arXiv:2603.01407v1 [cs.AI] 2 Mar 2026

观察者–情境格：视角感知认知的统一形式化基础

Saad Alqithami，巴哈大学计算机科学系，
巴哈 65779，沙特阿拉伯，
salqithami@bu.edu.sa

摘要

在复杂的多智能体环境中运行的自主智能体，必须从多个视角推理何为真实。既有方法往往难以整合不同主体在不同时间、不同情境下的推理，通常将这些维度交由彼此独立的专用模块处理。这种碎片化会导致推理过程脆弱且不完整，尤其是在主体必须理解他人信念（心智理论）时。我们提出观察者–情境格（OSL），这是一种统一的数学结构，为视角感知认知提供单一、连贯的语义空间。OSL 是一个有限完备格，其中每个元素都表示唯一的观察者–情境对，从而为信念管理提供了一种有原则且可扩展的方法。我们提出了两个在该格上运行的关键算法：（i）相对化信念传播，一种能够高效传播新信息的增量更新算法；（ii）最小矛盾分解，一种用于识别并隔离矛盾分量的基于图的过程。我们证明了该框架在理论上的可靠性，并通过一系列基准测试展示其实用价值，其中包括经典的心智理论任务，以及与基于假设的真值维护系统等既有范式的比较。结果表明，OSL 为构建稳健、具备视角感知能力的自主智能体提供了计算高效且表达力强的基础。

关键词

依赖观察者的语义、可解释的认知主体、信念-愿望-意图（BDI）、全局工作空间架构、心智理论规划、情境感知、元认知目标推理、神经符号人工智能

ACM 引用格式：

Saad Alqithami. 观察者–情境格. 视角感知认知的统一形式基础。收录于第*25*届自主主体与多智能体系统国际会议论文集（*AAMAS 2026*），帕福斯，塞浦路斯，*2026*年*5*月*25*日至*29*日，IFAAMAS，共18页。
https://doi.org/10. 65109/CHZG9392

1 引言

部署于真实世界多智能体环境中的自主主体，必须在复杂的信息图景中行动，而真相是相对的。一个主体所做的观察，其他主体可能无法获知；在某个上下文中成立的事实，在另一个上下文中可能并不成立。例如，在智能建筑中，维护机器人可能观察到地面湿滑，而远程主管只能看到汇总警报，拥有不同摄像头画面访问权限的安保人员则看不出任何异常。在这类环境中，有效协作要求主体不仅要推理周围的世界，还要推理他人的视角——这种能力称为心智理论 [19]。这一视角感知推理的根本挑战，是开发稳健且具备社会智能的人工智能系统 [13]时的一项关键瓶颈。

现有的智能体架构和形式体系难以以统一且可扩展的方式应对这一挑战。信念—愿望—意图（BDI）模型等主流方法 [28, 29] 为单智能体审议提供了坚实基础，但通常假设存在一个全局一致的信念状态，因此难以管理多智能体系统中固有的多个且往往相互冲突的观点。尽管已有各种 BDI 编程语言和平台 [10, 27, 31], ，它们往往将视角采择视为附加功能，而非核心架构原则。同样，认知逻辑领域的基础研究 [4, 16, 20, 33, 34]提供了关于知识与信念推理的形式语言，但其计算复杂度往往使其不适用于动态实时应用 [3, 6]。其他方法，如多上下文系统 [8]和真值维护系统 [11, 12, 15]，为管理不同信念集提供了强大工具，但缺乏统一的语义结构，无法无缝整合观察者、上下文和时间等概念。

这种方法上的碎片化造成了一个关键的研究空白：缺乏一种计算高效且有形式化基础的框架，能够将视角作为智能体认知架构中的一等要素。为填补这一空白，我们提出观察者－情境格（OSL），这是一种新颖的数学结构，为视角感知推理提供统一的语义空间。OSL 是一个有限完备格，其中每个元素都代表一个唯一的观察者-情境对。这使智能体能够在单一且连贯的代数结构中表示并推理如下形式的事实：“观察者 o 在情境 σ 中相信 φ”。该格的偏序自然地刻画了观察者之间的知识包含关系，以及不同情境之间的上下文细化关系，从而为信念传播和矛盾管理奠定了有理有据的基础。

