广西科学院学报
Journal of Guangxi Academy of Sciences

2010,26(4):401~405
Vol. 26, No. 4 November 2010

基于 OBDD 的描述逻辑 *ALCIO* 判定算法*

OBDD-Based Decision Algorithm for the Description Logic *ALCIO*

常 亮,高 申,李德波,古天龙
CHANG Liang, GAO Shen, LI De-bo, GU Tian-long

(桂林电子科技大学计算机科学与工程学院,广西桂林 541004)
(School of Computer Science and Technology, Guilin University of Electronic Technology, Guilin, Guangxi, 541004, China)

摘要:给定描述逻辑 *ALCIO* 中的任一知识库,应用 NNF 变换和 FLAT 规则对其进行预处理,通过一个重构过程将知识库中 TBox 模型转化为布尔函数,然后将布尔函数转换为有序二叉决策图(OBDD)表示形式,从而调用已有的 OBDD 软件包进行可满足性判定,实现描述逻辑 *ALCIO* 的判定算法。该算法在实现描述逻辑的推理方面与经典的 Tableau 判定算法在性能上可以相互弥补和配合。

关键词:描述逻辑 有序二叉决策图 枚举算子 可满足性判定
中图法分类号:TP301 **文献标识码**:A **文章编号**:1002-7378(2010)04-0401-05

Abstract: A satisfiability-checking algorithm based on Ordered Binary Decision Diagram (OBDD) is presented in this paper for the description logic *ALCIO*. Starting from an *ALCIO* ontology, the algorithm introduces the NNF transformation rule and the FLAT rule to do some preprocessing; then the TBox model of the knowledge base is reconstructed and transformed into some Boolean formulas; finally, these Boolean formulas are represented as OBDDs, based on the existing OBDD software package that can be called for deciding the satisfiability of *ALCIO* ontologies. The experimental results indicate that, according to the performance, the satisfiability-checking algorithm based on OBDD can complement the classical Tableau deciding algorithm.

Key words: description logic, ordered binary decision diagram, nominals, satisfiability-checking

作为一类用于知识表示的形式化工具,描述逻辑在信息系统、软件工程、自然语言处理等领域得到了成功应用。描述逻辑的基本思想是通过一系列构造符对应用领域的概念和角色等进行定义,然后应用这些概念和角色对应用领域中出现的对象和个体的性质进行刻画。描述逻辑的主要特点在于提供了命题逻辑所无法比拟的刻画能力,同时又保证了相关推理问题的可判定性,并且具有有效的推理机制作为支撑[1]。Tableau 算法是目前描述逻辑中最主要的推理方法,但是它并不是在所有情况下都表现最好。因此,有必要针对不同特征的本体寻找相应地更合适的推理算法[2]。

有序二叉决策图(OBDD)是一种有效地表示和处理大规模问题的数据结构,在大规模模型检测和验证等领域已经得到成功应用,在逻辑公式的可满足性判定方面也具有巨大的应用潜力[3]。为描述逻辑寻求更有效的判定算法是研究者一直在努力探索的问题。Pan 等[4]将 OBDD 应用于模态逻辑的判定过程,实验结果表明该方法在逻辑公式中出现模态词过多时具有明显的性能优势。Rudolph 等[5]将 OBDD 运用于描述逻辑 SHIQ 的推理,给出了相应的算法并从理论上证明了算法的正确性。但是 Rudolph 等并没有实现他们提出的算法,没有给出这

收稿日期:2010-09-21

作者简介:常 亮(1980-),男,博士,副教授,硕士生导师,主要从事知识表示与推理、形式化方法和智能规划研究。

* 国家自然科学基金项目(60903079, 60963010)资助。

种算法的性能分析，进而无法比较基于OBDD的描述逻辑判定算法与传统的Tableau判定算法的优劣；此外，Rudolph等给出的算法不支持描述逻辑中常见的枚举算子。

