arXiv:2303.05264v2 [cs.DB] 1 Jul 2023

含空值的不一致数据库中的 Belnap-Dunn逻辑与查询回答

C.A. MiddelburgORCID:
https://orcid.org/0000-0002-8725-0197

阿姆斯特丹大学理学院信息学研究所，荷兰阿姆斯特丹科学园区
900 号，1098 XH 阿姆斯特丹 C.A.Middelburg@uva.nl

摘要。本文探讨一阶 Belnap-Dunn 逻辑的一种扩展，称为 $\mathrm{BD}_{\perp}^{\supset,\mathsf{F}}$，以及该逻辑在关系数据库理论领域中的应用。本文从这一逻辑的角度考察关系数据库的概念、适用于关系数据库的查询概念，以及相对于关系数据库的若干查询答案概念，同时考虑数据库可能是不一致数据库和/或含空值的数据库。这一视角使得我们能够定义相对于可能不一致的数据库的查询一致答案概念，而无须借助数据库修复。对于所考察的每一种答案概念，判断某个答案是否为相对于所讨论类型数据库的查询答案都是可判定的。

Keywords: 关系数据库、不一致数据库、空值、一致查询回答、Belnap-Dunn逻辑、不定值。
1998 ACM Computing Classification: F.4.1、H.2.3、H.2.4

1 引言

在关系数据库理论领域，自20世纪80年代以来，人们普遍认为，数据库是一阶经典逻辑公式的集合，查询是一阶经典逻辑公式，而查询回答就是证明某个公式是一阶经典逻辑中一组公式的逻辑推论（例如参见 [27,12]）。采用一阶经典逻辑，使得人们难以考虑数据库可能是不一致数据库，或数据库可能是含空值的数据库的情况。

在关系数据库的逻辑视角中，不一致关系数据库是指一个不一致的公式集：按照这种意义，该公式集中至少存在一组由两个公式组成的合取，其中一个公式是另一个公式的否定，而该合取是该公式集的逻辑后承。由于在经典逻辑中，任何公式都是每个此类合取的逻辑后承，因此很难恰当地考虑数据库可能不一致这一情况。若一种逻辑中，并非每个公式都是由两个公式组成的每个此类合取的逻辑后承（其中一个公式是另一个公式的否定），则称这种逻辑为次协调逻辑。

本文对关系数据库中的空值所采取的观点可概述如下：(a) 在含有空值的关系数据库中，对于不确定的值，使用一个称为“空值”的虚设值；(b) 不确定的值是指未知或不存在的值；(c) 无论该值是未知还是不存在，都无法有意义地回答空值与任意值（包括空值本身）是否相同这一问题。关于关系数据库中的空值，这一观点的若干变体已得到研究（例如，参见 [11,30,15]）。不过，这些变体不那么基础，而且其中大多数在含有空值的关系数据库的逻辑视角中会引发类似的问题。

从关系数据库的逻辑视角来看，含空值的关系数据库是一组不完整的公式，具体而言，存在一个由两个公式构成的析取，其中一个公式是另一个公式的否定，而该析取并非这组公式的逻辑后承。由于在经典逻辑中，这类析取都是任意公式集的逻辑后承，因此很难恰当地将空值可能出现在数据库中的情况纳入考虑。若一种逻辑中，并非每个由两个公式构成的析取（其中一个公式是另一个公式的否定）都是任意公式集的逻辑后承，则称这种逻辑为次完备逻辑。

在 [22], 关系数据库的概念、适用于关系数据库的查询概念，以及相对于关系数据库的若干查询答案概念，都是从 $\mathrm{LPQ}^{\supset,\mathsf{F}}$的视角加以考察的。LPQ [26] 是 Priest悖论逻辑一阶版本的一种扩展。在此考虑了数据库可能是不一致数据库的情况，但未考虑数据库可能是含空值的数据库的情况。原因在于，$\mathrm{LPQ}^{\supset,\mathsf{F}}$ 是次协调的，却不是次完备的。还有许多其他次协调逻辑。选择 $\mathrm{LPQ}^{\supset,\mathsf{F}}$，是因为它的联结词和量词都为人们所熟悉，取自经典逻辑，而且它的逻辑后承关系与经典逻辑的逻辑后承关系密切相关。

本文引入并研究了 [23], $\mathrm{BD}^{\supset,\mathsf{F}}$，即一种带有经典联结词的一阶 Belnap-Dunn 逻辑扩展，重点关注：（a）$\mathrm{BD}^{\supset,\mathsf{F}}$ 的逻辑后承关系与具有相同联结词和量词的经典逻辑版本之间的联系；（b）在 $\mathrm{BD}^{\supset,\mathsf{F}}$ 中，可否定义一些有趣的非经典联结词，这些联结词已在先前研究过的 Belnap-Dunn 逻辑扩展中加入。$\mathrm{BD}^{\supset,\mathsf{F}}$ 既是次协调的，也是次完备的。与 $\mathrm{LPQ}^{\supset,\mathsf{F}}$ 类似，$\mathrm{BD}^{\supset,\mathsf{F}}$ 的联结词和量词都为经典逻辑中常见的形式，而且其逻辑后承关系与经典逻辑的逻辑后承关系密切相关。

[23] 的附录简要介绍了 $\mathrm{BD}^{\supset,\mathsf{F}}$ 的一种小变体，它涵盖含有不确定值的项。这种变体称为 $\mathrm{BD}^{\supset,\mathsf{F}}_{\perp}$，它也具有次一致性和次完备性。此外，它对方程的处理与上述观点相符：对于空值与任意值（包括空值本身）是否为同一值这一问题，无法给出有意义的答案。所有这些意味着，数据库既可能是不一致数据库，

也可能是含空值的数据库；当从 $\mathrm{BD}^{\supset}_{\bot}{}^{\mathsf{F},}$ 的角度考察 [22] 中讨论的概念时，这两种可能性都能得到恰当的考虑。

本文将对此展开讨论。为使本文内容完整，文中将介绍 $\mathrm{BD}^{\supset,\mathsf{F}}$ 的语言和逻辑后承关系，以及 $\mathrm{BD}^{\supset,\mathsf{F}}$ 的序列演算证明系统。$\mathrm{BD}^{\supset}_{\bot}{}^{\mathsf{F},}$ 是 $\mathrm{BD}^{\supset,\mathsf{F}}$ 的一个小幅变体。本文在 $\mathrm{BD}^{\supset}_{\bot}{}^{\mathsf{F},}$ 的框架下，定义关系数据库的概念、适用于关系数据库的查询概念，以及若干针对关系数据库中查询的答案概念，并考虑数据库可能是不一致数据库和/或含空值的数据库。与 [22], 中的做法类似，相关定义基于 [27] 中给出的定义。本文还引入了两种针对可能不一致的关系数据库中查询的一致答案概念。与 [22], 中的做法类似，其中一种令人联想到 [6] 中的一致答案概念，另一种则令人联想到 [2] 中的一致答案概念。

本文结构如下。第2、3和4节分别介绍 $\mathrm{BD}^{\supset,\mathsf{F}}$ 的语言、$\mathrm{BD}^{\supset,\mathsf{F}}$ 的逻辑后承关系，以及 $\mathrm{BD}^{\supset,\mathsf{F}}$ 的序列演算证明系统。第5节将 $\mathrm{BD}^{\supset}_{\bot}{}^{\mathsf{F},}$ 作为 $\mathrm{BD}^{\supset,\mathsf{F}}$ 的一个小变体加以介绍。第6节和第7节从 $\mathrm{BD}^{\supset,\mathsf{F}}$ 的视角，考察关系数据库，以及针对可能是不一致数据库和/或含空值的数据库进行的查询回答。第8节简要讨论本文对关系数据库中空值的看法的一些变体。第9节给出一些总结性评论。介绍 $\mathrm{BD}^{\supset,\mathsf{F}}$ 的部分章节与 [23] 的部分内容有所重叠。

2 $\mathrm{BD}^{\supset,\mathsf{F}}$ 的语言

本节介绍逻辑 $\mathrm{BD}^{\supset,\mathsf{F}}$ 的语言。首先介绍签名和字母表的概念，然后针对一个固定但任意的签名定义 $\mathrm{BD}^{\supset,\mathsf{F}}$ 的项和公式。此外，还介绍一些相关的记号约定，并对自由变量和代换作一些说明。在接下来的章节中，将针对一个固定但任意的签名介绍 $\mathrm{BD}^{\supset,\mathsf{F}}$ 的逻辑后承关系以及 $\mathrm{BD}^{\supset,\mathsf{F}}$ 的证明系统。

签名和字母表 假定已给定以下内容：(a) 一个由变量组成的可数无限集合 $\mathcal{V}\mathrm{ar}$；(b) 对每个 $n \in \mathbb{N}$，一个由元数为 n 的函数符号组成的可数无限集合 $\mathcal{F}\mathrm{unc}_n$；以及 (c) 对每个 $n \in \mathbb{N}$，一个由元数为 n 的谓词符号组成的可数无限集合 $\mathcal{P}\mathrm{red}_n$。此外，假定所有这些集合两两不相交，且均与集合 $\{=, \neg, \wedge, \vee, \supset, \forall, \exists\}$ 不相交。

来自 $\bigcup\{\mathcal{F}\mathrm{unc}_n \mid n \in \mathbb{N}\} \cup \bigcup\{\mathcal{P}\mathrm{red}_n \mid n \in \mathbb{N}\}$ 的符号称为非逻辑符号。元数为 0 的函数符号也称为常量符号，元数为 0 的谓词符号也称为命题符号。

签名 Σ 是 $\bigcup\{\mathcal{F}\mathrm{unc}_n \mid n \in \mathbb{N}\} \cup \bigcup\{\mathcal{P}\mathrm{red}_n \mid n \in \mathbb{N}\}$ 的子集。对于 Σ 为签名且 $n \in \mathbb{N}$ 的情形，我们分别将集合 $\Sigma \cap \mathcal{F}\mathrm{unc}_n$ 和 $\Sigma \cap \mathcal{P}\mathrm{red}_n$ 记作 $\mathcal{F}\mathrm{unc}_n(\Sigma)$ 和 $\mathcal{P}\mathrm{red}_n(\Sigma)$。

对于一个固定但任意的签名 Σ，将定义 $\mathrm{BD}^{\supset,\mathsf{F}}$ 的语言。该语言称为 $\mathrm{BD}^{\supset,\mathsf{F}}$ 在 Σ 上的语言，或简称 $\mathrm{BD}^{\supset,\mathsf{F}}(\Sigma)$ 的语言。相应的逻辑后承关系称为 $\mathrm{BD}^{\supset,\mathsf{F}}(\Sigma)$ 的逻辑后承关系，相应的证明系统称为 $\mathrm{BD}^{\supset,\mathsf{F}}(\Sigma)$ 的证明系统。

$\mathrm{BD}^{\supset,\mathsf{F}}(\Sigma)$ 的语言的字母表由以下符号组成：

– 来自 $\mathcal{V}\mathrm{ar}$ 的变量；– 来自 Σ 的非逻辑符号；– $\mathrm{BD}^{\supset,\mathsf{F}}$ 的逻辑符号，具体如下：

- 等号符号 $=$；
- 假联结词 F；
- 否定联结词 $\neg$；
- 合取联结词 $\wedge$；
- 析取联结词 $\vee$；
- 蕴涵联结词 $\supset$；
- 全称量词 $\forall$；
- 存在量词 $\exists$。

项和公式 $\mathrm{BD}^{\supset,\mathsf{F}}(\Sigma)$ 的语言由项和公式构成。它们按照下文给出的形成规则，由 $\mathrm{BD}^{\supset,\mathsf{F}}(\Sigma)$ 语言字母表中的符号构造而成。

$\mathrm{BD}^{\supset,\mathsf{F}}(\Sigma)$ 的所有项构成的集合，记作 $\mathcal{T}\mathrm{erm}(\Sigma)$，由以下形成规则归纳定义：

1. 若 $x \in \mathcal{V}\mathrm{ar}$，则 $x \in \mathcal{T}\mathrm{erm}(\Sigma)$；2. 若 $c \in \mathcal{F}\mathrm{unc}_0(\Sigma)$，则 $c \in \mathcal{T}\mathrm{erm}(\Sigma)$；3. 若 $f \in \mathcal{F}\mathrm{unc}_{n+1}(\Sigma)$ 且 $t_1, \ldots, t_{n+1} \in \mathcal{T}\mathrm{erm}(\Sigma)$，则 $f(t_1, \ldots, t_{n+1}) \in \mathcal{T}\mathrm{erm}(\Sigma)$。

$\mathrm{BD}^{\supset,\mathsf{F}}(\Sigma)$ 的所有闭项构成的集合，是 $\mathcal{T}\mathrm{erm}(\Sigma)$ 中仅通过应用形成规则 2 和 3 得到的子集。

$\mathrm{BD}^{\supset,\mathsf{F}}(\Sigma)$ 的所有公式构成的集合，记作 $\mathcal{F}\mathrm{orm}(\Sigma)$，由以下形成规则归纳定义：

