有信息熵的局限性

定义5[9]　设 U 是一个论域,属性集合 $Q(U/Q=\{Y_1,Y_2,\cdots,Y_m\})$ 相对于属性集合 $P(U/P=\{X_1,X_2,\cdots,X_n\})$ 的条件熵 $H(Q|P)$ 定义为:

$$H(Q|P)=-\sum_{i=1}^{n}p(X_i)\sum_{j=1}^{m}p(Y_j|X_i)\log(p(Y_j|X_i))$$

其中 $p(X_i)=|X_i|/|U|,p(Y_j|X_i)=|X_i\cap Y_j|/|X_i|$,$i=1,2,\cdots,n,j=1,2,\cdots,m,|X|$ 表示集合 X 的基数。

在决策表 $S=(U,C,D,V,f)$ 中,属性约简的最终目标是在保持决策表 S"决策能力"不变的前提下,去除多余条件属性。由文献[9]的基于条件信息熵的决策表约简算法分析可知,一个条件属性是否可以约简,取决于删除该条件属性后决策表 S 产生的条件熵是否改变。由于决策表 S 中一致对象集 $Pos_C(D)$ 产生的条件熵为0,所以决策表 S 的条件熵改变是由不一致对象集 $U-Pos_C(D)$ 产生的,而决策表 S 删除某一条件属性后,产生新的不一致对象集属于各决策属性分类的概率分布改变,就会引起条件熵发生变化。因而,现有基于条件信息熵的约简算法对决策表的决策能力衡量标准表现在以下两个方面:

1)产生的确定性决策规则数目不变;

2)产生的不确定性决策规则的可信度不变。

若决策表产生确定性决策规则的数目不变,就意味着这些决策规则的可信度不变(可信度仍为1)。因此,现有基于条件信息熵的决策表约简算法只考虑所有决策规则在约简后其可信度是否发生变化。

然而,在决策应用中,决策规则除了其可信度外,规则对样本(对象)的覆盖度也是衡量其"决策能力"的重要指标。因此,在经典粗糙集理论中,文献[9]的约简算法存在局限性,不能客观地反映决策能力的实质。

2.2　新的条件熵

为客观有效地反映知识约简后决策表"决策能力"的真实变化情况,本节提出新的条件熵,以克服现有信息熵的局限性。

定义6[10]　设 U 是一个论域,属性集合 $R(U/R=\{R_1,R_2,\cdots,R_m\})$ 的信息熵定义为:

$$E(R)=\sum_{i=1}^{m}\frac{|R_i|}{|U|}\left(1-\frac{|R_i|}{|U|}\right)$$

其中 $|R_i|/|U|$ 表示 R_i 在论域 U 上的概率。

为了研究能够体现对象覆盖度的知识信息熵,引入下面的引理。

引理1[11]　设 P、Q 为论域 U 上的两个等价关系集合,则有 $U/(P\cup Q)=U/P\cap U/Q$ 成立。

引理1的证明参考文献[11]。

这样,在决策表 $S=(U,C,D,V,f)$ 中,属性集合 $P\cup D(P\subseteq C)$ 的信息熵可有如下定义:

定义7　设 U 是一个论域,$P(U/P=\{X_1,X_2,\cdots,X_n\})$ 为一个条件属性集合,$D=\{d\}(U/D=\{Y_1,Y_2,\cdots,Y_m\})$ 为决策属性集,则属性集合 $P\cup D$ 的信息熵定义为:

$$E(P\cup D)=\sum_{i=1}^{n}\sum_{j=1}^{m}\frac{|X_i\cap Y_i|}{|U|}\left(1-\frac{|X_i\cap Y_i|}{|U|}\right)$$

在 $P\cup D$ 的信息熵定义中,$|X_i\cap Y_j|/|U|$ 代表了该决策规则的对象覆盖程度,而在现有信息熵的定义中,$p(Y_j|X_i)=|X_i\cap Y_j|/|X_i|$ 代表了决策表产生某一规则的可信度。这样可以把两种信息熵的定义结合起来,使其更能客观地反映决策表"决策能力"的实质。在此基础上,提出了一种信息论定义形式——新的条件熵。