本文在文献[5]研究工作的基础上，基于OBDD技术给出支持枚举算子的描述逻辑判定算法，并实现这种判定算法，从而能够从实际性能的角度将这种算法与传统的Tableau算法进行比较，验证OBDD技术在描述逻辑的可满足性判定方面存在的潜力。由于描述逻辑家族由一系列具体的描述逻辑系统组成，为了表述简便，我们从含有枚举算子并且可满足性问题为EXPTIME-完全的众多描述逻辑系统中选取最为简单的描述逻辑系统 *ALCIO*，以其为对象研究基于OBDD的判定算法。本文的算法在整体上遵循Rudolph等给出的判定算法的基本思路[5]，将Rudolph等给出的支持描述逻辑SHIQ的判定算法首先裁剪到支持描述逻辑ALC，然后在其中引入对枚举算子 O 的处理。具体来说，给定描述逻辑 *ALCIO* 中的任一知识库，我们首先应用NNF变换和FLAT规则对其进行预处理，然后通过一个重构过程将知识库中TBox的模型转化为布尔函数，最后将布尔函数转换为OBDD表示形式，从而调用已有的OBDD软件包实现相应的推理。

1 相关定义

描述逻辑家族由一系列描述逻辑系统组成，不同的描述逻辑系统提供了不同的构造符。描述逻辑 *ALCIO* 提供了用于构造角色的逆算子以及用于构造概念的合取、析取、否定、存在限定、值限定和枚举等构造符。作为一个逻辑语言，*ALCIO* 的基本符号包括由概念名组成的集合 N_C、由角色名组成的集合 N_R、以及由个体名组成的集合 N_I。从这些符号出发，可以通过上述构造符递归地生成 *ALCIO* 中的角色和概念。描述逻辑 *ALCIO* 的每个解释结构是一个二元组 $I=(\Delta, \cdot^I)$，其中的 Δ 是由个体组成的非空集合，解释函数 $\cdot^I$ 将每个概念名 $A_i \in N_C$ 解释为 Δ 的某个子集 $A_i^I \subseteq \Delta$，将每个角色名 $R_i \in N_R$ 解释为 Δ 上的某个二元关系 $R_i^I \subseteq \Delta \times \Delta$，将每个个体名 $a \in N_I$ 解释为 Δ 中的某个元素 $a^I \in \Delta$。在此基础上，可以将解释函数 $\cdot^I$ 递归地扩展到对 *ALCIO* 中的每个角色和每个概念进行解释。描述逻辑 *ALCIO* 中具体的语法和语义定义见表1。

表1 描述逻辑 *ALCIO* 的语法和语义

构造符	语法	语义
角色名	R_i	$R_i^I \subseteq \Delta \times \Delta$
概念名	A_i	$A_i^I \subseteq \Delta$
逆反	R^-	$\{(x,y) \mid (y,x) \in R^I\}$
否定	$\neg C$	Δ^I / C^I
合取	$C \sqcap D$	$C^I \cap D^I$
析取	$C \sqcup D$	$C^I \cup D^I$
存在限定	$\exists R.C$	$\{x \in \Delta \mid$ 存在 $y \in \Delta$ 使得 $(x,y) \in R^I$ 并且 $y \in C^I\}$
值限定	$\forall R.C$	$\{x \in \Delta \mid$ 对于任一 $y \notin \Delta$：如果 $(x,y) \in R^I$ 则必然有 $y \in C^I\}$
枚举	$\{a\}$	$\{a^I\}$

令 C、D 为 *ALCIO* 中的概念，R 为 *ALCIO* 中的角色，a、b 为个体名；则将 $C \sqsubseteq D$、$C(a)$ 和 $R(a,b)$ 分别称为一般概念包含公理、概念断言和角色断言。令 $I=(\Delta, \cdot^I)$ 为描述逻辑 *ALCIO* 的一个解释结构，在其基础上对一般概念包含公理、概念断言和角色断言的语义定义如下：

(1) $I \models C \sqsubseteq D$ 当且仅当 $C^I \subseteq D^I$；

(2) $I \models C(a)$ 当且仅当 $a^I \in C^I$；

(3) $I \models R(a,b)$ 当且仅当 $(a^I, b^I) \in R^I$。

描述逻辑中的知识库通常由TBox和ABox组成，其中的TBox是由一般概念包含公理组成的有限集合，ABox是由概念断言、角色断言、以及概念断言和角色断言的否定形式等组成的有限集合。在这里我们只考察对TBox的推理，即判断给定的任一TBox是否为可满足的。其中，对TBox的可满足性定义如下：