1. 若 $p \in \mathcal{P}\mathrm{red}_0(\Sigma)$，则 $p \in \mathcal{F}\mathrm{orm}(\Sigma)$；2. 若 $P \in \mathcal{P}\mathrm{red}_{n+1}(\Sigma)$ 且 $t_1, \ldots, t_{n+1} \in \mathcal{T}\mathrm{erm}(\Sigma)$，则 $P(t_1, \ldots, t_{n+1}) \in \mathcal{F}\mathrm{orm}(\Sigma)$；3. 若 $t_1, t_2 \in \mathcal{T}\mathrm{erm}(\Sigma)$，则 $t_1 = t_2 \in \mathcal{F}\mathrm{orm}(\Sigma)$；4. $\mathsf{F} \in \mathcal{F}\mathrm{orm}(\Sigma)$；5. 若 $A \in \mathcal{F}\mathrm{orm}(\Sigma)$，则 $\neg A \in \mathcal{F}\mathrm{orm}(\Sigma)$；6. 若 $A_1, A_2 \in \mathcal{F}\mathrm{orm}(\Sigma)$，则 $A_1 \wedge A_2, A_1 \vee A_2, A_1 \supset A_2 \in \mathcal{F}\mathrm{orm}(\Sigma)$；7. 若 $x \in \mathcal{V}\mathrm{ar}$ 且 $A \in \mathcal{F}\mathrm{orm}(\Sigma)$，则 $\forall x \bullet A, \exists x \bullet A \in \mathcal{F}\mathrm{orm}(\Sigma)$。

$\mathsf{BD}^{\supset,\mathsf{F}}(\Sigma)$ 的所有原子公式构成的集合 $\mathcal{A}\mathrm{tom}(\Sigma)$，是通过仅应用形成规则 1–3 可构成的 $\mathcal{F}\mathrm{orm}(\Sigma)$ 的子集。$\mathsf{BD}^{\supset,\mathsf{F}}(\Sigma)$ 的所有文字构成的集合 $\mathcal{L}\mathrm{it}(\Sigma)$，是通过仅应用形成规则 1–3 和 5 可构成的 $\mathcal{F}\mathrm{orm}(\Sigma)$ 的子集。

我们写作 $e_1 \equiv e_2$，其中 e_1 和 e_2 是来自 $\mathcal{T}\mathrm{erm}(\Sigma)$ 的项，或来自$\mathcal{F}\mathrm{orm}(\Sigma)$ 的公式，用以表示 e_1 在语法上等于 e_2。

在接下来的章节中，我们将用 $\mathsf{CL}^{\supset,\mathsf{F}}$ 表示与 $\mathsf{BD}^{\supset,\mathsf{F}}$具有相同语言的经典逻辑版本。

记号约定：以下记号有时会在不加说明的情况下使用（带或不带装饰）：x 作为遍历 $\mathcal{V}\mathrm{ar}$ 中所有变量的元变量，t作为遍历 $\mathcal{T}\mathrm{erm}(\Sigma)$ 中所有项的元变量，A作为遍历 $\mathcal{F}\mathrm{orm}(\Sigma)$ 中所有公式的元变量，以及 Γ 作为遍历 $\mathcal{F}\mathrm{orm}(\Sigma)$ 中所有公式集的元变量。

上述形成规则所规定的项和公式的字符串表示可能导致语法歧义。为避免此类歧义，我们使用括号。通过为逻辑联结词 $\neg$、$\wedge$、$\vee$、$\supset$设定优先级，可以减少使用括号的需要。列举顺序体现了从最高优先级到最低优先级的排列。此外，量词的辖域尽可能向右延伸，$\forall x_1 \bullet \cdots \forall x_n \bullet A$和 $\exists x_1 \bullet \cdots \exists x_n \bullet A$ 通常分别写作 $\forall x_1, \ldots, x_n \bullet A$ 和 $\exists x_1, \ldots, x_n \bullet A$。

自由变量与代换 项或公式的自由变量，以及在项或公式中对变量进行代换，均按通常方式定义。

设 x是来自 $\mathcal{V}\mathrm{ar}$ 的一个变量，t是来自 $\mathcal{T}\mathrm{erm}(\Sigma)$ 的一个项，而 e是来自 $\mathcal{T}\mathrm{erm}(\Sigma)$ 的一个项，或是来自 $\mathcal{F}\mathrm{orm}(\Sigma)$ 的一个公式。于是，我们用 $[x := t]e$表示在 e 中将变量 x 的自由出现替换为项 t 所得的结果；同时通过重命名约束变量，避免自由变量在 t 中变为约束变量。

3 $\mathsf{BD}^{\supset,\mathsf{F}}(\Sigma)$ 中的真值与逻辑后承

本节将定义 $\mathsf{BD}^{\supset,\mathsf{F}}(\Sigma)$ 中公式的真值，以及 $\mathsf{BD}^{\supset,\mathsf{F}}(\Sigma)$ 的公式集上的逻辑后承关系。定义将借助 $\mathsf{BD}^{\supset,\mathsf{F}}(\Sigma)$ 的逻辑矩阵来完成。

首先，定义 $\mathsf{BD}^{\supset,\mathsf{F}}(\Sigma)$ 的逻辑矩阵。接下来，定义 $\mathsf{BD}^{\supset,\mathsf{F}}(\Sigma)$ 的公式的真值，以及 $\mathsf{BD}^{\supset,\mathsf{F}}(\Sigma)$ 的公式集上的逻辑后承关系。公式的真值是相对于以下内容定义的：(a) 一个由值集以及对每个非逻辑符号和等号符号的解释构成的结构；(b) 一个将该值集中的值赋给变量的赋值。在相关定义之前介绍结构和赋值。

矩阵 $\mathrm{BD}^{\supset,}{}_{\mathsf{F}}(\Sigma)$ 的逻辑符号（等号符号除外）的解释通过逻辑矩阵给出。

在此矩阵的定义中，t（真）、f（假）、b（既真又假）和 n（非真非假）被视为真值。此外，还使用集合 {t,f,b,n} 上的偏序 $\leq$，其中 f 是最小元，t 是最大元，而 b 和 n 不可比较。我们分别用 $\inf V$ 和 $\sup V$ 表示 V 关于 $\leq$ 的最大下界和最小上界，其中$V \subseteq$ {t,f,b,n}。

$\mathrm{BD}^{\supset,}{}_{\mathsf{F}}(\Sigma)$ 的矩阵是三元组 $\langle \mathcal{V}, \mathcal{D}, \mathcal{O} \rangle$，其中：

– $\mathcal{V} =$ {t,f,b,n}； – $\mathcal{D} =$ {t,b}； – $\mathcal{O}$ 由以下常元和函数构成：

$$
\begin{array}{lll}
\widetilde{\mathsf{F}} \in \mathcal{V} & \text{defined by} & \widetilde{\mathsf{F}} = \mathsf{f}\,, \\
\widetilde{\neg} : \mathcal{V} \to \mathcal{V} & \text{defined by} & \widetilde{\neg}(a) = \begin{cases} \mathsf{t} & \text{if } a = \mathsf{f} \\ \mathsf{f} & \text{if } a = \mathsf{t} \\ a & \text{otherwise}\,, \end{cases} \\
\widetilde{\wedge} : \mathcal{V} \times \mathcal{V} \to \mathcal{V} & \text{defined by} & \widetilde{\wedge}(a_1, a_2) = \inf\{a_1, a_2\}\,, \\
\widetilde{\vee} : \mathcal{V} \times \mathcal{V} \to \mathcal{V} & \text{defined by} & \widetilde{\vee}(a_1, a_2) = \sup\{a_1, a_2\}\,,
\end{array}
$$

$$
\begin{array}{lll}
\widetilde{\supset} : \mathcal{V} \times \mathcal{V} \to \mathcal{V} & \text{defined by} & \widetilde{\supset}(a_1, a_2) = \begin{cases} \mathsf{t} & \text{if } a_1 \notin \{\mathsf{t}, \mathsf{b}\} \\ a_2 & \text{otherwise}\,, \end{cases} \\
\widetilde{\forall} : \mathcal{P}(\mathcal{V}) \setminus \{\emptyset\} \to \mathcal{V} & \text{defined by} & \widetilde{\forall}(V) = \inf V\,, \\
\widetilde{\exists} : \mathcal{P}(\mathcal{V}) \setminus \{\emptyset\} \to \mathcal{V} & \text{defined by} & \widetilde{\exists}(V) = \sup V\,,
\end{array}
$$

[1]

其中，a、a_1 和 a_2 遍历 $\mathcal{V}$ 中的所有真值，V 遍历 $\mathcal{V}$ 的所有非空子集。

$\mathcal{V}$ 是 $\mathrm{BD}^{\supset,}{}_{\mathsf{F}}(\Sigma)$ 的真值集合，$\mathcal{D}$ 是 $\mathrm{BD}^{\supset,}{}_{\mathsf{F}}(\Sigma)$ 的指定真值集合，而 $\widetilde{\mathsf{F}}$、$\widetilde{\neg}$、$\widetilde{\wedge}$、$\widetilde{\vee}$、$\widetilde{\supset}$、$\widetilde{\forall}$ 和 $\widetilde{\exists}$ 分别是逻辑符号 F、$\neg$、$\wedge$、$\vee$、$\supset$、$\forall$ 和 $\exists$ 的解释所对应的真值函数。指定真值的理念是：如果某公式相对于所有结构以及这些结构中的所有赋值（两者均将在下文定义）的真值都是指定真值，那么该公式就是有效的。

$\mathrm{BD}^{\supset,}{}_{\mathsf{F}}(\Sigma)$ 的非指定真值集合，记作 $\overline{\mathcal{D}}$，是集合 $\mathcal{V} \setminus \mathcal{D}$。

结构 $\mathrm{BD}^{\supset,}{}_{\mathsf{F}}(\Sigma)$ 的非逻辑符号和等号符号的解释通过结构给出。

$\mathrm{BD}^{\supset,}{}_{\mathsf{F}}(\Sigma)$ 的一个结构 A 是一对 $\langle \mathcal{U}^{\mathbf{A}}, \mathcal{I}^{\mathbf{A}} \rangle$，其中：

– $\mathcal{U}^{\mathbf{A}}$ 是一个集合，称为 A 的论域，满足 $\mathcal{U}^{\mathbf{A}} \neq \emptyset$ 和 $\mathcal{U}^{\mathbf{A}} \cap \mathcal{V} = \emptyset$； – $\mathcal{I}^{\mathbf{A}}$ 由以下部分组成：

[1] 我们用 $\mathcal{P}(S)$ 表示 S 的幂集，其中 S 是一个集合。

- 一个元素 $c_{\mathbf{A}} \in \mathcal{U}_{\mathbf{A}}$ 对每个 $c \in \mathcal{F}unc_{0}(\Sigma)$;$n+1$

次

- 一个函数 $f_{\mathbf{A}}$： $\mathcal{U}_{\mathbf{A}} \times \cdots \times \mathcal{U}_{\mathbf{A}} \to \mathcal{U}_{\mathbf{A}}$ 对每个 $f \in \mathcal{F}\mathrm{unc}_{n+1}(\Sigma)$ 和 $n \in \mathbb{N}$;

- 一个元素 $p_{\mathbf{A}} \in \mathcal{V}$ 对每个 $p \in \mathcal{P}$红$_{0}(\Sigma)$;$n+1$

次

- 一个函数 $P_{\mathbf{A}}$： $\mathcal{U}_{\mathbf{A}} \times \cdots \times \mathcal{U}_{\mathbf{A}} \to \mathcal{V}$ 对每个 $P \in \mathcal{P}$红$_{n+1}(\Sigma)$ 和 $n \in \mathbb{N}$;

- 一个函数 $=_{\mathbf{A}}$： $\mathcal{U}_{\mathbf{A}} \times \mathcal{U}_{\mathbf{A}} \to \mathcal{V}$ 其中，对所有 $d_1, d_2 \in \mathcal{U}_{\mathbf{A}}$， $=_{\mathbf{A}}(d_1, d_2) \in \mathcal{D}$ 当且仅当 $d_1 = d_2$。

我们不写 $w^{\mathbf{A}}$，而写 w，前提是上下文已清楚表明所指的是符号 w 在结构 A 中的解释。

赋值 BD$^{\supset}$,$_{\mathrm{F}}(\Sigma)$ 中变量的解释通过赋值给出。

设 A 是 BD$^{\supset}$, $_{\mathrm{F}}(\Sigma)$ 的一个结构。那么，A 中的一个赋值是一个函数α： $\mathcal{V}$ar $\to \mathcal{U}^{\mathbf{A}}$。对于 A 中的每个赋值 α 、变量 $x \in \mathcal{V}$ar 和元素$d \in \mathcal{U}^{\mathbf{A}}$，我们用 $\alpha($ $x \to d)$ 表示 A 中满足以下条件的赋值 α'： $\alpha'(x) = d$ 且$\alpha'(y) = \alpha(y)$，如果 $y \not\equiv x$。