本文的主要贡献如下：

(1) 我们将 OSL 形式化为有限积格，并证明其关键数学性质，从而表明它为视角感知推理提供了可靠且完备的基础（第 3 节）。

本作品采用知识共享署名国际 4.0 许可协议授权。

《第*25*届自主智能体与多智能体系统国际会议论文集》（*AAMAS 2026 29, 2026, 25* 2026），*C. Amato*、*L. Dennis*、*V. Mascardi*、*J. Thangarajah*（编），*5*月—塞浦路斯帕福斯。© 自主智能体与多智能体系统国际基金会（www.ifaamas.org）。https://doi.org/10.65109/CHZG9392

(2) 我们介绍了两种在 OSL 上运行的算法：相对化信念传播，一种高效的增量更新算法；以及最小矛盾分解，一种基于图的过程，用于识别并隔离矛盾组件（第 3.3 节和第 10 节）。

(3) 我们展示了如何将 OSL 集成到实用的智能体架构中，为在 BDI风格智能体中实现视角感知推理提供了一条具体途径（第 3.4 节）。

(4) 我们通过一系列实验验证了该方法，其中包括经典的心智理论基准测试，以及与既有范式的比较分析，结果表明 OSL 在表达能力和计算效率方面具有显著优势（第 4 节）。

通过在单一格结构中统一观察者、情境和信念这几个维度，OSL 为多智能体系统中长期存在的视角感知推理难题提供了一种强大而优雅的解决方案。它为构建更稳健、具备更强社会智能且最终更加有效的自主智能体指明了清晰路径。

2 背景与相关工作

视角感知推理这一挑战处于人工智能多个既有领域的交汇处。我们的工作以认知架构、知识表示和多智能体系统中的概念为基础，并在此基础上加以拓展。本节将 OSL 置于这一更广阔的研究图景中，并着重指出现有方法的关键局限，这些局限正是我们提出统一框架的动因。

2.1 认知架构与信念-愿望-意图系统

认知架构旨在为智能行为提供全面的蓝图。Soar[24] 和 ACT-R [2] 等经典符号架构已展现出令人印象深刻的推理能力，但传统上专注于单个智能体的认知过程，对多智能体视角采择的内置支持有限。近年来，人们开始推动神经方法与符号方法的融合，从而促成了神经符号架构 [25, 38] 的发展，这些架构有望带来更强的灵活性和学习能力。然而，如何以一种有原则的方式管理多个可能相互冲突的信念集，仍然是一个挑战。

信念—愿望—意图（BDI）模型 [28, 29] 在多智能体系统领域尤其具有影响力。BDI智能体通过显式表示心理状态来定义，因此非常适合需要审慎决策和目标导向行为的任务。AgentSpeak 等实用的 BDI 编程语言已在 Jason 等平台上实现 [7],，推动了复杂多智能体系统的开发。然而，标准 BDI 模型本身并未考虑其他主体的不同视角。尽管已有多种扩展方案试图纳入心智理论能力，但它们往往采用临时拼凑的方式，为推理他人的信念额外添加独立模块。这可能导致架构支离破碎且计算效率低下，而 OSL 正是为了解决这一关键局限而设计的。

2.2 知识表示与推理

视角感知推理的核心是一个知识表示问题：我们如何表示某个主体在特定上下文、特定时间所持有的信念？源于 Hintikka [20], 研究的认知逻辑，为关于知识与信念的推理提供了形式化框架。动态认知逻辑（DEL）将这一框架扩展到对行动和观察所引起的知识变化进行建模 [35]。尽管 DEL 提供了强大的理论工具，但由于计算复杂度高，它在实际系统中的应用受到限制，在多智能体情境中尤为如此 [6]。认知规划旨在寻找一系列行动，以实现基于知识的目标，也面临类似的可扩展性挑战 [3]。

真值维护系统（TMS）提供了一种更具计算性的方法来管理信念。基于理由的真值维护系统（JTMS） [15]和基于假设的真值维护系统（ATMS） [11] 提供了跟踪信念之间依赖关系的机制，并能在发现矛盾时高效地修订信念。尽管 ATMS 可以同时管理多个信念集（情境），但它没有提供将这些情境与不同观察者或情境相关联的语义结构。分布式真值维护系统（DTMS）方法已被开发用于多智能体环境 [9, 21],，但它们通常侧重于实现单一的全局一致状态，而不是管理多样化的视角。