定义8　新的条件熵。设 U 是一个论域,P 是 U 上的一个条件属性集合,$D=\{d\}$ 为决策属性集,则 P 关于决策属性 d 的新的条件熵记为 $H(D;P)$,定义为:

$$H(D;P)=H(D|P)-E(P\cup D)。$$

2.3　决策树规则提取方法

在各种构造决策树方法中,比较有影响的是ID3算法[6]。它用信息增益作为在各级非叶节点上选择属性的标准,获得对象集最大的类别信息。但这种方法并不是最优的,即决策树的节点不是最少的。这种启发式方法往往偏向于选择属性取值较多的属性,而属性值较多的属性却不总是最优的属性,并且ID3学习简单的逻辑表达式能力较差[7]。本文针对以上问题提出如下的改进方案:在构造决策树的过程中,改进各级非叶节点属性的选择标准,重点考虑决策树中不同分支节点上属性重要性的计算,避免ID3算法中子树重复和某些属性被多次选择的缺点,便于得到更优的决策树。

定义9　新的属性重要性。设 U^* 为论域 U 上与决策树某分支节点相关的对象集(若 U^* 为决策树根节点相关的对象集,那么 $U^*=U$。),C 为条件属性集,$D=\{d\}$ 为决策属性集,$B\subseteq C$,则任意属性 $a\in C-B$ 关于 U^* 的属性重要性定义为:

$$SGF(a,B,U^*,D)=H(D;B)-H(D;B\cup\{a\})$$

特别当 $B=\varnothing$ 时,$SGF(a,\varnothing,U^*,D)=-H(D;\{a\})$。

$SGF(a,B,U^*,D)$ 的值越大,说明在已知 B 的条件下,属性 $a\in C-B$ 关于知识 B 就越重要。在计算 $SGF(a,B,U^*,D)$ 的过程中,每次循环时条件属性子集 B 的 $H(D;B)$ 均不变,这使得 $SGF(a,B,U^*,D)$ 最大的属性 a 就是 $H(D;B\cup\{a\})$ 最小的属性。因此,把 $SGF(a,B,U^*,D)$ 作为搜索最小或次优知识约简的启发式信息时,只需计算 $H(D;B\cup\{a\})$,就可以避

免计算 $H(D;B)$,减少了计算量,进而减小了搜索空间和时间。

由此可见,以 $SGF(a,B,U^*,D)$ 为启发式信息的约简算法,必须计算 $H(D;B\cup\{a\})$。为降低该方法的时间复杂度,就需要研究计算 $H(D;B\cup\{a\})$ 的高效算法,由文献[9]中的定理1可得到算法1。

算法1

输入:决策表 $S=(U^*,C,D,V,f)$ 和 $B\subseteq C$;

输出:划分 $U^*/(D\cup B\cup\{a\})$ 和 $H(D;B\cup\{a\})$。

步骤1:计算划分 $U^*/(B\cup\{a\})$ 和 $U^*/(D\cup B\cup\{a\})$;

步骤2:计算 $H(B\cup\{a\})$,$H(D\cup B\cup\{a\})$ 和 $E(D\cup B\cup\{a\})$;

步骤3:计算 $H(D;B\cup\{a\})=H(D\cup B\cup\{a\})-H(B\cup\{a\})-E(D\cup B\cup\{a\})$;

步骤4:输出 $H(D;B\cup\{a\})$ 和划分 $U^*/(D\cup B\cup\{a\})$,结束。

用文献[12]中计算划分的方法,步骤1的时间复杂度为 $O((|B|+2)|U|)$,步骤2的时间复杂度为 $O(|U|)$,因而算法1最坏的时间复杂度为 $O(|C||U|)$。