赋值和模型 设 A 是 BD$^{\supset}$, $_{\mathrm{F}}(\Sigma)$ 的一个结构，并设 α 是 A 中的一个赋值。那么，在赋值 α下，结构 A 中 $\mathcal{T}$erm(Σ) 的赋值是函数 tval$^{\mathbf{A}}_{\alpha}$: $\mathcal{T}$erm$(\Sigma) \to \mathcal{U}^{\mathbf{A}}$，它将每个项 t映射到 $\mathcal{U}^{\mathbf{A}}$ 中的一个元素，该元素是在赋值 α下 t 在 A 中的取值。类似地，在赋值 α下，结构 A 中 $\mathcal{F}$orm(Σ) 的赋值是函数 fval $_{\mathbf{A}\alpha}$: $\mathcal{F}$ orm$(\Sigma) \to \mathcal{V}$，它将每个公式 A映射到 $\mathcal{V}$中的一个元素，该元素是在赋值 α下 A 在 A 中的真值。我们分别用 $[\![t]\!]^{\mathbf{A}}_{\alpha}$ 和 $[\![A]\!]^{\mathbf{A}}_{\alpha}$ 表示 tval$^{\mathbf{A}}_{\alpha}(t)$ 和 fval$^{\mathbf{A}}_{\alpha}(A)$。

这些赋值函数在表1中以归纳方式定义。在此表中，x 是一个遍历 $\mathcal{V}$ar中所有变量的元变量， c 是一个遍历 $\mathcal{F}\mathrm{unc}_0(\Sigma)$中所有函数符号的元变量， f 是一个遍历 $\mathcal{F}\mathrm{unc}_{n+1}(\Sigma)$中所有函数符号的元变量， p 是一个遍历 $\mathcal{P}\mathrm{red}_0(\Sigma)$中所有谓词符号的元变量， P是一个遍历 $\mathcal{P}\mathrm{red}_{n+1}(\Sigma)$中所有谓词符号的元变量， t_1、.....、 t_{n+1}是遍历 $\mathcal{T}$erm(Σ)中所有项的元变量，而 A、 A_1和 A_2 是遍历 $\mathcal{F}$ orm(Σ)中所有公式的元变量。

下面的定理给出了一个可判定性结果，涉及有限论域结构中公式的赋值。

命题 1. 设 A 是 BD$^{\supset}$,$_{\mathrm{F}}(\Sigma)$ 的一个结构，其中 $\mathcal{U}^{\mathbf{A}}$ 是有限的，且

α $A \in$ 是 A 中的一个赋值。

$\mathcal{F}$ Σ

那么，对于公式 orm( )， $[A]^{\mathbf{A}}_{\alpha} \in \mathcal{D}$ 是否成立是可判定的。

表1. $\mathsf{BD}^{\supset}_{,\mathsf{F}}(\Sigma)$ 中项和公式的赋值

$$
\begin{array}{rl}
[\![x]\!]^{\mathbf{A}}_{\alpha} & = \alpha(x)\,, \\
[\![c]\!]^{\mathbf{A}}_{\alpha} & = c^{\mathbf{A}}\,, \\
[\![f(t_1,\ldots,t_{n+1})]\!]^{\mathbf{A}}_{\alpha} & = f^{\mathbf{A}}([\![t_1]\!]^{\mathbf{A}}_{\alpha},\ldots,[\![t_{n+1}]\!]^{\mathbf{A}}_{\alpha}) \\
[\![p]\!]^{\mathbf{A}}_{\alpha} & = p^{\mathbf{A}}\,, \\
[\![P(t_1,\ldots,t_{n+1})]\!]^{\mathbf{A}}_{\alpha} & = P^{\mathbf{A}}([\![t_1]\!]^{\mathbf{A}}_{\alpha},\ldots,[\![t_{n+1}]\!]^{\mathbf{A}}_{\alpha})\,, \\
[\![t_1 = t_2]\!]^{\mathbf{A}}_{\alpha} & = {=}^{\mathbf{A}}([\![t_1]\!]^{\mathbf{A}}_{\alpha},[\![t_2]\!]^{\mathbf{A}}_{\alpha})\,, \\
[\![\mathsf{F}]\!]^{\mathbf{A}}_{\alpha} & = \widetilde{\mathsf{F}}\,, \\
[\![\neg A]\!]^{\mathbf{A}}_{\alpha} & = \widetilde{\neg}([\![A]\!]^{\mathbf{A}}_{\alpha})\,, \\
[\![A_1 \wedge A_2]\!]^{\mathbf{A}}_{\alpha} & = \widetilde{\wedge}([\![A_1]\!]^{\mathbf{A}}_{\alpha},[\![A_2]\!]^{\mathbf{A}}_{\alpha})\,, \\
[\![A_1 \vee A_2]\!]^{\mathbf{A}}_{\alpha} & = \widetilde{\vee}([\![A_1]\!]^{\mathbf{A}}_{\alpha},[\![A_2]\!]^{\mathbf{A}}_{\alpha})\,, \\
[\![A_1 \supset A_2]\!]^{\mathbf{A}}_{\alpha} & = \widetilde{\supset}([\![A_1]\!]^{\mathbf{A}}_{\alpha},[\![A_2]\!]^{\mathbf{A}}_{\alpha})\,, \\
[\![\forall x \bullet A]\!]^{\mathbf{A}}_{\alpha} & = \widetilde{\forall}(\{[\![A]\!]^{\mathbf{A}}_{\alpha(x\to d)} \mid d \in \mathcal{U}^{\mathbf{A}}\})\,, \\
[\![\exists x \bullet A]\!]^{\mathbf{A}}_{\alpha} & = \widetilde{\exists}(\{[\![A]\!]^{\mathbf{A}}_{\alpha(x\to d)} \mid d \in \mathcal{U}^{\mathbf{A}}\})\,,
\end{array}
$$

证明. 根据 A 的结构进行归纳即可轻松证明这一点。 □

下文将用赋值来定义 $\mathsf{BD}^{\supset}_{,\mathsf{F}}(\Sigma)$ 中公式集的模型这一概念。

设 Γ 是 $\mathcal{F}\mathsf{orm}(\Sigma)$ 中的一个公式集，A 是 $\mathsf{BD}^{\supset}_{,\mathsf{F}}(\Sigma)$ 的一个结构。那么，A 是 Γ 的模型，当且仅当对于 A 中的所有赋值 α，对于所有 $A \in \Gamma$，$[\![A]\!]^{\mathbf{A}}_{\alpha} \in \mathcal{D}$。

逻辑后承 给定 $\mathsf{BD}^{\supset}_{,\mathsf{F}}(\Sigma)$ 中项和公式的赋值，就很容易明确 $\mathsf{BD}^{\supset}_{,\mathsf{F}}(\Sigma)$ 中一个公式集何时是另一个公式集的逻辑后承。

设 Γ 和 Δ 是 $\mathcal{F}\mathsf{orm}(\Sigma)$ 中的公式集。那么，Δ 是 Γ 的逻辑后承，记作 $\Gamma \vDash \Delta$，当且仅当对于 $\mathsf{BD}^{\supset}_{,\mathsf{F}}(\Sigma)$ 的所有结构 A，以及 A 中的所有赋值 α，若 $[\![A]\!]^{\mathbf{A}}_{\alpha} \in \mathcal{D}$ 对所有 $A \in \Gamma$ 都成立，则 $[\![A']\!]^{\mathbf{A}}_{\alpha} \in \mathcal{D}$ 对某个 $A' \in \Delta$ 成立。我们用 $\Gamma \nvDash \Delta$ 表示 $\Gamma \vDash \Delta$ 并非如此。

下文提到的关于 $\mathsf{BD}^{\supset}_{,\mathsf{F}}(\Sigma)$ 逻辑后承关系的两个性质在 [23] 中得到了证明。

$\mathsf{BD} \vDash^{\supset}_{,\mathsf{F}}(\Sigma)$ 的逻辑后承关系满足

$$\Gamma \vDash \Delta, A_1 \supset A_2 \text{ iff } A_1, \Gamma \vDash \Delta, A_2$$

对于所有 $\Gamma, \Delta \subseteq \mathcal{F}\mathsf{orm}(\Sigma)$ 和 $A_1, A_2 \in \mathcal{F}\mathsf{orm}(\Sigma)$，此外还存在一种逻辑 $\mathcal{L}$，其语言与 $\mathsf{BD}^{\supset}_{,\mathsf{F}}(\Sigma)$ 相同，并且存在一个逻辑后承关系 $\vDash'$，使得：

– $\models \subseteq \models'$;
– $\mathcal{L}$ 的矩阵 $\langle \mathcal{V}', \mathcal{D}', \mathcal{O}' \rangle$ 满足：$\mathcal{V}' = \{\mathsf{t}, \mathsf{f}\}$, $\mathcal{D}' = \{\mathsf{t}\}$，而 $\neg$ 在 $\mathcal{O}'$ 中的解释如下：

$$\widetilde{\neg}'(a) = \begin{cases} \mathsf{t} & \text{if } a = \mathsf{f} \\ \mathsf{f} & \text{if } a = \mathsf{t} \end{cases}$$

对所有 $a \in \mathcal{V}'$均成立。

这意味着，按照 [3]的意义，$\mathsf{BD}^{\supset,\mathsf{F}}(\Sigma)$与经典逻辑是 $\neg$-协调的。

存在 $A, A' \in \mathcal{F}\mathsf{orm}(\Sigma)$，使得 $A, \neg A \nvdash A'$。由于$\mathsf{BD}^{\supset,\mathsf{F}}(\Sigma)$也与经典逻辑 $\neg$-协调，这意味着，按照 [3]的意义，$\mathsf{BD}^{\supset,\mathsf{F}}(\Sigma)$是次协调的。

存在一个 $\Gamma \subseteq \mathcal{F}\mathsf{orm}(\Sigma)$以及 $A, A' \in \mathcal{F}\mathsf{orm}(\Sigma)$，使得 $\Gamma, A \vDash A'$和$\Gamma, \neg A \vDash A'$，但 $\Gamma \nvDash A'$。由于$\mathsf{BD}^{\supset,\mathsf{F}}(\Sigma)$也与经典逻辑 $\neg$-协调，这意味着，按照 [3]的意义，$\mathsf{BD}^{\supset,\mathsf{F}}(\Sigma)$是次完备的。

缩写 下文将使用以下缩写：

$$\begin{array}{rll} t_1 \neq t_2 & \text{stands for} & \neg(t_1 = t_2), \\ \mathsf{T} & \text{stands for} & \neg\mathsf{F}, \\ A_1 \to A_2 & \text{stands for} & (A_1 \supset A_2) \wedge (\neg A_2 \supset \neg A_1), \\ \Delta A & \text{stands for} & \neg(A \supset \mathsf{F}) \qquad \text{(designatedness)}, \\ \circ A & \text{stands for} & \neg(\Delta(A \wedge \neg A)) \qquad \text{(consistency)}, \\ \star A & \text{stands for} & \Delta(A \vee \neg A) \qquad \text{(determinacy)}. \end{array}$$

根据相关定义可得：

$$\begin{aligned} [\![\Delta A]\!]^{\mathbf{A}}_{\alpha} &= \begin{cases} \mathsf{t} & \text{if } [\![A]\!]^{\mathbf{A}}_{\alpha} \in \{\mathsf{t}, \mathsf{b}\} \\ \mathsf{f} & \text{otherwise}, \end{cases} \\ [\![\circ A]\!]^{\mathbf{A}}_{\alpha} &= \begin{cases} \mathsf{t} & \text{if } [\![A]\!]^{\mathbf{A}}_{\alpha} \in \{\mathsf{t}, \mathsf{f}, \mathsf{n}\} \\ \mathsf{f} & \text{otherwise}, \end{cases} \\ [\![\star A]\!]^{\mathbf{A}}_{\alpha} &= \begin{cases} \mathsf{t} & \text{if } [\![A]\!]^{\mathbf{A}}_{\alpha} \in \{\mathsf{t}, \mathsf{f}, \mathsf{b}\} \\ \mathsf{f} & \text{otherwise}. \end{cases} \end{aligned}$$

这意味着，缩写 ΔA、$\circ A$ 和 $\star A$ 对应于所研究的 Belnap-Dunn逻辑扩展中的公式，其联结词包括 Δ、$\circ$ 或 $\star$。例如，在一种称为 $\mathsf{BD}\Delta$ [28] 的 Belnap-Dunn逻辑扩展中就出现了联结词 Δ。例如，文献 [9] 在 Belnap-Dunn逻辑的框架下研究了联结词 $\circ$ 和 $\star$。联结词 $\circ$也出现在 Priest悖论逻辑的一阶版本 [26] 的扩展中，该扩展称为 LP° [25]。联结词 $\star$ 是 Kleene强三值逻辑 [16, （第 64] 节）框架下 $\circ$ 的对应项。

此外，请注意：

$$[\![\circ A \wedge \star A]\!]^{\mathbf{A}}_{\alpha} = \begin{cases} \mathsf{t} & \text{if } [\![A]\!]^{\mathbf{A}}_{\alpha} \in \{\mathsf{t}, \mathsf{f}\} \\ \mathsf{f} & \text{otherwise}. \end{cases}$$