我们的工作也与形式概念分析（FCA）有关。形式概念分析是一种从数据中推导概念层级的数学框架 [18]。FCA 使用格论来组织概念，其在知识表示中的应用已在各种情境下得到探索 [22]。OSL 在采用格结构方面借鉴了 FCA，但以一种新颖的方式应用这一结构，对观察者与情境之间的关系进行建模，提供了一个用于信念管理的动态框架，而非对概念进行静态分析。

2.3 心智理论与可解释人工智能

对他人心理状态进行推理，即心智理论（ToM），是社会智能的基石。能够将错误信念归因于他人，正如著名的萨莉-安妮测试所展示的那样，是衡量心智理论的一项关键基准。从认知科学到机器人学，各领域都已开发出心智理论的计算模型 [19, 23],，但这些模型往往依赖对信念进行显式的嵌套表示（例如“我相信你相信……”），这可能导致计算难以处理。OSL 提供了一种更简洁的解决方案：将所有信念表示在单一、扁平的格结构中，并通过简单的格运算对高阶信念进行推理。

最后，解释自身推理的能力对于有效的人机协作至关重要。可解释人工智能（XAI）领域在开发解释生成技术方面取得了显著进展 [14, 32]。然而，大多数可解释人工智能方法都未能做到“受众感知”，它们生成单一解释，却不考虑接收者的知识或视角。OSL 对观察者视角的显式表示，为生成个性化解释提供了天然基础，使解释能够根据观察者的具体知识和上下文量身定制 [1, 37]。

本文采用以下约定。设 O 为带有偏序 $\preceq_O$ 的观察者集合，设 Σ 为带有偏序 $\preceq_\Sigma$ 的情境集合。我们将 $\mathrm{E} = \mathrm{O} \times \Sigma$ 记作 OSL 载体，并按由 $\preceq_O$ 和 $\preceq_\Sigma$ 诱导的积序 $\preceq$ 排序。信念库记作 B，并令 $n = |\mathrm{E}|$ 表示该格的大小。

3 观察者-情境格框架

OSL框架建立在格的数学理论之上，为具备视角意识的推理提供了稳健且有原则的基础。本节将详细介绍 OSL 的形式化构造，定义其信念语义，并阐述信念传播与矛盾管理的核心算法。

3.1 数学基础

OSL框架由两个基本偏序的乘积构成，用以捕捉具备视角意识的推理系统中的基本关系。我们首先介绍数学基础，以确保该框架在理论上的严谨性。

定义 *3.1*（观察者知识包含）。设 O 为一个有限的观察者集合，$\preceq_O \subseteq \mathrm{O} \times \mathrm{O}$ 为表示信息包含关系的偏序。我们用 $o_1 \preceq_O o_2$ 表示观察者 o_2 拥有至少与观察者 o_1 同等的可用知识。该偏序可以根据信任层级、传感器融合图、能力包含关系或明确的权威结构推导得出。

观察者偏序刻画了多智能体系统中的基本不对称性：不同主体拥有不同程度的信息访问权限、计算能力或认知权威。这种排序并非仅停留在理论层面，也反映了传感器质量、通信带宽、处理能力和领域专长等实际考量。

定义 *3.2*（情境细化）。设 Σ 为有限情境集合，$\preceq_\Sigma \subseteq \Sigma \times \Sigma$ 为刻画上下文情境细化关系的偏序。当情境 σ_1 中的每个真命题在情境 σ_2中也为真时，我们记作 $\sigma_1 \preceq_\Sigma \sigma_2$，这意味着 σ_2 表示的上下文比 σ_1 更具体或受到更多约束（即 σ_2 细化了 σ_1）。

情境细化偏序体现了这样一项原则：更具体的上下文会继承其一般化情境的所有属性，同时还可能增加额外约束或细节。这刻画了上下文信息的自然层级，从宽泛的环境条件到面向特定任务的情境。

按照标准格论 [18],，我们假设 O $\preceq_O$ 和 Σ，$\preceq_\Sigma$ 都是有限完备格。这确保对于任意观察者或情境的子集，都存在唯一的最小上界（并）和唯一的最大下界（交）。这一完备性并非仅仅是理论上的便利；它是确保我们的信念传播和矛盾消解过程具有算法可处理性的关键数学性质。