对于任一TBox $\mathcal{T}$，称 $\mathcal{T}$ 是可满足的当且仅当存在某个解释结构 $I=(\Delta, \cdot^I)$ 使得对于每个一般概念包含公理 $C \sqsubseteq D \in \mathcal{T}$ 都有 $I \models C \sqsubseteq D$。

作为例子，我们抽取著名的Wine本体中的如下片断：

Wine $\sqsubseteq \exists$ locatedin. Region，

Dessertwine $\sqsubseteq \forall$ hassugar. {offdry, sweet}。

这两个一般概念包含公理描述了如下信息：每一种葡萄酒wine都存在相应的产地；餐后葡萄酒含有的糖分只有半干型和甜型两种。下面我们用 $\mathcal{T}$wine 表示由以上一般概念包含公理组成的TBox，并以其为例子对基于OBDD的 *ALCIO* 判定算法本进行说明。

2 基于 OBDD 的描述逻辑 *ALCIO* 判定算法

对于待判定可满足性的任一 TBox $\mathcal{T}$,我们依次对其进行以下处理。首先,将 $\mathcal{T}$ 中每个形如 $C \sqsubseteq D$ 的一般概念包含公理转换为 $\top \sqsubseteq \neg C \sqcup D$ 的形式,进而改写为概念表达式 $\neg C \sqcup D$ 的形式。在此基础上,对 $\mathcal{T}$ 中的每个概念进行 NNF 变换,即将其中出现的每个概念表达式转化为与之等价的否定范式。这种转换可以很容易地借助德摩根定律来实现,并且转化过程所需要的时间开销与 TBox 的大小成线性关系;限于篇幅,这里不再阐述详细的转换过程。在下文中我们假设所考察的 TBox 均已经进行上述预处理。

对于任一 TBox $\mathcal{T}$,我们通过以下步骤对其进行处理:

(1)在 $\mathcal{T}$ 中选择最外层形如 $\Diamond U.D$ 的任一概念,其中的 $\Diamond \in \{\exists, \forall\}$,$D$ 为非原子概念;用某个在 $\mathcal{T}$ 中没有出现过的概念名 F 对 D 进行替换,同时将 $(\neg F \sqcup D)$ 添加到 $\mathcal{T}$ 中;反复进行上述过程,直到 $\mathcal{T}$ 中不存在形如 $\Diamond U.D$ 并且 D 为非原子概念的概念表达式;

(2)将 $\mathcal{T}$ 中所有形如 $\{a_1, a_2, \cdots, a_n\}$ 的概念改写为概念表达式 $\{a_1\} \sqcup \{a_2\} \sqcup \cdots \sqcup \{a_n\}$。

对于任一 TBox $\mathcal{T}$,将经过上述各个步骤转换后得到的概念集合记为 FLAT($\mathcal{T}$)。根据以上转换过程容易证明如下定理。

定理 1 对于描述逻辑 *ALCIO* 中的任一 TBox $\mathcal{T}$,$\mathcal{T}$ 是可满足的当且仅当 FLAT($\mathcal{T}$)是可满足的。

例如,对于前面例子中的 TBox $\mathcal{T}$wine,经过转换后得到的 FLAT($\mathcal{T}$wine)由以下概念表达式组成:

$\neg$ Wine $\sqcup \exists$ locatedin. Region

$\neg$ Dessertwine $\sqcup \forall$ hassugar. F

$\neg$ F $\sqcup$ {offdry} $\sqcup$ {sweet}。

在进行了上述处理之后,接下来的算法步骤与文献[5]类似。其中,为了对算法的正确性进行严格证明,文献[5]引入了 Domino 集的概念,将其作为语义模型的一种抽象表示来证明算法的正确性。与文献[5]类似,我们可以针对描述逻辑 *ALCIO* 引入相应的 Domino 集,进而对下面将给出的判定算法的正确性进行证明。引入 Domino 集之后的相关术语、符号以及证明过程参见文献[5]给出的内容。在给出基于 OBDD 的描述逻辑 *ALCIO* 判定算法之前我们先引入几个基本符号。

首先,对于任一概念表达式 C,用 $P(C)$ 表示由 C 中出现的所有概念表达式组成的集合,递归定义如下:

$$P(C) := \begin{cases} P(D), \text{若 } C \text{ 形如 } \neg D; \\ P(D) \cup P(E), \text{若 } C \text{ 形如 } D \sqcap E \text{ 或 } D \sqcup E; \\ \{C\} \cup P(D), \text{若 } C \text{ 形如 } \exists U.D \text{ 或 } \forall U.D; \\ \{C\}, \text{若 } C \text{ 为概念名或形如} \{a_i\}; \end{cases}$$