4 $\mathrm{BD}^{\supset,\mathsf{F}}(\Sigma)$ 的证明系统

本节将介绍 $\mathrm{BD}^{\supset,\mathsf{F}}(\Sigma)$ 的序列演算证明系统。这意味着推理规则的前提和结论都是序列。首先介绍序列的概念，然后介绍 $\mathrm{BD}^{\supset,\mathsf{F}}(\Sigma)$ 证明系统的推理规则。之后，将介绍从一组序列推导出一个序列的推导概念，以及序列的证明概念。本文还将介绍 $\mathrm{BD}^{\supset,\mathsf{F}}(\Sigma)$ 证明系统的扩展，这些扩展可用作密切相关逻辑的证明系统。

序列 在 $\mathrm{BD}^{\supset,\mathsf{F}}(\Sigma)$ 的序列演算证明系统中，序列是形如 $\Gamma \Rightarrow \Delta$ 的表达式，其中 Γ 和 Δ 是来自 $\mathcal{F}\mathrm{orm}(\Sigma)$ 的有限公式集。对于序列两侧的 $\Gamma \cup \Gamma'$ 和 A，若 A 是来自 $\mathcal{F}\mathrm{orm}(\Sigma)$ 的公式，我们将其写作 Γ, Γ'。此外，我们用 $\Rightarrow \Delta$ 代替 $\emptyset \Rightarrow \Delta$。

序列 $\Gamma \Rightarrow \Delta$ 表示第3节中定义的逻辑后承关系在 Γ 与 Δ 之间成立。如果序列 $\Gamma \Rightarrow \Delta$ 可以通过下文给出的推理规则证明，那么该逻辑后承关系就在 Γ 与 Δ之间成立。

推理规则 $\mathrm{BD}^{\supset,\mathsf{F}}(\Sigma)$ 的序列演算证明系统由表 2 中给出的推理规则构成。在此表中，x 和 y 是取遍 $\mathcal{V}\mathrm{ar}$ 中所有变量的元变量，t、t_1 和 t_2 是取遍 $\mathcal{T}\mathrm{erm}(\Sigma)$ 中所有项的元变量，A、A_1 和 A_2 是取遍 $\mathcal{F}\mathrm{orm}(\Sigma)$ 中所有公式的元变量，而 Γ 和 Δ 是取遍 $\mathcal{F}\mathrm{orm}(\Sigma)$ 中所有有限公式集的元变量。来自 $\mathcal{F}\mathrm{orm}(\Sigma)$ 的公式 A 是文字，当且仅当$A \in \mathcal{L}\mathrm{it}(\Sigma)$。

推导与证明 在 $\mathrm{BD}^{\supset,\mathsf{F}}(\Sigma)$ 的序列演算证明系统中，若有限个序列 $\mathcal{H}$ 的一个有限序列$\langle s_1, \ldots, s_n \rangle$ 满足 s_n 等于 $\Gamma \Rightarrow \Delta$，且对每个 $i \in \{1, \ldots, n\}$，下列条件之一成立，则称该序列为从有限个序列 $\mathcal{H}$ 推导序列 $\Gamma \Rightarrow \Delta$ 的推导：

– $s_i \in \mathcal{H}$；– s_i 是 $\mathrm{BD}^{\supset,\mathsf{F}}(\Sigma)$ 证明系统中某条推理规则的一个实例的结论，且其前提属于 $s_1, \ldots, s_{i-1}$。

序列 $\Gamma \Rightarrow \Delta$ 的证明是从空序列集推导 $\Gamma \Rightarrow \Delta$ 。如果存在序列 $\Gamma \Rightarrow \Delta$ 的证明，则称序列 $\Gamma \Rightarrow \Delta$ 可证。

设 Γ 和 Δ 是 $\mathcal{F}\mathrm{orm}(\Sigma)$ 中的公式集。若存在有限集 $\Gamma' \subseteq \Gamma$ 和 $\Delta' \subseteq \Delta$，使得序列 $\Gamma' \Rightarrow \Delta'$ 可证，则称 Δ 可从 Γ 推导，记作 $\Gamma \vdash \Delta$。

$\mathrm{BD}^{\supset,\mathsf{F}}(\Sigma)$ 的序列演算证明系统相对于第3节中定义的逻辑后承关系 $\models$ 是可靠且完备的。

表 2. $\mathrm{BD}^{\supset,\,\mathsf{F}}$ 的序列演算证明系统

$$\boxed{\text{Id}}\ \frac{}{A, \Gamma \Rightarrow \Delta, A}\ *$$

$$\boxed{\text{Cut}}\ \frac{\Gamma \Rightarrow \Delta, A \quad A, \Gamma' \Rightarrow \Delta'}{\Gamma', \Gamma \Rightarrow \Delta, \Delta'}$$

$$\boxed{\mathsf{F}\text{-L}}\ \frac{}{\mathsf{F}, \Gamma \Rightarrow \Delta}$$

$$\boxed{\wedge\text{-L}}\ \frac{A_1, A_2, \Gamma \Rightarrow \Delta}{A_1 \wedge A_2, \Gamma \Rightarrow \Delta} \qquad \boxed{\wedge\text{-R}}\ \frac{\Gamma \Rightarrow \Delta, \quad A_1 \quad \Gamma \Rightarrow \Delta, \quad A_2}{\Gamma \Rightarrow \Delta, \quad A_1 \wedge A_2}$$

$$\boxed{\vee\text{-L}}\ \frac{A_1, \Gamma \Rightarrow \Delta \quad A_2, \Gamma \Rightarrow \Delta}{A_1 \vee A_2, \Gamma \Rightarrow \Delta} \qquad \boxed{\vee\text{-R}}\ \frac{\Gamma \Rightarrow \Delta, \quad A_1, \quad A_2}{\Gamma \Rightarrow \Delta, \quad A_1 \vee A_2}$$

$$\boxed{\supset\text{-L}}\ \frac{\Gamma \Rightarrow \Delta, A_1 \quad A_2, \Gamma \Rightarrow \Delta}{A_1 \supset A_2, \Gamma \Rightarrow \Delta} \qquad \boxed{\supset\text{-R}}\ \frac{A_1, \quad \Gamma \Rightarrow \Delta, \quad A_2}{\Gamma \Rightarrow \Delta, A_1 \supset A_2}$$

$$\boxed{\forall\text{-L}}\ \frac{[x := t]A, \Gamma \Rightarrow \Delta}{\forall x \bullet A, \Gamma \Rightarrow \Delta} \qquad \boxed{\forall\text{-R}}\ \frac{\Gamma \Rightarrow \Delta, [x := y]A}{\Gamma \Rightarrow \Delta, \forall x \bullet A}\ \dagger$$

$$\boxed{\exists\text{-L}}\ \frac{[x := y]A, \Gamma \Rightarrow \Delta}{\exists x \bullet A, \Gamma \Rightarrow \Delta}\ \dagger \qquad \boxed{\exists\text{-R}}\ \frac{\Gamma \Rightarrow \Delta, [x := t]A}{\Gamma \Rightarrow \Delta, \exists x \bullet A}$$

$$\boxed{\neg\mathsf{F}\text{-R}}\ \frac{}{\Gamma \Rightarrow \Delta, \neg\mathsf{F}}$$

$$\boxed{\neg\neg\text{-L}}\ \frac{A, \Gamma \Rightarrow \Delta}{\neg\neg A, \Gamma \Rightarrow \Delta} \qquad \boxed{\neg\neg\text{-R}}\ \frac{\Gamma \Rightarrow \Delta, A}{\Gamma \Rightarrow \Delta, \neg\neg A}$$

$$\boxed{\neg\wedge\text{-L}}\ \frac{\neg A_1, \Gamma \Rightarrow \Delta \quad \neg A_2, \Gamma \Rightarrow \Delta}{\neg(A_1 \wedge A_2)\ \ \Gamma \Rightarrow \Delta} \qquad \boxed{\neg\wedge\text{-R}}\ \frac{\Gamma \Rightarrow \Delta, \neg A_1, \neg A_2}{\Gamma \Rightarrow \Delta, \neg(A_1 \wedge A_2)}$$

$$\boxed{\neg\vee\text{-L}}\ \frac{\neg A_1, \neg A_2, \Gamma \Rightarrow \Delta}{\neg(A_1 \vee A_2)\ \ \Gamma \Rightarrow \Delta} \qquad \boxed{\neg\vee\text{-R}}\ \frac{\Gamma \Rightarrow \Delta, \neg A_1 \quad \Gamma \Rightarrow \Delta, \neg A_2}{\Gamma \Rightarrow \Delta, \neg(A_1 \vee A_2)}$$

$$\boxed{\neg\supset\text{-L}}\ \frac{A_1, \neg A_2, \Gamma \Rightarrow \Delta}{\neg(A_1 \supset A_2)\ \ \Gamma \Rightarrow \Delta} \qquad \boxed{\neg\supset\text{-R}}\ \frac{\Gamma \Rightarrow \Delta, A_1 \quad \Gamma \Rightarrow \Delta, \neg A_2}{\Gamma \Rightarrow \Delta, \neg(A_1 \supset A_2)}$$

$$\boxed{\neg\forall\text{-L}}\ \frac{\neg[x := y]A, \Gamma \Rightarrow \Delta}{\neg\forall x \bullet A, \Gamma \Rightarrow \Delta}\ \dagger \qquad \boxed{\neg\forall\text{-R}}\ \frac{\Gamma \Rightarrow \Delta, \neg[x := t]A}{\Gamma \Rightarrow \Delta, \neg\forall x \bullet A}$$

$$\boxed{\neg\exists\text{-L}}\ \frac{\neg[x := t]A, \Gamma \Rightarrow \Delta}{\neg\exists x \bullet A, \Gamma \Rightarrow \Delta} \qquad \boxed{\neg\exists\text{-R}}\ \frac{\Gamma \Rightarrow \Delta, \neg[x := y]A}{\Gamma \Rightarrow \Delta, \neg\exists x \bullet A}\ \dagger$$

$$\boxed{=\text{自反性}}\ \frac{t = t, \quad \Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta} \qquad \boxed{=\text{-替换}}\ \frac{[x := t_1]A, \Gamma \Rightarrow \Delta}{t_1 = t_2, [x := t_2]A, \Gamma \Rightarrow \Delta}\ *$$

$*$ 限制： A是一个文字。

† 限制：除非 $x \equiv y$，否则 y 在 Γ中不是自由的， y 在 Δ中不是自由的， y 在 A 中不是自由的。

定理 1。设 Γ 和 Δ 是来自 $\mathcal{F}\mathrm{orm}(\Sigma)$ 的公式集。那么 $\Gamma \vdash \Delta$ 当且仅当$\Gamma \vDash \Delta$。

证明。见 [23]的附录 A。 □

证明系统的扩展 $\mathbf{CL}^{\supset,\mathsf{F}}$ 和 $\mathbf{BD}^{\supset,\mathsf{F}}$ 的语言相同。通过在 $\mathbf{BD}^{\supset,\mathsf{F}}$的序列演算证明系统中加入以下两条推理规则，可以得到 $\mathbf{CL}^{\supset,\mathsf{F}}$ 的可靠且完备的序列演算证明系统：

$$\boxed{\neg\text{-L}}\ \frac{\Gamma \Rightarrow \Delta, A}{\neg A, \Gamma \Rightarrow \Delta} \qquad\qquad \boxed{\neg\text{-R}}\ \frac{A, \Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, \neg A}$$

如果只在 $\mathbf{BD}^{\supset,\mathsf{F}}$的序列演算证明系统中加入推理规则 $\neg$-R，那么我们就得到一个可靠且完备的次协调（但非次完备）逻辑 $\mathbf{LPQ}^{\supset,\mathsf{F}}$ 的证明系统，该逻辑见 [22]。如果只在 $\mathbf{BD}^{\supset,\mathsf{F}}$ 的序列演算证明系统中加入推理规则 $\neg$-L，那么我们就得到一个可靠且完备的次完备（但非次协调）命题逻辑 $\mathbf{K3}^{\supset,\mathsf{F}}$ 的显然的一阶版本，该逻辑见 [21]。

5 含有不定值的项覆盖

本节将介绍 $\mathbf{BD}^{\supset,\mathsf{F}}(\Sigma)$ 的一个小变体，称为 $\mathbf{BD}^{\supset,\mathsf{F}}_{\bot}(\Sigma)$，它能够处理含有不定值的项。从语义上说，这是通过将结构的论域限制为包含一个特殊虚设值 $\bot$ 的集合，并将该值赋给含有不定值的项来实现的。因此，$\bot$ 被视为一个真正的值。它与其他值的区别仅在于对结构的等同关系施加的条件（见下文）。这些条件符合这样一种直觉：含有不定值的项具有未知值，或者没有值。

$\mathbf{BD}^{\supset,\mathsf{F}}(\Sigma)$ 的语言与 $\mathbf{BD}^{\supset,\mathsf{F}}_{\bot}(\Sigma)$ 的语言相同。$\mathbf{BD}^{\supset,\mathsf{F}}_{\bot}(\Sigma)$ 的逻辑后承关系的定义方式与 $\mathbf{BD}^{\supset,\mathsf{F}}(\Sigma)$ 相同，但采用 $\mathbf{BD}^{\supset,\mathsf{F}}_{\bot}(\Sigma)$ 的结构来定义。这些结构与 $\mathbf{BD}^{\supset,\mathsf{F}}(\Sigma)$ 的结构略有不同。