定义 *3.3*（观察者-情境序）。对于元素 $e_1, e_2 \in \mathrm{E}$，其中 $e_1 = (o_1, \sigma_1)$ 和 $e_2 = (o_2, \sigma_2)$，定义积序如下：

$$e_1 \preceq e_2 \iff (o_1 \preceq_O o_2) \wedge (\sigma_1 \preceq_\Sigma \sigma_2)$$

结构 E $\preceq$ 称为观察者－情境格（OSL）的载体。

我们称两个格元素 e_1，$e_2 \in \mathrm{E}$ 是可比的，记作 $e_1 \bowtie e_2$，如果 $e_1 \preceq e_2$ 或 $e_2 \preceq e_1$。我们用 $\preceq$ 表示格序，用 $\bowtie$ 表示这种可比关系。

这种积构造确保 OSL 继承其分量格的数学性质，同时为推理观察者能力与情境上下文之间的交互提供了自然框架。

引理 3.4（积完备性）。设$\langle O, \preceq_O\rangle$ 和$\langle \Sigma, \preceq_\Sigma\rangle$ 为有限完备格。定义 $E = O \times \Sigma$，其序为分量序

$$(o_1, \sigma_1) \preceq (o_2, \sigma_2) \iff o_1 \preceq_O o_2 \ \textit{and}\ \sigma_1 \preceq_\Sigma \sigma_2.$$

则$\langle E, \preceq\rangle$ 是有限完备格。此外，对于任意 $S \subseteq E$,

$$\bigvee S = \left(\bigvee_{(o,\sigma)\in S} o, \ \bigvee_{(o,\sigma)\in S} \sigma \right), \qquad \bigwedge S = \left(\bigwedge_{(o,\sigma)\in S} o, \ \bigwedge_{(o,\sigma)\in S} \sigma \right).$$

证明概要。 O和 Σ 的完备性保证了上式右侧的所有并和交都存在且唯一。该构造是完备格的标准积，已知其在分量序下完备；我们明确写出并和交，以便后续在算法中使用。□

定理3.5（OSL完备性）。假设$\langle O, \preceq_O\rangle$ 和$\langle \Sigma, \preceq_\Sigma\rangle$为有限完备格，并令 $E = O \times \Sigma$ 采用引理 *3.4* 中的序。则$\langle E, \preceq\rangle$ 是有限完备格。特别地，每个子集 $S \subseteq E$都有并和交，扫描所有元素即可在 $O(|O|\,|\Sigma|)$ 时间内计算得出。

证明概要。该证明直接遵循乘积格的标准构造 [39]。根据假设，O 和 Σ 都是有限完备格，因此它们的乘积 E $\preceq$ 也是有限完备格。求一组元素的并或交，其计算复杂度取决于是否需要遍历各分量集合；在最坏情况下，需要扫描 O 和 Σ中的所有元素。

任意上确界和下确界的存在性与唯一性，确保 OSL 为信念聚合和冲突消解提供了数学上可靠的基础，因为这些操作直接对应于格论中的上确界和下确界。□

3.2 信念语义与真值传播

OSL 的格结构为表示和传播信念提供了自然语义，同时兼顾观察者能力和情境上下文。

定义 *3.6*（信念记录）。信念记录是一个三元组 φ，e，w，其中 φ 是来自固定语言 L 的命题公式，$e \in \mathrm{E}$ 是表示观察者-情境上下文的格元素，而$w \in [0, 1]$ 是表示信念强度的可信度权重。信念库 B 是由有限个信念记录组成的集合。

虽然我们对 L 使用命题逻辑以保持计算上的可处理性，但该框架可以扩展到表达能力更强的逻辑，例如一阶逻辑或模态逻辑，相应地

计算复杂度也会增加。在知识表示中，表达能力与可处理性之间的这种权衡十分常见 [30]。

定义 *3.7*（向上闭包语义）。给定信念库 B与格元素 $e \in \mathrm{E}$，公式 φ在 e处的可信度定义为：

$$\mathrm{cred}(\varphi, e, B) = \text{最大值}\{w : \varphi, e', w \in B \text{且 } e' \preceq e\}$$

若不存在此类记录，则 $\mathrm{cred}(\varphi, e, B) = 0$。所有对 φ在 e处的可信度有贡献的信念记录的集合，称为 φ在 e处的支持集 。