在此基础上,对于将一般概念包含公理表示为概念表达式形式之后的任一 TBox $\mathcal{T}$,用 $P(\mathcal{T})$ 表示集合 $\bigcup_{C \in \mathcal{T}} P(C)$。

其次,令 $\mathcal{T}$ 是相应对其可满足性进行判定的 TBox,令 $\mathcal{R}$ 是由 $\mathcal{T}$ 中出现的所有角色名组成的集合;则将基于 OBDD 的判定算法中用到的变量集合定义为 $\mathrm{Var}(\mathcal{T}) := \mathcal{R} \cup (P(FLAT(\mathcal{T})) \times \{1,2\})$。在此基础上,对于每个概念表达式 C,用 $[[C]]$ 表示与该表达式对应的变量,递归定义如下:

$$[[C]] := \begin{cases} <C, l>, \text{如果 } C \in P(FLAT(\mathcal{T})), \\ \neg [[D]], \text{如果 } C \text{ 形如 } \neg D, \\ [[D]] \wedge [[E]], \text{如果 } C \text{ 形如 } D \sqcap E, \\ [[D]] \vee [[E]], \text{如果 } C \text{ 形如 } D \sqcup E。 \end{cases}$$

算法 1 对于描述逻辑 *ALCIO* 中的任一 TBox $\mathcal{T}$,令 $C := P(\mathrm{FLAT}(\mathcal{T}))$,依次进行以下操作:

(1) 令 $i := 0$;构造布尔函数 $[[C]]_0 := f^{Bc} \wedge f^{Ex} \wedge f^{Uni}$,其中:

$f^{Bc} := \bigwedge_{G \in T} [[G]]$,

$f^{Ex} := \bigwedge_{\exists U.C \in C} <C, 2> \wedge U \rightarrow < \exists U.C, 1>$,

$f^{Uni} := \bigwedge_{\exists U.C \in C} < \forall U.C, 1> \wedge U \rightarrow <C, 2>$。

(2) 令 $i := i+1$;在 $[[C]]_i$ 的基础上构造布尔函数 $[[C]]_{i+1} := [[C]]_i \wedge f^{Delex} \wedge f^{Deluni} \wedge f^{Sym}$,其中:

$f^{Delex} := \bigwedge_{\exists U.C \in C} < \exists U.C, 1> \rightarrow [[C]]_i \wedge U \wedge <C, 2>$,

$f^{Deluni} := \bigwedge_{\exists U.C \in C} < \forall U.C, 1> \rightarrow \neg([[C]]_i \wedge U \wedge \neg <C, 2>)$,

$f^{Sym} := [[C]]_i(\{<D,1> | <D,2> \in V\} \cup \{\mathrm{Inv}(R) \mid R \in V\} \cup \{<D,2> | <D,1> \in V\})$。

(3) 如果 $[[C]]_{i+1} = [[C]]_i$,则令布尔函数 $[[C]] := [[C]]_i$,否则跳转到步骤(2)继续执行。

(4) 如果带入变量集 $\mathrm{Var}(\mathcal{T})$ 的每个子集 V 之后

都使得布尔函数$[[C]]$为*false*，则返回结果“TBox $\mathcal{T}$为不可满足”，否则返回结果“TBox $\mathcal{T}$可满足”。

借助于Domino集等概念，与文献[5]类似，可以证明算法1具有下面的性质。

定理2 算法1是可终止的，并且算法1返回结果“TBox $\mathcal{T}$为不可满足”当且仅当$\mathcal{T}$是不可满足的。

例如，对于前面例子中得到的FLAT($\mathcal{T}$wine)，应用算法1将得到如下布尔函数：

$f^{Bc} := (\neg <\text{Wine}, 1> \vee <\exists \text{locatedin.Region}, 1>) \wedge (\neg <\text{Dessertwine}, 1> \vee <\forall \text{hassugar.F}, 1>) \wedge (\neg <\text{F}, 1> \vee <\{\text{offdry}\}, 1> \vee <\{\text{sweet}\}, 1>)$，