结构 $\mathbf{BD}^{\supset,\mathsf{F}}_{\bot}(\Sigma)$ 的一个结构 $\mathbf{A}$ 是一对 $\langle \mathcal{U}^{\mathbf{A}}, \mathcal{I}^{\mathbf{A}} \rangle$，其中：

– $\mathcal{U}^{\mathbf{A}}$ 是一个集合，称为 $\mathbf{A}$ 的论域，其中 $\bot \in \mathcal{U}^{\mathbf{A}}$，$\mathcal{U}^{\mathbf{A}} \setminus \{\bot\} \neq \emptyset$，且 $\mathcal{U}^{\mathbf{A}} \cap \mathcal{V} = \emptyset$；– $\mathcal{I}^{\mathbf{A}}$ 由以下内容组成：

- 一个元素 $c^{\mathbf{A}} \in \mathcal{U}_{\mathbf{A}}$ 对于每个 $c \in \mathcal{F}unc^{0}(\Sigma)$；
- 一个函数 $f_{\mathbf{A}}\colon \overbrace{\mathcal{U}_{\mathbf{A}} \times \cdots \times \mathcal{U}_{\mathbf{A}}}^{n+1\text{ 次}} \to \mathcal{U}$ 对于每个 $f \in \mathcal{F}\mathrm{unc}^{n+1}(\Sigma)$ 和 $n \in \mathbb{N}$；

- 一个元素 $p_{\mathbf{A}} \in \mathcal{V}$ 对于每个 $p \in \mathcal{P}$红色$_0(\Sigma)$;
- 一个函数 $P_{\mathbf{A}}\colon \overbrace{\mathcal{U}_{\mathbf{A}} \times \cdots \times \mathcal{U}_{\mathbf{A}}}^{n+1\text{ 次}} \to \mathcal{V}$对于每个 $P \in \mathcal{P}$红色$_{n+1}(\Sigma)$ 和$n \in \mathbb{N}$;
- 一个函数 $=_{\mathbf{A}}\colon \mathcal{U}_{\mathbf{A}} \times \mathcal{U}_{\mathbf{A}} \to \mathcal{V}$ 其中，对于所有 $d_1, d_2 \in \mathcal{U}_{\mathbf{A}}$: $=_{\mathbf{A}}(d_1, d_2) \in \mathcal{D}$ 当且仅当 $d_1, d_2 \in \mathcal{U}_{\mathbf{A}} \setminus \{\bot\}$ 且 $d_1 = d_2$; $=_{\mathbf{A}}(d_1, d_2) =$ 非当且仅当 $d_1 = \bot$ 或 $d_2 = \bot$。

根据对 $=^{\mathbf{A}}$ BD 结构 A 的定义中施加的第一个条件$^{\supset\,\mathsf{F}}_{\bot,}(\Sigma)$，$[\![t = t]\!]^{\mathbf{A}}_{\alpha} \in \mathcal{D}$ 当且仅当 $[\![t]\!]^{\mathbf{A}}_{\alpha} \neq \bot$。换言之，公式 $t = t$ 的真值是指定真值，当且仅当项 t 的值是确定的。

根据对 $=^{\mathbf{A}}$ BD 结构 A 的定义中施加的第二个条件$^{\supset\,\mathsf{F}}_{\bot,}(\Sigma)$，$[\![t_1 = t_2]\!]^{\mathbf{A}}_{\alpha} =$ 非当且仅当 $[\![t_1]\!]^{\mathbf{A}}_{\alpha} = \bot$ 或 $[\![t_2]\!]^{\mathbf{A}}_{\alpha} = \bot$。换言之，方程 $t_1 = t_2$ 的真值既非真也非假（n），当且仅当t_1 或 t_2 或二者的值是不确定的。对于来自 Σ的函数符号和谓词符号的解释，没有施加类似条件。然而，由于对 $=^{\mathbf{A}}$施加了第一个条件，相关条件可以用 BD$^{\supset\,\mathsf{F}}_{\bot,}(\Sigma)$ 表示。

逻辑后承 设 Γ 和 Δ 是来自 $\mathcal{F}$orm(Σ) 的公式集。那么，Δ是 Γ 的逻辑后承，记作 $\Gamma \models_{\bot} \Delta$，当且仅当对于 BD$^{\supset\,\mathsf{F}}_{\bot,}(\Sigma)$ 的所有结构 A 以及 A 中的所有赋值 α，若对于所有 $A \in \Gamma$都有 $[\![A]\!]^{\mathbf{A}}_{\alpha} \in \mathcal{D}$，则对于某个 $A' \in \Delta$有 $[\![A']\!]^{\mathbf{A}}_{\alpha} \in \mathcal{D}$。

根据 BD$^{\supset\,\mathsf{F}}_{\bot,}(\Sigma)$ 的结构 A 定义中关于 $=^{\mathbf{A}}$ 的条件，对于所有 t_1, $t_2 \in \mathcal{T}$erm(Σ)，我们有:

$$t_1 = t_1 \wedge t_2 = t_2 \models_{\bot} t_1 = t_2 \vee \neg(t_1 = t_2)\,,$$
$$t_1 = t_2 \vee \neg(t_1 = t_2) \models_{\bot} t_1 = t_1 \wedge t_2 = t_2\,.$$

证明系统 BD$^{\supset\,\mathsf{F}}_{\bot,}(\Sigma)$ 的序列演算证明系统，是通过在 BD$^{\supset,\,\mathsf{F}}(\Sigma)$ 的序列演算证明系统中，将推理规则 $=$-Refl 替换为以下两条推理规则而得到的:

$$\boxed{\delta\text{-}=\text{-L}}\ \frac{t_1 = t_1 \wedge t_2 = t_2, \Gamma \Rightarrow \Delta}{t_1 = t_2 \vee t_1 \neq t_2, \Gamma \Rightarrow \Delta}$$

$$\boxed{\delta\text{-}=\text{-R}}\ \frac{\Gamma \Rightarrow \Delta, t_1 = t_1 \wedge t_2 = t_2}{\Gamma \Rightarrow \Delta, t_1 = t_2 \vee t_1 \neq t_2}$$

由此得到的证明系统是可靠且完备的。很容易看出，由此得到的证明系统是可靠的。如 [23], 所述，完备性证明只需对该论文中给出的 BD$^{\supset,\mathsf{F}}(\Sigma)$ 证明系统的完备性证明稍作调整。

缩写 以下内容中将使用以下缩写：

$$\begin{aligned} t{\downarrow} &\text{ stands for } \Delta(t=t) && \text{(term determinacy)},\\ t_1 == t_2 &\text{ stands for } t_1 = t_2 \vee \neg(t_1{\downarrow} \vee t_2{\downarrow}) && \text{(strong equality)}. \end{aligned}$$

由这些定义可得：

$$[\![t{\downarrow}]\!]^{\mathbf{A}}_{\alpha} = \begin{cases} \mathsf{t} & \text{if } [\![t]\!]^{\mathbf{A}}_{\alpha} \in \mathcal{U}^{\mathbf{A}} \setminus \{\bot\} \\ \mathsf{f} & \text{otherwise,} \end{cases}$$

$$[\![t_1 == t_2]\!]^{\mathbf{A}}_{\alpha} = \begin{cases} \mathsf{t} & \text{if } [\![t_1]\!]^{\mathbf{A}}_{\alpha} = \bot \text{ and } [\![t_2]\!]^{\mathbf{A}}_{\alpha} = \bot \\ =^{\mathbf{A}}([\![t_1]\!]^{\mathbf{A}}_{\alpha}, [\![t_2]\!]^{\mathbf{A}}_{\alpha}) & \text{otherwise.} \end{cases}$$

注：$\mathrm{BD}^{\supset\,\mathsf{F}}_{\bot}(\Sigma)$看起来像一种既是次协调又是超完备的自由逻辑 [18,24] 。然而，$\bot$ 属于 $\mathrm{BD}^{\supset\,\mathsf{F}}_{\bot}(\Sigma)$中的变量取值范围，而在自由逻辑中，$\bot$ 则会被排除在变量取值范围之外。这一区别反映出，在 $\mathrm{BD}^{\supset\,\mathsf{F}}_{\bot}(\Sigma)$中，$\bot$ 被视为真正的值，而在自由逻辑中，$\bot$ 并不被视为真正的值。

$\mathrm{BD}^{\supset\,\mathsf{F}}_{\bot}(\Sigma)$旨在处理此类逻辑理论（即公式集）：例如，当可能不一致且可能含空值的关系数据库被视为逻辑理论时，就会产生这类理论。

6 从 $\mathrm{BD}^{\supset\,\mathsf{F}}_{\bot}$ 的视角看关系数据库

本节从 $\mathrm{BD}^{\supset\,\mathsf{F}}_{\bot}$ 的视角考察关系数据库，并考虑到数据库可能是不一致数据库（参见 [6,2]）和／或含空值的数据库（参见 [30,31,15]）。本节采用证明论观点，即将关系数据库视为逻辑理论。在关系数据库概念的定义中，使用了关系语言和关系理论这两个概念。下面先定义这两个概念。本节给出的定义在很大程度上基于 [27] 中的定义。不过，为求简洁，本文忽略类型（参见 [12,29]）。本文对关系数据库中空值的看法已在第 1 节作了非正式说明。

关系语言 若 Σ是一个签名，则二元组（Σ、$\mathcal{F}\mathrm{orm}(\Sigma)$）称为 $\mathrm{BD}^{\supset\,\mathsf{F}}_{\bot}(\Sigma)$ 的语言。若 Σ 满足特定条件，则 $\mathrm{BD}^{\supset\,\mathsf{F}}_{\bot}(\Sigma)$ 的语言被视为一种关系语言。

设 Σ是一个签名。当且仅当语言 $R = (\Sigma, \mathcal{F}\mathrm{orm}(\Sigma))$ 满足以下条件时，它是 $\mathrm{BD}^{\supset\,\mathsf{F}}_{\bot}(\Sigma)$ 的一种关系语言：

– $\mathcal{F}\mathrm{unc}_0(\Sigma)$ 是有限的，$\mathsf{nil} \in \mathcal{F}\mathrm{unc}_0(\Sigma)$，且 $\mathcal{F}\mathrm{unc}_0(\Sigma) \setminus \{\mathsf{nil}\}$ 非空；– $\bigcup\{\mathcal{F}\mathrm{unc}_{n+1}(\Sigma) \mid n \in \mathbb{N}\}$ 为空；– $\mathcal{P}\mathrm{red}_0(\Sigma)$ 为空；– $\bigcup\{\mathcal{P}\mathrm{red}_{n+1}(\Sigma) \mid n \in \mathbb{N}\}$ 有限。

关系理论 下面，我们将介绍关系理论这一概念。在关系理论的定义中，会用到若干辅助概念。首先定义这些辅助概念。

设 $R = (\Sigma, \mathcal{F}\mathsf{orm}(\Sigma))$ 是一种关系语言。那么，对于 R，原子事实是 $\mathcal{F}\mathsf{orm}(\Sigma)$ 中形如 $P(c_1, \ldots, c_{n+1})$ 的公式，其中 $P \in \mathcal{P}\mathsf{red}^{n+1}(\Sigma)$ 且 $c_1, \ldots, c_{n+1} \in \mathcal{F}\mathsf{unc}_0(\Sigma)$。

设 $R = (\Sigma, \mathcal{F}\mathsf{orm}(\Sigma))$ 是一种关系语言。那么，对于 R，nil 不确定性公理是公式

$$\neg(\mathsf{nil}\downarrow)$$

而对于 R，等号半正规性公理是公式

$$\forall x, x' \bullet \circ(x = x') \wedge ((x\downarrow \wedge x'\downarrow) \rightarrow \star(x = x')) \,.$$

设 $R = (\Sigma, \mathcal{F}\mathsf{orm}(\Sigma))$ 是一种关系语言，并设 $c_1, \ldots, c_n$ 是 $\mathcal{F}\mathsf{unc}_0(\Sigma)$ 的所有成员。那么，对于 R，域闭包公理是公式

$$\forall x \bullet (x == c_1 \vee \ldots \vee x == c_n)$$

而对于 R，唯一名称公理集是以下公式集

$$\{\neg(c_i == c_j) \mid 1 \leq i < j \leq n\} \,.$$

设 $R = (\Sigma, \mathcal{F}\mathsf{orm}(\Sigma))$ 是一种关系语言，且 $P \in \mathcal{P}\mathsf{red}^{n+1}(\Sigma)$ $(n \in \mathbb{N})$。那么，对于 R， P-确定性公理是公式

$$\forall x_1, \ldots, x_{n+1} \bullet \star P(x_1, \ldots, x_{n+1}) \,.$$

令 $R = (\Sigma, \mathcal{F}\mathsf{orm}(\Sigma))$ 为关系语言，令 $\Lambda \subseteq \mathcal{F}\mathsf{orm}(\Sigma)$ 为关于 R 的有限原子事实集，并令 $P \in \mathcal{P}\mathsf{red}^{n+1}(\Sigma)$ $(n \in \mathbb{N})$。假设 Λ in 中存在包含 P 的公式，并令 $P(c_1^1, \ldots, c_{n+1}^1), \ldots, P(c_1^m, \ldots, c_{n+1}^m)$ 为 Λ in 中所有包含 P 的公式。那么， Λ 的 P-完备性公理为以下公式

$$\begin{array}{l}\forall x_1, \ldots, x_{n+1} \bullet P(x_1, \ldots, x_{n+1}) \rightarrow \\ \quad x_1 == c_1^1 \wedge \ldots \wedge x_{n+1} == c_{n+1}^1 \vee \ldots \vee x_1 == c_1^m \wedge \ldots \wedge x_{n+1} == c_{n+1}^m \,.\end{array}$$