在算法1的基础上,从空树 T 开始,以 $SGF(a,B,U^*,D)$ 最大的属性 a 为分支节点(包括根节点),也就是选择 $H(D;B\cup\{a\})$ 最小的属性 a 为分支节点,自顶向下递归构造决策树。在选择一个新的属性时,不仅要考虑它基于所有对象集是最重要的,而且要考虑它在相关对象集上也是最重要的,这样就能有效改进选择新属性的启发式函数,达到更好的分类效果,弥补ID3算法容易导致决策树中子树重复和某些属性在同一决策树中被多次选择的不足,可以得到更优的决策树。

算法2

输入:对象集 U^*,条件属性集 C,决策属性集 $D=\{d\}$;

输出:最简决策树 T。

步骤1:合并 U^* 中的相同对象;

步骤2:初始化 $B=\varnothing$,T 为空树;

步骤3:对任意属性 $a\in C-B$,计算 $H(D;B\cup\{a\})$;

步骤4:选择使 $H(D;B\cup\{a\})$ 最小的属性 a 为决策树 T 的根节点(或分支节点),

1)如果有多个属性同时使 $H(D;B\cup\{a\})$ 达到最小值,那么从中选取使 $U^*/(D\cup B\cup\{a\})$ 构成等价类最少的属性 a,即 $D\cup\{a\}$ 构成的等价类能够覆盖更多的对象;

2)如果仍有多个属性使 $|U^*/(D\cup B\cup\{a\})|$ 达到最小值,那么选择顺序靠前的属性。

步骤5:用选择的属性 a 对 U^* 进行分类,即计算 $U^*/\{a\}=\{U_1^*,U_2^*,\cdots,U_t^*\}$,开始建立子决策表(即决策树的分支) $S_i=(U_i^*,C,D,V,f)$,其中 $i=1,2,\cdots,t$;

步骤6:如果分支 $S_i(i=1,2,\cdots,t)$ 中 U_i^* 的所有对象具有相同的决策属性值,那么在分支 S_i 下生成一个叶子节点,标识其决策属性值,否则 $B=B\cup\{a\}$;

步骤7:如果 $B=C$ 或 U^* 被决策树分支完全分类,那么输出决策树 T,算法结束;否则转步骤3。

用算法1的方法,可得步骤3到步骤7总的最坏时间复杂度为 $O(|C||U|)+O((|C|-1)|U|)+O((|C|-2)|U|)+\cdots+O(|U|)=O(|C|^2|U|)$,因而算法2最坏的时间复杂度为 $O(|C|^2|U|)$。

在决策树中,每个叶子节点就是一个分类,从根到叶子节点的一条路径就对应一条决策规则。下面在算法2的基础上设计一个规则约简过程,用来简化所提取的决策规则。

算法3

步骤1:遍历决策树每个分支中根到叶子节点的所有路径,生成决策规则集;

步骤2:简化决策树分支中的每一条决策规则,如果该决策规则中的任一非叶节点去掉后,在所属分支中仍能唯一表示,那么继续去掉第2,3,… 个非叶节点,直到不能在所属分支中唯一表示;

步骤3:输出最小决策规则集,结束。

分析可得算法3最坏的时间复杂度为 $O(|C||U|)$。

算法3中步骤2对循环提取的原始决策规则进行简化,删除所有不影响规则表达的冗余条件属性及属性值,简化决策规则,这就能保证所提取的决策规则最小,即规则所含的属性及属性值最少,且在约简表中唯一表示。

3 应用实例分析与比较

表1给出了一致决策表 $S=(U,C,D,V,f)$,其中 $U=\{1,2,\cdots,14\}$,$C=\{a_1,a_2,a_3,a_4\}$,$D=\{d\}$。

表1 一致决策表

论域 U		1	2	3	4	5	6	7	8	9	10	11	12	13	14
条件属性集 C	a_1	1	1	2	3	3	3	2	1	1	3	1	2	2	3
	a_2	1	1	1	2	3	3	3	2	3	2	2	2	1	2
	a_3	1	1	1	1	0	0	0	1	0	0	0	1	0	1
	a_4	0	1	0	0	0	1	1	0	0	0	1	1	0	1
决策属性 d		0	0	1	1	1	0	1	0	1	1	1	1	1	0