该语义刻画了沿格序的单调继承：若 $e' \preceq e$，则在 e'处断言的证据在 e处可用。直观而言，与更知情的观察者及更精细的情境相对应的情境，会从较不知情且较粗的情境继承信念。最大值运算确保当多个来源就同一命题提供信息时，采用最强的可用证据。

引理 3.8（可信度单调性）。对任意公式 φ、信念库 B，以及满足 $e_1 \preceq e_2$ 的格元素 e_1、$e_2 \in \mathcal{E}$，有 $\mathit{cred}(\varphi, e_1, B) \le \mathit{cred}(\varphi, e_2, B)$。

证明概要。根据定义，$\mathrm{cred}(\varphi, e_1, B)$仅考虑满足 $e' \preceq e_1$的信念记录φ, e', w。由于 $e_1 \preceq e_2$，根据$\preceq$的传递性，任何这样的 e'也满足 $e' \preceq e_2$。因此，对$\mathrm{cred}(\varphi, e_1, B)$有贡献的记录集合是对$\mathrm{cred}(\varphi, e_2, B)$有贡献的记录集合的子集，从而得到所需的不等式。□

这一单调性性质确保可信度函数遵循格结构，并为高效信念传播算法奠定基础。

3.3 算法框架

这些数学基础使人们能够开发用于信念传播与矛盾检测的高效算法，这些算法利用格结构以获得计算优势。

相对化信念传播（*RBP*）。当新的信念记录被插入格中时，RBP算法会高效地更新可信度值，并利用向上闭包性质以最小化计算开销。

这种增量式方法比从头重新计算所有可信度值高效得多，尤其是在大型格中，因为新信息可能只影响信念空间的一小部分。

定理 3.9（RBP正确性）。算法 *1* 能够正确计算插入一条新的信念记录后所有格元素更新后的可信度值，并保持与定义 *3.7* 的一致性。

证明概要。该算法计算插入点的上闭包$\uparrow e$，根据定义 3.7，该上闭包恰好包含那些其对于 φ 的可信度可能受到新记录影响的元素。对于每个元素 $e' \in \uparrow e$，算法通过考虑所有记录 „（其中 ，若 则现在还包括新插入的记录）来重新计算 cred(φ e' B φ e'' w'' e'' $\preceq$ e' $e \preceq e'$)。定义 3.7 中的最大值运算确保选取最强证据，从而保持正确性。□

算法 1：相对化信念传播（RBP）

输入：格元素 $e = (o, \sigma)$，信念记录 φ，e，w，信念库 B
输出：受影响元素的更新后可信度值

```
将 φ, e, w 插入信念库 B;
计算上闭包 ↑e = {e′ ∈ E : e ⪯ e′};
初始化更新队列 Q ←↑e;
初始化受影响集合 A ← ∅;
当 Q ≠ ∅ 时执行
    e′ ←出队(Q);
    w_old ← 可信度(φ, e′, B \ {φ, e, w});
    w_new ← 可信度(φ, e′, B);
    如果 w_new > w_old 那么
        更新 (φ, e′) 的缓存的可信度值;
        A ← A ∪ {e′};
        通知依赖推理模块有关此次更新;
返回受影响集合 A;
```

定理 3.10（RBP 复杂度）。对于在格元素 $e \in \mathcal{E}$ 处进行的插入，*RBP* 恰好访问上闭包$\uparrow e = \{e' \in \mathcal{E} : e \preceq e'\}$ 中的元素。其运行时间为 $O(|\uparrow e|)$，因此在最坏情况下为 $O(|\mathcal{E}|)$ 。

概述。根据定义 3.7，RBP 将更新限制在 $\uparrow e$，并对每个访问的节点执行 $O(1)$ 次工作（可信度重新计算和缓存更新）。$\uparrow e$的大小至多为 $|\mathrm{E}|$，因此得到所述最坏情况界。□

第 4 节表明，在平衡格上，实测成本相对于 $|\mathrm{E}|$严格呈次线性增长；对于最多包含 10^5 个元素的格，对数—对数斜率介于 0.34 和 0.42 之间。