假设 Λ in 中不存在包含 P 的公式。那么， Λ 的 P-完备性公理为以下公式

$$\forall x_1, \ldots, x_{n+1} \bullet P(x_1, \ldots, x_{n+1}) \rightarrow \mathsf{F} \,.$$

论域闭包、唯一名称和 P-完备性公理取自 [27]。nil 不确定性公理、等号半正规性公理和 P-确定性公理则是新提出的。nil 不确定性公理指出，nil 的取值是不确定的。等号半正规性公理指出，方程按经典方式解释；但若涉及取值不确定的项，其真值既非真也非假。 P-确定性公理指出， P 的真值绝不会既非真也非假。

设 $R = (\Sigma, \mathcal{F}\mathsf{orm}(\Sigma))$ 为一种关系语言。那么， R 的关系结构公理记作 RSA(R)，是满足以下任一条件的所有公式 $A \in \mathcal{F}\mathsf{orm}(\Sigma)$ 的集合：

– A 是 R 的 nil 不确定性公理；– A 是 R 的等号半正规性公理；– A 是 R 的域闭包公理；– A 属于 R 的唯一名称公理集；– 对某个 $P \in \bigcup\{\mathcal{P}\text{red}_{n+1}(\Sigma) \mid n \in \mathbb{N}\}$，$A$ 是 R 的 P-确定性公理。

设 $R = (\Sigma, \mathcal{F}\text{orm}(\Sigma))$ 是一种关系语言，并设 $\Lambda \subseteq \mathcal{F}\text{orm}(\Sigma)$ 是关于 R 的有限原子事实集。那么，以 Λ 为基的 R 的关系理论记作 RT (R, Λ)，它由所有满足以下任一条件的公式 $A \in \mathcal{F}\text{orm}(\Sigma)$ 构成：

– $A \in \text{RSA}(R)$; – $A \in \Lambda$; – 对某个 $P \in \bigcup\{\mathcal{P}\text{red}_{n+1}(\Sigma) \mid n \in \mathbb{N}\}$，$A$ 是 P 完备性公理。

若集合 $\Theta \subseteq \mathcal{F}\text{orm}(\Sigma)$ 对某个有限的原子事实集 $\Lambda \subseteq \mathcal{F}\text{orm}(\Sigma)$（关于 R）满足 $\Theta =$ RT (R, Λ)，则称其为 R 的关系理论。这个唯一公式集中的元素称为 Θ 的原子事实。

以下定理给出了关于序列可证明性的可判定性结果 $\Gamma \Rightarrow A$，其中 Γ 包含某种关系语言的关系结构公理。

定理 2. 设 $R = (\Sigma, \mathcal{F}\text{公式}(\Sigma))$ 是一种关系语言，并设 Γ是 $\mathcal{F}$公式(Σ) 的有限子集，且 RSA$(R) \subseteq \Gamma$。那么，对于公式 $A \in \mathcal{F}$公式(Σ)，$\Gamma \Rightarrow A$ 是否可证明是可判定的。

证明. 根据定理 1，已知 $\Gamma \Rightarrow A$ 可证明当且仅当 $\Gamma \vDash A$，因此只需证明：对于公式 $A \in \mathcal{F}$公式(Σ)，$\Gamma \vDash A$是否可判定。

由于 RSA$(R) \subseteq \Gamma$，只需考虑作为 RSA(R) 模型的结构。这些结构的论域具有相同的有限基数。此外，由于 Σ 中只有有限多个谓词符号，这些结构的数量也有限。

显然，只需考虑赋值在 $\Gamma \cup \{A\}$ 中出现的所有变量集合上的限制。由于 $\Gamma \cup \{A\}$ 中出现的变量集合有限，且要考虑的结构的论域有限，因此这样的限制只有有限多个，并且每个限制都是有限的。

根据上述有限性性质和命题 1，很容易得出，对于公式 $A \in \mathcal{F}\text{orm}(\Sigma)$，$\Gamma \vDash A$ 是否可判定。□

关系数据库　在定义了关系语言和关系理论的概念后，我们现在可以在 $\text{BD}^{\supset}_{\perp}{}^{\mathsf{F}}$, 的框架下定义关系数据库的概念。

关系数据库 DB 是一个三元组（R，Θ，Ξ），其中：

– $R = (\Sigma, \mathcal{F}\text{orm}(\Sigma))$ 是一种关系语言；– Θ 是关于 R 的关系理论；– Ξ 是 $\mathcal{F}\text{orm}(\Sigma)$ 的有限子集。

Θ 称为数据库的关系理论， Ξ 称为数据库的完整性约束集。

关系数据库 DB $=(R, \Theta, \Xi)$ 的完整性约束集 Ξ可视为关于关系数据库 Θ 的关系理论的一组假设。如果关系理论符合这些假设，则称该关系数据库是一致的。

设 $R=(\Sigma, \mathcal{F}\mathrm{orm}(\Sigma))$ 是一种关系语言，且 DB $=(R, \Theta, \Xi)$ 是一个关系数据库。那么，当且仅当对于每个 $A \in \mathcal{F}\mathrm{orm}(\Sigma)$，若 A 是关于 R 的原子事实，或 A 的形式为 $\neg A'$，其中 A' 是关于 R 的原子事实时，DB 是一致的：

$$\Theta \Rightarrow A \text{ is provable only if } \Theta, \Xi \Rightarrow \circ A \text{ is provable.}$$

请注意，如果 DB 不一致，那么对于所有 $A' \in \mathcal{F}\mathrm{orm}(\Sigma)$，都可以用 $\mathbf{CL}^{\supset, F}(\Sigma)$ 的序列演算证明系统证明 $\Theta, \Xi \Rightarrow A'$。然而，$\mathbf{BD}_{\perp}^{\supset F,}(\Sigma)$ 的序列演算证明系统排除了这种爆炸原理。

关系理论的模型 对于关系语言 $R=(\Sigma, \mathcal{F}\mathrm{orm}(\Sigma))$，关系理论的模型是具有特殊类型的 $\mathbf{BD}_{\perp}^{\supset F,}(\Sigma)$ 结构。

设 $R=(\Sigma, \mathcal{F}\mathrm{orm}(\Sigma))$ 为一种关系语言。那么， R 的关系结构是一个 $\mathbf{BD}_{\perp}^{\supset F,}(\Sigma)$ 结构 A，满足：

– $\mathrm{nil}^{\mathbf{A}}=\perp$；– 对所有 $d_1, d_2 \in \mathcal{U}^{\mathbf{A}}$:$=^{\mathbf{A}}(d_1, d_2) \neq \mathsf{b}$；若 $d_1 \neq \perp$ 且 $d_1 \neq \perp$，则 $=^{\mathbf{A}}(d_1, d_2) \neq \mathsf{n}$；– 对所有 $d \in \mathcal{U}^{\mathbf{A}}$：若 $d \neq \perp$，则存在一个 $c \in \mathcal{F}\mathrm{unc}_0(\Sigma)$，使得 $=^{\mathbf{A}}(d, c^{\mathbf{A}})=\mathsf{t}$；若 $d=\perp$，则存在一个 $c \in \mathcal{F}\mathrm{unc}_0(\Sigma)$，使得 $=^{\mathbf{A}}(d, c^{\mathbf{A}})=\mathsf{n}$；– 对所有 $c_1, c_2 \in \mathcal{F}\mathrm{unc}_0(\Sigma)$：若 $=^{\mathbf{A}}(c_1{}^{\mathbf{A}}, c_1{}^{\mathbf{A}}) \neq \mathsf{n}$ 且 $=^{\mathbf{A}}(c_2{}^{\mathbf{A}}, c_2{}^{\mathbf{A}}) \neq \mathsf{n}$，且 $=^{\mathbf{A}}(c_1{}^{\mathbf{A}}, c_2{}^{\mathbf{A}})=\mathsf{t}$ 或$=^{\mathbf{A}}(c_1{}^{\mathbf{A}}, c_1{}^{\mathbf{A}})=\mathsf{n}$ 且 $=^{\mathbf{A}}(c_2{}^{\mathbf{A}}, c_2{}^{\mathbf{A}})=\mathsf{n}$，则 $c_1 \equiv c_2$；– 对所有 $n \in \mathbb{N}$，对所有 $P \in \mathcal{P}\mathrm{red}^{n+1}(\Sigma)$，对所有 $d_1, \ldots, d_{n+1} \in \mathcal{U}^{\mathbf{A}}$:$P^{\mathbf{A}}(d_1, \ldots, d_{n+1}) \neq \mathsf{n}$。

以下推论由 R 的关系结构公理定义以及 R 的关系结构定义得出，它说明了“关系结构公理”这一术语的合理性。

推论 1. 设 $R=(\Sigma, \mathcal{F}$ 或 $\mathrm{m}(\Sigma))$ 是一种关系语言，且 A 是 $\mathbf{BD}_{\perp}^{\supset F,}(\Sigma)$ 的一个结构。当且仅当对于 A 中的所有赋值 α，以及所有 $A \in \mathrm{RSA}(R)$，都有 $[\![A]\!]_{\alpha}^{\mathbf{A}} \in \mathcal{D}$ 时，A 才是 R 的关系结构。

令 $R=(\Sigma, \mathcal{F}\mathrm{orm}(\Sigma))$ 为一种关系语言，并令 Θbe 为 R 的关系理论。那么， Θ 的所有模型都是 R 的关系结构，因为 $\mathrm{RSA}(R) \subseteq \Theta$。 Θ 在同构意义下没有唯一模型。Θ 的谓词

完备性公理无法确保模型在同构意义下唯一。不过，在 Θ 的模型中将 t 和 b 识别为同一对象，即可保证同构意义下的唯一性。

设 $R=(\Sigma,\mathcal{F}\mathsf{orm}(\Sigma))$ 是一种关系语言，A 是 R 的一个关系结构。于是，我们将 ∇A 写作满足以下条件的关系结构 A′：

- $\mathcal{U}^{\mathbf{A}'}=\mathcal{U}^{\mathbf{A}}$;
- for each $c\in\mathcal{F}unc_0(\Sigma)$, $c^{\mathbf{A}'}=c^{\mathbf{A}}$;
- for each $n\in\mathbb{N}$, for each $P\in\mathcal{P}red_{n+1}(\Sigma)$, for each $d_1,\ldots,d_{n+1}\in\mathcal{U}^{\mathbf{A}'}$, $P^{\mathbf{A}'}(d_1,\ldots,d_{n+1})=\begin{cases}\mathsf{t} & \text{if } P^{\mathbf{A}}(d_1,\ldots,d_{n+1})\in\{\mathsf{t},\mathsf{b}\}\\ \mathsf{f} & \text{otherwise;}\end{cases}$
- for each $d_1,d_2\in\mathcal{U}^{\mathbf{A}'}$, $=^{\mathbf{A}'}(d_1,d_2)\;=\;=^{\mathbf{A}}(d_1,d_2)$.