用一致决策表(见表1)来验证上述算法的有效性,可以得到一棵与最小确定性决策规则集(见表2)对应的最小决策树(见图1)。

表2 最小确定性决策规则集

序号	决策规则
1	$(a_1,1)\wedge(a_3,1)\rightarrow(d,0)$
2	$(a_1,1)\wedge(a_3,0)\rightarrow(d,1)$
3	$(a_1,2)\rightarrow(d,1)$
4	$(a_1,3)\wedge(a_4,0)\rightarrow(d,1)$
5	$(a_1,3)\wedge(a_4,1)\rightarrow(d,0)$

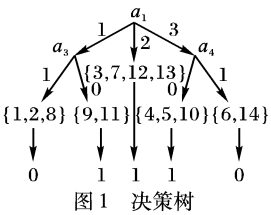

图1 决策树

由图1可知,算法2可得到与文献[5]相同的单变量决策树。对于表1所示的一致决策表,文献[3]中RITIO算法得到的规则集共有7条规则,其中有一条规则是不一致的,它与表1的第6个对象矛盾,文献[13]中LEM2算法对于表1

得到的规则集也是 7 条规则,以上两种算法得到的规则集均比本文算法 3 得到的规则集数目多;对于不一致决策表,在算法 2 得到的决策树中,不一致对象对应的决策属性值为两个,算法 3 得到的不一致对象对应简化后的不确定性决策规则的可信度均小于 1。

4　结语

本文提出一种决策树规则提取方法,它以新的条件熵来度量属性重要性。该方法具有如下特点:1)弥补了现有信息熵反映决策表"决策能力"的局限性;2)改进了传统启发式方法中选择属性的标准;3)不需要在构造决策树阶段前进行属性约简;4)设计了一个规则约简过程来简化决策规则,增强规则的泛化能力。实例分析的结果表明,该方法不仅有助于进一步加深对粗糙集理论中规则提取算法的认识,也有助于时效性更优算法的推广使用。

需要指出的是,该方法没有考虑到在大型数据集分析中,存在数据测量的误差、数据获取能力的不足、噪声干扰等原因,在一定程度上可能制约其处理复杂应用问题的有效性。

致谢　向对本文工作给予支持和建议的闫林副教授、田沄博士研究生表示感谢。

参考文献:

[1] PAWLAK Z. Rough sets: theoretical aspects of reasoning about data [M]. Dordrecht: Kluwer Academic Publishers, 1991.

[2] 代建华, 潘云鹤. 一种基于分类一致性的决策规则获取算法[J]. 控制与决策, 2004, 19(10): 1086 - 1090.

[3] WU XD, URPANI D. Induction by attribute elimination[J]. IEEE Trans on Knowledge and Data Engineering, 1999, 11(5): 803 - 812.

[4] 常犁云, 王国胤, 吴渝. 一种基于 Rough Set 理论的属性约简及规则提取方法[J]. 软件学报, 1999, 10(11): 1206 - 1211.

[5] 苗夺谦, 王珏. 基于粗糙集的多变量决策树构造方法[J]. 软件学报, 1997, 8(6): 425 - 431.

[6] QUINLAN JR. Induction of decision trees[J]. Machine Learning, 1986, 1: 81 - 106.

[7] TU PL, CHUNG JY. A new decision-tree classification algorithm for machine learning[A]. Proceedings of the 1992 IEEE International Conference on Tools for Artificial Intelligence[C]. Arlington, Virginia, USA: IEEE Computer Society, 1992. 370 - 377.

[8] 王国胤. 决策表核属性的计算方法[J]. 计算机学报, 2003, 26(5): 611 - 615.

[9] 王国胤, 于洪, 杨大春. 基于条件信息熵的决策表约简[J]. 计算机学报, 2002, 25(7): 759 - 766.