$f^{Ex} := [<\text{Region}, 2>] \wedge [\text{locatedin}] \rightarrow [<\exists \text{locatedin.Region}, 1>]$，

$f^{Uni} := [<\forall \text{hassugar.F}, 1>] \wedge [\text{hassugar}] \rightarrow [<\text{F}, 2>]$，

$f^{Delex} := [<\exists \text{locatedin.Region}, 1>] \rightarrow [\text{locatedin}] \wedge [<\text{Region}, 2>]$，

$f^{Deluni} := [<\forall \text{hassugar.F}, 1>] \rightarrow \neg([\text{hassugar}] \wedge \neg[<\text{F}, 2>])$，

$[[C]]_0 := f^{Bc} \wedge f^{Ex} \wedge f^{Uni}$，

$[[C]]_1 := [[C]]_0 \wedge f^{Delex} \wedge f^{Deluni}$。

在布尔函数已知的情况下，可以通过建立OBDD来判定布尔函数的可满足性。即，在对布尔函数进行OBDD表示之后，判断是否存在从根节点出发到达1的路径；若存在，则表明相应的布尔函数是可满足的(即不是永假式)；否则表明布尔函数是不可满足的。

例如，对于上面的例子，$[[C]]_1$所对应的OBDD如图1所示。其中，为了增强可读性，图1删除了到达终端节点0的所有路径。由图1可以看出，该OBDD存在一条终点到达1的路径，因而相应的TBox $\mathcal{T}$wine是可满足的。

3 算法实验分析

由于基于Tableau算法的Pellet推理机是目前处理带有Nominals的本体时性能最好的描述逻辑推理机[6]，因此我们在实验过程中选择Pellet作为实验对比对象。为了评估算法的优劣，我们所有的实验都是在一台CPU为Inter(R) Core(TM) i7 920 @ 2.67 GHz，内存为6GB的PC机上测试，并使用了Java程序语言，以Eclipse作为开发平台，利用javabdd_1.0b2开源软件包，开发了适应于*ALCIO* - TBox可满足性检测的推理机$\text{DLR}_{\text{OBDD}}(ALCIO)$。

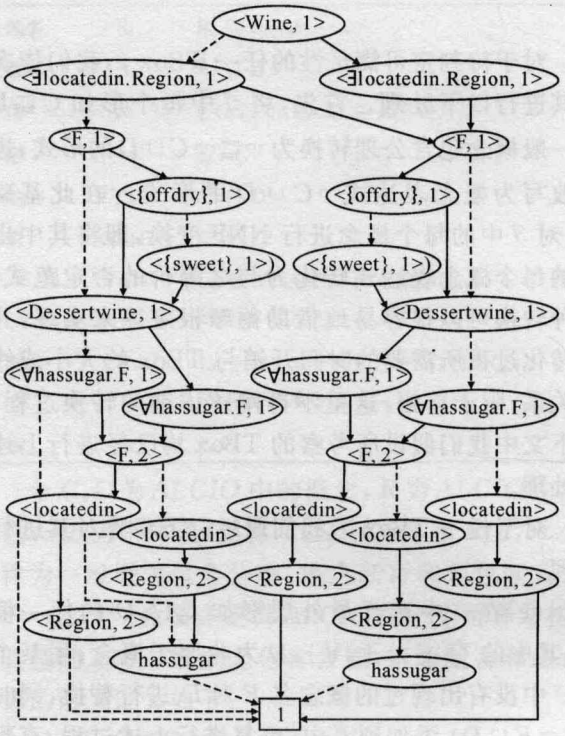

图1 $[T]_{W1}$的OBDD表示

为了说明基于OBDD的判定算法与基于Tableau算法由于重构模型的不同而导致时间开销上的差异，实验使用http://swoogle.umbc.edu/网站提供的一些著名本体作为测试的Benchmark，时间单位均为ms(表2)；由于部分本体的表达能力超出*ALCIO*的描述范围，因此我们在实验过程中对超出*ALCIO*描述能力的内容进行了裁剪。图2是对应表2的一致性检测结果。