定理 3。设 $R=(\Sigma,\mathcal{F}\mathsf{orm}(\Sigma))$ 是一种关系语言，Θ 是 R 的一个关系理论，且 A 和 A′ 是 Θ 的模型。那么，∇A 和 ∇A′ 是同构的关系结构。

证明。证明过程与 [27] 中定理 3.1 第 1 部分的证明几乎相同。唯一需要注意的是，对于某些 $P\in\mathcal{P}\mathsf{red}_{n+1}(\Sigma)$ 和 $c_1,\ldots,c_{n+1}\in\mathcal{F}\mathsf{unc}_0(\Sigma)$ $(n\in\mathbb{N})$，可能出现以下情况：要么 $[\![P(c_1,\ldots,c_{n+1})]\!]_\alpha^{\mathbf{A}}=\mathsf{t}$ 且 $[\![P(c_1,\ldots,c_{n+1})]\!]_\alpha^{\mathbf{A}'}=\mathsf{b}$，要么 $[\![P(c_1,\ldots,c_{n+1})]\!]_\alpha^{\mathbf{A}}=\mathsf{b}$ 且 $[\![P(c_1,\ldots,c_{n+1})]\!]_\alpha^{\mathbf{A}'}=\mathsf{t}$。不过，若确实如此，则 $[\![P(c_1,\ldots,c_{n+1})]\!]_\alpha^{\nabla\mathbf{A}}=\mathsf{t}$ 且 $[\![P(c_1,\ldots,c_{n+1})]\!]_\alpha^{\nabla\mathbf{A}'}=\mathsf{t}$。□

定理 4. 设 $R=(\Sigma,\mathcal{F}\text{公式集}(\Sigma))$ 是一种关系语言，A 是 R 的关系结构。那么存在一个关于 R 的关系理论 Θ，使得 A 是 Θ 的模型。

证明. 证明过程与 [27] 中定理 3.1 第 2 部分的证明相同。□

7 从 $\mathsf{BD}^{\supset,\mathsf{F}}_{\perp}$ 的角度看查询回答

本节从 $\mathsf{BD}^{\supset,\mathsf{F}}_{\perp}$ 的角度考察适用于关系数据库的查询及其答案。事实上，下面介绍的查询与 [10] 最初提出的关系演算导向的查询密切相关。本文介绍了三种查询答案，其中两种会考虑数据库的完整性约束。它们处理不一致性的方式各不相同。关于这些方法的讨论见 [22, 第 8] 节。

查询 如当前设定所预期的，适用于关系数据库的查询涉及一个 $\mathsf{BD}^{\supset,\mathsf{F}}_{\perp}$ 公式。

设 $R=(\Sigma,\mathcal{F}\mathsf{orm}(\Sigma))$ 为一种关系语言。那么，R 的查询是形如 $(x_1,\ldots,x_n)\bullet A$ 的表达式，其中：

– $x_1, \ldots, x_n \in \mathcal{V}$元；– $A \in \mathcal{F}\mathsf{orm}(\Sigma)$，且在 A 中自由出现的所有变量都属于 $x_1, \ldots, x_n$。

设 DB $= (R, \Theta, \Xi)$ 为一个关系数据库。那么，当且仅当某个查询是针对 R 的查询时，该查询才适用于 DB。

答案 对一致关系数据库进行查询回答，相当于寻找相关公式的闭实例，这些实例是关系理论的逻辑后承。查询回答的主要问题是如何处理不一致关系数据库。

设 $R = (\Sigma, \mathcal{F}\mathsf{orm}(\Sigma))$ 是一种关系语言，设 DB $= (R, \Theta, \Xi)$ 是一个关系数据库，并设 $(x_1, \ldots, x_n) \bullet A$ 是一个适用于 DB 的查询。那么，相对于 DB，$(x_1, \ldots, x_n) \bullet A$ 的一个答案是一个 $(c_1, \ldots, c_n) \in \mathcal{F}\mathsf{unc}_0(\Sigma)^n$，且 $\Theta \Rightarrow [x_1 := c_1] \ldots [x_n := c_n]A$ 可证。

上述定义的针对数据库的查询答案，并未考虑相关数据库的完整性约束。

一致答案 以下给出的一致答案定义基于以下几点：

– 与某个答案相对应的公式，是某组原子事实及原子事实的否定的逻辑后承，而这些事实和否定是数据库的关系理论的逻辑后承；– 对于一致答案，必须存在这样一组事实，其中不包含会导致数据库不一致的原子事实或原子事实的否定。

设 $R = (\Sigma, \mathcal{F}\mathsf{orm}(\Sigma))$ 是一种关系语言。那么，对于 R，半原子事实是一个 $A \in \mathcal{F}\mathsf{orm}(\Sigma)$，满足 A 是 R 的原子事实，或者 A 具有 $\neg A'$ 的形式，其中 A' 是 R 的原子事实。

设 $R = (\Sigma, \mathcal{F}\mathsf{orm}(\Sigma))$ 是一种关系语言，设 DB $= (R, \Theta, \Xi)$ 是一个关系数据库，并设 $(x_1, \ldots, x_n) \bullet A$ 是一个适用于 DB 的查询。那么，针对 DB 的 $(x_1, \ldots, x_n) \bullet A$ 的一致答案，是一个 $(c_1, \ldots, c_n) \in \mathcal{F}\mathsf{unc}_0(\Sigma)^n$，且存在一组 Φ 半原子事实（针对 R），使得：

– 对所有 $A' \in \Phi$，$\Theta \Rightarrow A'$ 均可证，且 Θ，$\Xi \Rightarrow \circ A'$ 均可证；– Φ，$\mathsf{RSA}(R) \Rightarrow [x_1 := c_1] \ldots [x_n := c_n]A$ 可证。

上述关于相对于数据库的查询一致答案的定义，让人想起 [6] 中给出的关于相对于数据库的查询一致答案的定义。它只是承认数据库可能不一致，并将导致不一致的源排除在一致查询回答的使用范围之外。

强一致答案 下面给出的强一致答案定义对不一致性的容忍度较低，并使用数据库的一致修复。其思路是，如果某个答案相对于原始数据库的每个最小修复版本都是答案，那么它就是强一致的。

设 $R = (\Sigma, \mathcal{F}\text{orm}(\Sigma))$ 是一种关系语言，且 $\Lambda \subseteq \mathcal{F}\text{orm}(\Sigma)$ 是关于 R 的有限原子事实集。那么，按照 [2], the binary relation $\leq_\Lambda$ ，在所有关于 R 的有限原子事实集构成的集合上定义如下：

$$\Lambda' \leq_\Lambda \Lambda'' \text{ iff } (\Lambda \setminus \Lambda') \cup (\Lambda' \setminus \Lambda) \subseteq (\Lambda \setminus \Lambda'') \cup (\Lambda'' \setminus \Lambda) .$$

直观而言， $\Lambda' \leq_\Lambda \Lambda''$ 表示 Λ' 与 Λ 的差异程度小于 Λ'' 与 Λ 的差异程度。

设 $R = (\Sigma, \mathcal{F}\text{orm}(\Sigma))$ 是一种关系语言，设 $\Lambda \subseteq \mathcal{F}\text{orm}(\Sigma)$ 是关于 R 的有限原子事实集，并设 Ξ 是 $\mathcal{F}\text{orm}(\Sigma)$ 的有限子集。那么， Λ 与 Ξ 一致，当且仅当对于关于 R 的所有半原子事实 A，只有在 RT (R, Λ) $\Xi \Rightarrow \circ A$ 可证时，RT (R, Λ) $\Rightarrow A$ 才可证。我们用 Con(Ξ) 表示所有关于 R 且与 Ξ 一致的有限原子事实集构成的集合。

设 $R = (\Sigma, \mathcal{F}\text{orm}(\Sigma))$ 是一种关系语言， $\Lambda \subseteq \mathcal{F}\text{orm}(\Sigma)$ 是关于 R 的有限原子事实集，DB $= (R,$ RT (R, Λ) $\Xi)$ 是一个关系数据库，而 $(x_1, \ldots, x_n) \bullet A$ 是一个适用于 DB 的查询。那么，相对于 DB，$(x_1, \ldots, x_n) \bullet A$ 的强一致答案是一个 $(c_1, \ldots, c_n) \in \mathcal{F}\text{unc}_0(\Sigma)^n$，满足：对于 Con$(\Xi)$ 中每个 Λ'，若它在 Con(Ξ) 中是 $\leq_\Lambda$-极小的，则 RT (R, Λ') $\Rightarrow [x_1 := c_1] \ldots [x_n := c_n] A$ 可证。Con(Ξ) 中在 Con(Ξ) 中为 $\leq_\Lambda$-极小的元素称为 Λ 的修复。

上述关于相对于数据库的查询强一致答案的定义，实质上与 [2] 中给出的相对于数据库的查询一致答案定义相同。它大概代表了关于不一致数据库的修复是什么的第一种观点。关于不一致数据库的修复是什么，其他观点见例如 [20,14,8,7,4]。

答案中的空值 如果 $(c_1, \ldots, c_n)$ 是相对于含空值的数据库而言，某个查询的答案、一致答案或强一致答案，那么 $c_1, \ldots, c_n$ 中的每一项在语法上都可能等于 nil。这类答案称为含空值的答案。在大多数关于相对于含空值的数据库进行查询回答的研究中，都不考虑含空值的答案。值得注意的例外包括 [15,19], ，其中对关系数据库中的空值所持的观点与本文不同。

可判定性 以下定理是关于相对于数据库而言某个查询的答案是否成立的可判定性结果。

定理 5. 设 $R = (\Sigma, \mathcal{F}\text{orm}(\Sigma))$ 是一种关系语言，设 DB $= (R, \Theta, \Xi)$ 是一个关系数据库，且 $(x_1, \ldots, x_n) \bullet A$ be 是一个适用于 DB 的查询。那么，对于 $(c_1, \ldots, c_n) \in \mathcal{F}\text{unc}_0(\Sigma)^n$，以下问题均可判定：

– $(c_1, \ldots, c_n)$ 是 $(x_1, \ldots, x_n) \bullet A$ 关于 DB 的一个答案；– $(c_1, \ldots, c_n)$ 是 $(x_1, \ldots, x_n) \bullet A$ 关于 DB 的一个一致答案；– $(c_1, \ldots, c_n)$ 是 $(x_1, \ldots, x_n) \bullet A$ 关于 DB 的一个强一致答案。

证明。所有这些可判定性结果都可直接由定理 2 以及相关答案类型的定义得出。□

作为定理 5 的推论，我们知道，查询的答案集合、查询的一致答案集合以及查询的强一致答案集合都是可计算的。

例子 下面给出的例子极其简单，以便尚未熟悉 $\mathrm{BD}^{\neg}_{\perp}$ 序列演算证明系统的读者也能理解关于序列可证明性的说明。

考虑这样一个关系数据库：其关系语言（记作 R）包含常量符号 a、 b、 c、 d 和 nil，以及三元谓词符号 P；其关系理论包含原子事实 $P(a, b, \mathsf{nil})$、 $P(a, \mathsf{nil}\ c)$、 $P(a, \mathsf{nil}\ d)$ 和 $P(b, c, d)$；其唯一的完整性约束为 $\forall x, y, z, y', z' \bullet (P(x, y, z) \land P(x, y', z')) \rightarrow y = y'$。此外，考虑查询 $x \bullet \exists y, z, z' \bullet P(x, y, z) \land P(x, y, z') \land z \neq z'$。显然，答案集合为 $\{a\}$。

关系理论的逻辑后承中、且不会导致数据库不一致的半原子公式集，只包含以下各项中的一项： $P(a, b, \mathsf{nil})$、 $P(a, \mathsf{nil}\ c)$ 和 $P(a, \mathsf{nil}\ d)$。这意味着，对于每个这样的集合，例如 Φ，不存在常元符号 $k \in \{a, b, c, d, \mathsf{nil}\}$ 使得 Φ，$\mathrm{RSA}(R) \Rightarrow [y := k](\exists x, z, z' \bullet P(x, y, z) \land P(x, y, z') \land z \neq z')$ 可被证明。因此，一致答案集为 $\emptyset$。

$\{P(a, b, \mathsf{nil})\ P(a, \mathsf{nil}\ c)\ P(a, \mathsf{nil}\ d)\ P(b, c, d)\}$ 的修复只包含以下各项中的一项： $P(a, b, \mathsf{nil})$、 $P(a, \mathsf{nil}\ c)$ 和 $P(a, \mathsf{nil}\ d)$。这意味着，对于每个修复，例如 Λ，不存在常元符号 $k \in \{a, b, c, d, \mathsf{nil}\}$ 使得 $\mathrm{RT}(R, \Lambda) \Rightarrow [y := k](\exists x, z, z' \bullet P(x, y, z) \land P(x, y, z') \land z \neq z')$ 可被证明。因此，强一致答案集也为 $\emptyset$。

现在，考虑一个与上述数据库唯一不同之处在于完整性约束 $y = y'$ 被 $y == y'$ 替换的关系数据库。不难看出，在这种情况下，上述查询的答案集和强一致答案集保持不变，但一致答案集变为 $\{a\}$。

在 [22] 中可以找到几个空值不起作用的例子。

8 数据库中关于空值的 Viewson

本节简要介绍关于关系数据库中的空值的不同观点。

我们的观点 本文对关系数据库中的空值所持的观点可概括如下：

– 在含有空值的关系数据库中，对于不确定的值，使用一个称为“空值”的虚设值；– 不确定的值是指未知或不存在的值；– 无论该值是未知还是不存在，对于“空值是否与某个值（包括空值本身）相同”这一问题，都无法给出有意义的答案。

在数据库文献中，将表示所有不确定值（即所有未知或不存在的值）的空值通常称为无信息空值。此外，常用“不适用”一词代替“不存在”。

其他观点 首先，本文采用的观点认为，未知值和不存在的值并没有各自独立的空值。有关未知值和不存在的值分别使用独立空值的研究，例如见 [30,13]。在这些论文中，查询回答基于一种为处理这两种不同空值而设计的四值逻辑。相关逻辑彼此不同，而且都与 $\mathbf{BD}^{\supset}_{\perp}{}^{\mathsf{F}}$ 有很大差异。

其次，本文采用的观点意味着，空值作为虚设值所代表的取值范围，并不限于未知值。例如， [11,5,15]研究过仅表示未知值的单一空值。在 [15] 及其后续出版物中，考虑了这类空值；对于含空值的数据库，通常采用一种基于如下思想的方式来进行查询回答：含空值的数据库代表一组不含空值的可能数据库。这种方式让人想起 [2]中针对不一致数据库进行查询回答的数据库修复方法。