[10] LIANG JY, CHIN KS, DANG CY, *et al*. A new method for measuring uncertainty and fuzziness in rough set theory[J]. International Journal of General System, 2002, 31(4): 331 - 342.

[11] GUAN JW, BELL DA. Rough computational methods for information systems[J]. Artificial Intelligence, 1998, 105: 77 - 103.

[12] 徐章艳, 刘作鹏, 杨炳儒, 等. 一个复杂度为 max(O(|C||U|), O(|C|²|U|))的快速属性约简算法[J]. 计算机学报, 2006, 29(3): 391 - 399.

[13] GRZYMALA-BAUSSE DM, GRZYMALA-BAUSSE JW. The usefulness of a machine learning approach to knowledge acquisition [J]. Computational Intelligence, 1995, 11(2): 268 - 279.

(上接第 874 页)

5)如果 isSet == TRUE,转(10);

6)定义一个变量集合的局部变量 useundef;

7)运行操作 useundef = recursiveUseunDef(CallBB);

8)如果 useundef 集合为空,则设置函数 Callee 的返回类型为 void,转(10);

9)根据集合 useundef 中排在第一位的位置变量信息 Q 设置被调函数 Callee 的函数返回类型,如 Q 是 r8 的表示形式,则设置函数 Callee 的返回类型为 int;如 Q 是 f8 的表示形式则设置函数 Callee 的返回类型为 float;

10)退出。

应用改进的函数返回类型恢复算法对例子 returncallee.c 及变体(数据类型改为 float)的 gcc 0 ~ 2 和 icc 0 ~ 2 级优化程序进行函数返回类型恢复测试,结果显示函数 add4 和 add2 的返回类型都被正确恢复。

3　实验结果

上述改进后的函数返回类型恢复算法已应用到 ITA 翻译器系统中,该系统成功翻译了 Cexample 测试集的 64 个整型和浮点型实例,以及 SPECcpu2000 测试集中 gzip、bzip2 压缩解压程序、mcf 车辆调度程序和 parser 语法分析程序,翻译前后的运行结果完全相同。这些程序,尤其是 SPECcpu2000 中的四个测试程序规模较大,其中含有大量的函数调用,且返回值类型各异。对它们应用上述改进的函数返回类型恢复算法进行处理,得到了正确运行结果,这就验证了上述改进的函数返回类型恢复算法的正确性和可行性。

4　结语

传统的函数返回类型恢复方法是从 caller 的角度,以 useundef 变量集合为基础实施函数返回类型的恢复;改进的函数返回类型恢复方法是从 callee 的角度,以 def 变量集合为基础实施函数返回类型的恢复。一个为正向流分析法,一个为反向流分析法,不论哪一个都有不足之处,只有将两者有机的结合起来才能达到实用的目的。但由于优化级别的不同和机器体系结构的差异,可能会存在某些细节需要特殊考虑,这也是以后要研究的内容。

致谢　忠心的感谢赵荣彩和庞建民老师耐心的教导和指导以及实验室的各位同事。

参考文献:

[1] ALTMAN ER, KAELI K, SHEFFER Y. Welcome to the opportunities of binary translation computer[J]. IEEE Computer Society Press, 2000, 33(3): 40 - 45.

[2] CIFUENTES C, EMMERIK M, REMSEY N. The university of queensland binary translator(UQBT) framework[Z]. 2000.

[3] EVANS JS. 安腾体系结构[M]. 蒋敬旗, 译. 北京: 清华大学出版社, 2005.

[4] 齐宁, 赵荣彩, 付文. 二进制翻译中的库函数识别技术研究[J]. 计算机应用, 2006, 26(4): 983 - 985.

[5] 陈火旺, 刘春林. 程序设计语言编译原理[M]. 北京: 国防工业出版社, 2000.

[6] CIFUENTES C, FRABOULET A. Interprocedural data flow recovery of high-level language code from assembly[Z]. 2001.