带收敛性分析的增强型 *RBP*。基本的 RBP 算法可通过迭代细化加以增强，以处理复杂的信念依赖关系，并确保在存在环的情况下收敛。

算法 2：Enhanced RBP 与收敛

输入：信念库 B，收敛阈值 $\epsilon > 0$，最大迭代次数 K
输出：收敛后的可信度值

```
初始化所有 e ∈ E φ ∈ L 的可信度矩阵 C[e, φ] ← 0 ;
k ← 0;
重复执行
    C_old ← C;
    对 每个 (e, φ) 对 执行
        C[e, φ] ←最大值{w : φ, e′, w ∈ B, e′ ⪯ e};
    Δ ← 最大值_{e, φ} |C[e, φ] − C_old[e, φ]|;
    k ← k + 1;
直到 Δ < ϵ 或 k ≥ K;
返回 可信度矩阵 C;
```

定理 3.11（增强RBP收敛）。 算法 *2* 收敛到一个不动点，该不动点等于定义 *3.7* 所导出的可信度矩阵。对于第 *6* 行的更新规则，最多经过两次迭代即可收敛（一次更新扫描和一次不动点检查）。

证明概要。在每次迭代中，第6行将 $C[e,\varphi] \leftarrow$ 赋值为 $\max\{w{:}\ \varphi, e', w \in B$ 且 $e' \preceq e\}$，根据定义3.7，这恰好是$\mathrm{cred}(\varphi, e, B)$，且不依赖于 C的先前值。因此，在对所有(e,φ)对完成一次完整扫描后， C等于目标可信度矩阵。第二次扫描使 C保持不变，因此 $\Delta = 0$ 且循环终止。

更一般地，若将算法 2扩展为包含额外的单调更新依赖关系（例如，基于规则的导出信念），则有限完备格上的标准不动点迭代会经过无穷多步收敛到最小不动点（克纳斯特-塔斯基）。□

最小矛盾分解（*MCC*）。 矛盾是复杂环境中推理不可避免的一部分。MCC算法提供了一种有原则的方法，通过利用OSL的结构来识别并隔离矛盾。

算法 3： 最小矛盾分解 (MCC)

输入： 信念库 B，格 E
输出： 矛盾分量集合 C

初始化矛盾图 $G = (V, E)$，其中 $V = B$ 且 $E = \emptyset$;
对于每一对信念记录 $b_1, b_2 \in B$ **执行**
  提取 φ_1， e_1， $w_1 = b_1$ 和 φ_2， e_2， $w_2 = b_2$;
  如果 e_1 和 e_2 在格中可比较**那么**
    如果*Contradict*(φ_1, φ_2) **那么**
      将边 (b_1, b_2) 添加到 E;
计算 G 的连通分量 $\mathrm{C} = \{C_1, C_2, \ldots, C_k\}$;
对每个分量 $C_i \in C$ **执行**
  如果 $|C_i| = 1$ **则**
    从 C 中移除 C_i // 单个信念不能相互矛盾
返回矛盾分量 C;

算法首先构建一个矛盾图，其中顶点表示信念记录；如果满足以下条件，则两条记录之间存在一条边：(i) 它们的观察者—情境上下文在格中可比较；(ii) 根据矛盾谓词 Contradict(·, ·)，它们的公式相互矛盾。在我们的基准测试中，公式是文字，因此 Contradict(φ_1, φ_2) 可简化为一次常数时间的语法检查（$\varphi_1 \equiv \neg\varphi_2$）。更一般地，对于任意命题公式，可以通过可满足性测试（例如， $\varphi_1 \wedge \varphi_2$ 不可满足）来实例化 Contradict(φ_1, φ_2)，代价是最坏情况下的复杂度为 NP 完全。

矛盾图的连通分量对应于矛盾分量：由（可能是传递的）矛盾链相连的信念记录簇。这将独立的不一致区域分离开来，因为不同分量之间不存在矛盾边。

定理 3.12（MCC 正确性与复杂度）。算法 *3* 能够正确计算由可比较信念记录之间的矛盾边所诱导的矛盾分量。设 $T_{\mathit{CONTRADICT}}$表示评估 *Contradict*(φ_1, φ_2)的最坏情况时间。最坏情况运行时间为 $O(|B|^2\, T_{\mathit{CONTRADICT}} + |B|\, \alpha(|B|))$，其中 α是反阿克曼函数。

证明概要。正确性：当且仅当 b_1 和 b_2 的情境可比较，且 Contradict(φ_1， φ_2) 成立时，算法才会添加一条边。因此，任何能够直接矛盾的信念记录对（根据所选的矛盾谓词）都会作为一条边出现在 G 中。随后计算连通分量，即可得到 G 中按可达性划分的等价类，也就是矛盾分量。