第三，本文采用的观点意味着不存在多个空值。例如， [15,19]研究过多个空值，其中每个空值都仅表示未知值。在那些论文中，对于单一空值的情形，采用了与 [15] 相同的查询回答方法。

此前也有人研究过一种同时表示未知值和不存在值的单一空值，见 [31,17]。在 [31], 中，查询回答基于一种与 $\mathbf{BD}^{\supset}_{\perp}{}^{\mathsf{F}}$ 密切相关的三值逻辑。在 [17], 中，作者尝试将 [15] 中的查询回答方法应用于同时表示未知值和不存在值的单一空值情形。

人们对关系数据库中的空值提出了如此多种观点，这一事实表明，先前研究的观点一次又一次地被证明在某些方面并不令人满意。本文所采用的观点主要基于这样的考虑：从逻辑视角来看，这种观点应当相对简单，同时也要考虑到，数据库无论如何都不过是对一部分现实的近似表示。

相等性问题 本文所采用的观点与 [31] 中采取的观点基本相同，并基于类似的考虑。这种观点有一个方面在关于关系数据库中空值的研究中常被认为不尽如人意，即无法对“空值是否等于其自身”这一问题给出有意义的答案。在本文的语境下将其精确定义后，常被认为不尽如人意的内容如下：

$$[\![t_1 = t_2]\!]^{\mathbf{A}}_{\alpha} = \mathsf{n} \quad \text{if} \quad [\![t_1]\!]^{\mathbf{A}}_{\alpha} = [\![\mathsf{nil}]\!]^{\mathbf{A}}_{\alpha} \text{ and } [\![t_2]\!]^{\mathbf{A}}_{\alpha} = [\![\mathsf{nil}]\!]^{\mathbf{A}}_{\alpha} .$$

认为这种做法不可取的主要理由是，它可能导致对查询给出违背直觉的答案。然而，只有在（隐含地）假设空值仅仅是未知值的虚设值时，才会出现这种情况。在这种情况下，以下做法被认为更为恰当：

$$[\![t_1 = t_2]\!]^{\mathbf{A}}_{\alpha} = \mathsf{t} \quad \text{if} \quad [\![t_1]\!]^{\mathbf{A}}_{\alpha} = [\![\mathsf{nil}]\!]^{\mathbf{A}}_{\alpha} \text{ and } [\![t_2]\!]^{\mathbf{A}}_{\alpha} = [\![\mathsf{nil}]\!]^{\mathbf{A}}_{\alpha}$$

（例如，参见 [19]。）

值得一提的是，在本文所采用的框架下，可以表达经过调整的等号谓词。考虑前文介绍的缩写 $t_1 == t_2$ 。如前所述，根据该缩写的定义可得：

$$[\![t_1 == t_2]\!]^{\mathbf{A}}_{\alpha} = \begin{cases} \mathsf{t} & \text{if } [\![t_1]\!]^{\mathbf{A}}_{\alpha} = \bot \text{ and } [\![t_2]\!]^{\mathbf{A}}_{\alpha} = \bot \\ =^{\mathbf{A}}([\![t_1]\!]^{\mathbf{A}}_{\alpha}, [\![t_2]\!]^{\mathbf{A}}_{\alpha}) & \text{otherwise.} \end{cases}$$

显然， $==$ 对应于经过调整的等号谓词。这意味着，尽管第6节和第7节中的定义基于这样的观点：空值是未知值和不存在值的虚设值，$\mathrm{BD}^{\supset\,\mathsf{F}}_{\bot}$，仍允许将空值视为仅仅是未知值的虚设值。

9 结语

本文提出了一种关于关系数据库和查询回答的协调一致的逻辑视角，兼顾数据库可能含有空值以及数据库可能是不一致数据库这两种情况。所提出的视角结合了：

– Reiter 的证明论逻辑观点 [27]，关于什么是关系数据库、适用于关系数据库的查询，以及针对不含空值的一致关系数据库的查询答案；– Zaniolo 的观点 [31]，关于关系数据库中的空值；– Bry 的观点 [6] 以及 Arenas 等人的观点 [2]，关于针对不一致关系数据库的查询的一致答案是什么。

该观点是在次协调且次完备的逻辑 $\mathrm{BD}^{\supset\,\mathsf{F}}_{\bot}$，框架下提出的。在非次完备逻辑中，很难

恰当地考虑数据库可能是含空值的数据库这一情况；而在非次协调逻辑中，也很难恰当地考虑数据库可能是不一致数据库这一情况。

关于相对于不一致关系数据库而言，什么才是查询的一致答案，主要有两种观点：Bry 的观点 [6] 以及 Arenas 等人的观点 [2]。后者基于不一致数据库的最小修复这一概念。其他大多数关于一致查询回答的观点，例如 [20,14,8,7,4], 中提出的观点，都基于一种与 [2] 不同的不一致数据库最小修复概念，其差异体现在多个方面的不同选择上，包括为获得修复而对原始数据库所做的修改（删除、添加、变更）的类型，以及如何衡量两个数据库之间的差异程度。

关于关系数据库中的空值，主要有四种观点，分别是 Zaniolo 的观点 [31], 、Imieli´nski 和 Lipski 的两种观点 [15], ，以及 Vassilion 的观点 [30]。其他观点大多是其中某一种观点的变体。 [15], 中的 Codd 表视图不存在多个空值，可以在 $\mathrm{BD}^{\supset}_{\perp}{}^{\mathsf{F}}{}_{,}$ 的框架下表达，因为按照该观点，可以使用 == 而非 =来处理这个单一空值。 [15], 中的 V 表视图则可能包含多个空值（称为标记空值）。要表达这种视图，需要对 $\mathrm{BD}^{\supset}_{\perp}{}^{\mathsf{F}}{}_{,}$ 作相应变体。 [30], 中的观点将未知值和不存在值分别对应于不同的空值。$\mathrm{BD}^{\supset}_{\perp}{}^{\mathsf{F}}{}_{,}$ 是否适合表达这种关于关系数据库中的空值的观点，值得商榷；尤其是还必须考虑数据库可能不一致的情况。

本文表明，对于所介绍的三种答案类型中的每一种，判断某个答案是否为相对于关系数据库的查询答案都是可判定的。然而可以预见，若不对完整性约束和/或空值的使用施加严格限制，这几类答案的查询回答将难以处理。在 $\mathrm{BD}^{\supset}_{\perp}{}^{\mathsf{F}}{}_{,}$的设定下，研究这几类答案的查询回答的计算复杂性，是未来工作中一个有趣的方向。

参考文献

1. A. R. Anderson 和 N. D. Belnap。一阶蕴涵。*Mathematische Annalen* 149 4 302 319, 1963 10 1007 , ( ): – 。

doi: . /BF01471125 2. M. Arenas、L. Bertossi 和 J. Chomicki。不一致数据库中的一致查询答案。载于 *PODS 1999*，第 68–79 页。ACM Press，1999。doi:10.1145/303976.303983 3. O. Arieli 和 A. Avron。四值旁一致逻辑。*Studia Logica*, 105 6 1087 1122, 2017 10 1007 11225 017 9721 ( ): – 。doi: . /s - - -4 4. L. Bertossi。数据库修复与一致查询回答：起源及后续发展。载于 *PODS 2019*，第 48–58 页。ACM Press，2019。doi:10.1145/3294052.3322190 5. J. Biskup。Codd 关系型 maybe 操作的基础。*ACM Transactions on Database Systems* 8 4 608 636, 1983 10 1145 319996 320014 , ( ): – 。

doi: . / 。

6. F. Bry。带完整性约束的信息系统中的查询回答。载于 S. Ja- jodia、W. List、G. McGregor 和 L. Strous 编， *IICIS 1997*，IFIPAICT，第 113 130, 1997 10 1007 978 0 387 35317 3_6 – 伦敦，. Chapman 7 and Hall。doi: . / - - - - 。M. Calautti、L. Libkin 和 A. Pieris。一种一致查询回答的操作性方法。载于 *PODS 2018*，第 239–251 页。ACM Press，2018。doi:10.1145/3196959.3196966 8. B. ten Cate、G. Fontaine 和 P. G. Kolaitis。关于一致查询回答的数据复杂性。*Theory of Computing Systems*, 57(4):843–891, 2015。doi:10.1007/s00224-014-9586-0 9. R. Ciuni 和 M. Carrara。多值逻辑中的正常性算子与经典逻辑回收。*Logic Journal of the IGPL*, 28(5):657–683, 2020。doi:10 055 10 1093. /jigpal/jzy 。E. F. Codd。数据库子语言的关系完备性。技术报告 RJ 987，IBM，加利福尼亚州圣何塞，1972。11. E. F. Codd。扩展数据库关系模型以涵盖更多含义。*ACM Transactions on Database Systems*, 4(4):397–434, 1979。doi:10.1145/320107.320109 12. H. Gallaire、J. Minker 和 J.-M. Nicolas。逻辑与数据库：一种演绎方法。*Computing Surveys* 16 2 153 185, 1984 10 1145 356924 356929 , ( ): – 。doi: . / 。G. H. 13 Gessert。关系数据库系统的四值逻辑。*ACM SIGMOD Record* 19 3 29 35, 1990 10 1145 382274 , ( ): – 。doi: . / .382401

14. S. Greco 和 C. Molinaro。在不一致数据库上的一致查询回答。*International Journal of Knowledge-based and Intelligent Engineering Systems*, 15 3 119 129, 2011 10 3233 2011 ( ): – 。doi: . /KES- -15 0218 ´ . T. Imielinski 和 W. Lipski。关系数据库中的不完整信息。*Journal of the ACM* 31 4 761 791, 1984 10 1145 1634 , ( ): – 。doi: . / . 1886 16 *Introduction to Metamathematics* . S. C. Kleene。 1886 16 *Introduction to Metamathematics* 。North-Holland，阿姆斯特丹，1952。

17. H.-J. Klein。关系数据库中的空值与确定信息答案。载于 L. Bertossi、G. O. H. Katona、K.-D. Schewe 和 B. Thalheim 编，*in Databases* 2582 *Lecture Notes in Computer Science* 119 138 ，第 2003 10 1007 3 540 36596 6_7 卷（ 18 2003 10 1007 3 540 36596 6_7 ），第 – 页。Springer-Verlag, 18 2003 10 1007 3 540 36596 6_7。doi: . / - - - 。K. Lambert。自由逻辑与存在概念。 18 *Semantics Notre Dame Journal of Formal Logic* 8 1 2 133 144, 1967 10 1305 , (–): – 。doi: . / ndjfl/1093956251 19. L. Libkin。SQL 的三值逻辑与确定答案。*ACM Transactions on Database Systems* 41 1 1, 2016 10 1145 , ( ):Article 。doi: . / 2877206 20. A. Lopatenko 和 L. Bertossi。通过最小规模修复实现一致查询回答。载于 *DEXA 2006* 558 562 2006 10 1109 2006 43 , 第 – 页。IEEE, 。doi: . / 21 DEXA. . . C. A. Middelburg。论经典逻辑所包含的最强三值次协调逻辑及其对偶。*Journal of Logic and Computation* 31 2 597 611, , ( ): – 。doi: . / 2021 10 1093 logcom/exaa084 22. C. A. Middelburg。不一致数据库中的次协调逻辑与查询回答，2022 12976 23 2208。arXiv: . 。C. A. Middelburg。一阶 Belnap-Dunn 逻辑的常规扩展，2023。arXiv:2301.10555 24. J. Nolt。自由逻辑。载于 D. Jacquette 编， *Philosophy of Logic*，第 1023–1060 页。Elsevier，2007 0 25 10 1016 978 044451541 4 50027 。doi: . /B - - / - 。L. Picollo。形式不一致逻辑中的真值：经典性究竟能达到何种程度? *Logic Journal of the IGPL* 28 5 771 806, 2020 10 1093 , ( ): – 。doi: . / jigpal/jzy059 26 8 1 219 。G. Priest。悖论逻辑。 *Journal of Philosophical Logic*, ( ): –241, 1979。doi:10.1007/BF00258428

27. R. Reiter. 关系数据库理论的逻辑重构。载于 M. L. Brodie、J. Mylopoulos 和 J. W. Schmidt 编，*On Conceptual Modelling*，第 191 238 1984 10 1007 978 1 4612 5196 5_8 – 页。Springer-Verlag，。doi: 。/ - - - - 。K. Sano 和 H. Omori。一阶 Belnap-Dunn 逻辑的扩展。*Journal of the IGPL* 22 3 458 481, 2014 10 1093 ，( ): – 。doi: 。/ jigpal/jzt044 29. M. Y. Vardi。查询逻辑数据库。*Journal of Computer and System Sciences* 33 2 142 160, 1986 10 1016 0022 0000 86 90016 ，( ): – 。doi: 。/ - ( ) -4 30. Y. Vassiliou。数据管理中的空值：一种指称语义方法。载于 *SIGMOD '79*，第 162–169 页。ACM Press，1979。doi:10.1145/582095.582123 31. C. Zaniolo。含空值的数据库关系。*Journal of Computer and System Sciences* 28 1 142 166, 1984 10 1016 0022 0000 84 90080 1 ，( ): – 。doi: 。/ - ( ) -