从图2可以看出，表2的Benchmark本体虽然具有不同的表达能力，但是均适应于*ALCIO*术语推理，且具有很强代表性。同时通过图2不难发现，当对各Benchmark本体进行一致性检测时，基于本文判定算法的DLR_{OBDD}(*ALCIO*)的时间开销，虽不能在所有Benchmark本体推理中都低于Pellet的时间开销，甚至在对Tambis本体推理时，其时间开销远高于Pellet，但是在对大多数本体进行一致性检测时，其时间开销均低于Pellet时间开销；特别是，当对GO本体检测时，DLR_{OBDD}(*ALCIO*)时间开销接近Pellet时间开销的1/3。

实验结果表明，基于OBDD的描述逻辑判定算法与当前主流的Tableau判定算法之间在性能上存

表 2 各本体的参数特征

本体名	表达能力	概念名/角色名/个体名
Forest	$\mathcal{EL}$	18/2/0
Koala	ALCO	44/2/6
University	*ALCI*	97/6/0
Wine	*ALCIO*	125/6/9
Food	*ALCO*	141/6/11
Genidl	*ALCO*	159/13/46
Tambis	*ALCI*	197/11/0
Units	*ALCO*	138/4/96
Go	$\mathcal{EL}^{(\neg)}$	261/1/0
Material. thing	*AL*	306/0/0

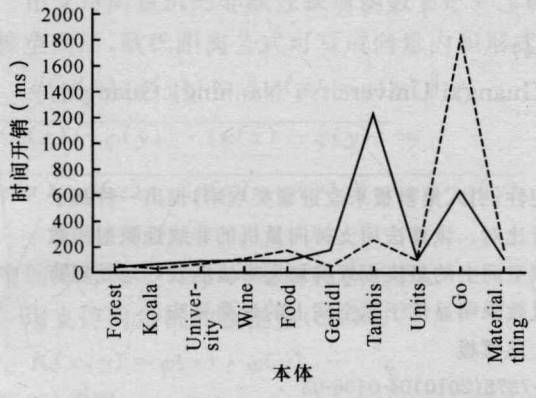

图 2 DLR_{OBDD}(*ALCIO*)和 Pellet 的一致性检测结果
——:DLR_{OBDD}(*ALCIO*); ----:Pellet.

在可以相互弥补和配合的地方。此外,由于我们只是直接地实现了基于 OBDD 的判定算法,在实现过程中尚未考虑各种优化机制,因此可以认为 OBDD 技术在描述逻辑的可满足性判定方面还存在着巨大的潜力。

4 结束语

在文献[5]的基础上,本文引入了对枚举算子的处理,给出并实现了基于 OBDD 的 *ALCIO* 判定算法。据我们所知,本文首次为基于 OBDD 的描述逻辑判定算法给出了实验分析。结果表明,在实现描述逻辑的推理方面,基于 OBDD 的判定算法与经典的 Tableau 判定算法之间存在可以相互弥补和配合的地方。该实验结果为我们进一步研究基于 OBDD 的描述逻辑判定算法打下了基础。我们下一步的工作是将基于 OBDD 的判定算法扩展到支持描述逻辑 SHOIQ 以及支持对 ABox 的推理,并对相应的算法进行优化和实现。

参考文献:

[1] Baader F, Calvanese D, McGuinness D, et al. The description logic handbook: theory, implementation and applications[M]. Cambridge: Cambridge University Press, 2002.

[2] 常亮, 史忠植, 邱莉榕, 等. 动态描述逻辑的 Tableau 判定算法[J]. 计算机学报, 2008, 31(6): 896-909.

[3] 古天龙, 徐周波. 有序二叉决策图及应用[M]. 北京: 科学出版社, 2009.

[4] Pan G Q, Sattler U, Vardi M Y. BDD-based decision procedures for the modal logic K[J]. Journal of Applied Non-Classical Logics, 2006, 16(1):169-208.

[5] Rudolph S, Krötzsch M, Hitzler P. Terminological reasoning in SHIQ with ordered binary decision diagrams: proc of the 23rd AAAI Conference on Artficial Intelligence (AAAI-08)[C]. Cambridge: AAAI Press, 2008:529-534.

[6] Sirin E, Parsia B, Grau BC, et al. Pellet: a practical OWL-DL reasoner[J]. Journal of Web Semantics, 2007, 5(2):51-53.

[7] Drechsler R, Sieling D. Binary decision diagrams in theory and practice[J]. International Journal on Software Tools for Technology Transfer, 2001, 2(3):112-136.

(责任编辑:韦廷宗)