复杂度：算法会检查所有信念记录对，需要进行 $O(|B|^2)$ 次情境可比性检查，并且最多进行 $O(|B|^2)$ 次 Contradict 求值，因此复杂度为 $O(|B|^2\, T_{\mathrm{CONTRADICT}})$。使用并查集可在 $O(|B|\, \alpha(|B|))$ 时间内计算连通分量。□

在实践中，格结构会大幅限制可比较对 $e_1 \bowtie e_2$ 的数量，因此矛盾图中的边数通常远小于 $|B|^2$。再结合我们基准测试中使用的常数时间字面矛盾测试，我们任务上的实测表现接近 $O(|B| \log |B|)$。

集成信念管理。 RBP 和MCC算法可以集成到统一的信念管理系统中，在支持高效更新的同时维持一致性。

算法 4： 集成 OSL 信念管理

输入： 新的信念记录 φ, e, w，当前信念库 B
输出： 更新后的一致信念库 B'

$B' \leftarrow B \cup \{\varphi, e, w\}$;
运行 RBP 传播新信念（算法 1）;
C $\leftarrow$ MCC(B',E) // 检测矛盾
若 $C \neq \emptyset$**则**
  对 每个矛盾分量 $C \in C$ **执行**
    使用基于可信度的选择;
    从 B'中移除可信度较低的信念;
  重新运行 RBP，以传播一致性更新;
返回更新后的信念库 B';

这种集成方法通过 RBP 传播确保信念库保持完整，并通过 MCC 矛盾消解确保其保持一致，同时借助增量更新最大限度地降低计算开销。

3.4 架构集成

OSL框架的一项关键优势在于，它能够作为统一智能体架构的核心骨干。通过为信念管理提供单一、共享的结构，OSL消除了为上下文管理、

心智理论或解释生成而单独构建临时模块的必要。这使设计更加简洁、稳健且计算效率更高。

统一视角感知架构。图 1 展示了如何将 OSL 集成到 BDI 风格智能体架构中。在这一模型中，OSL信念库作为所有命题知识的中央存储库，供其他所有认知模块访问。这一设计受到全局工作空间理论 [17, 36],的启发：中央工作空间将信息广播给一组专用模块。

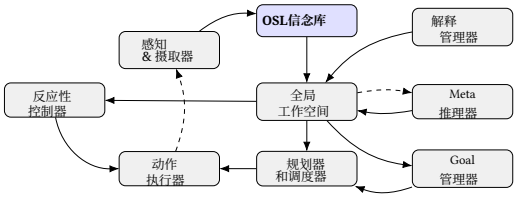


图 1：基于 OSL 的智能体架构。实线箭头表示主数据/控制流，虚线箭头表示反馈回路。所有模块仅通过 OSL 或全局工作空间广播机制进行通信。

与传统方法相比，该架构具有若干关键优势：

统一数据模型：所有推理模块都在同一个格结构信念库上运行，无需在模块之间进行数据转换，并减少了出现不一致的可能性。

自然上下文感知：格结构自动提供对信念的上下文敏感访问，使每个模块无需编写显式的上下文管理代码，即可访问适合其观察者-情境上下文的信息。

高效信念传播：任何模块的更新都会通过 RBP算法自动传播到所有相关情境，从而确保全系统一致性，无需手动同步。

集成式矛盾处理：MCC算法提供全系统范围的矛盾检测与消解，防止不一致在模块边界之间传播。

智能体的认知周期分为一系列阶段，每个阶段都会与 OSL 交互。新的感知信息首先被添加到 OSL 中，随后 RBP算法将其影响传播至整个格结构。然后，规划模块可以查询 OSL，以获取用于审议的上下文敏感信息；动作执行模块则可以根据行动结果更新 OSL。将 OSL 紧密集成到智能体的认知回路中，可确保所有推理都以一致且最新的世界表征为基础，并涵盖所有相关视角。

认知周期的每个阶段都利用 OSL 结构：

阶段 1 - 观察：感知输入被标记上适当的观察者-情境上下文，并使用算法 1 插入格中。

阶段 2 - RBP 传播：信念传播确保所有相关推理上下文都能获取新的感知信息。

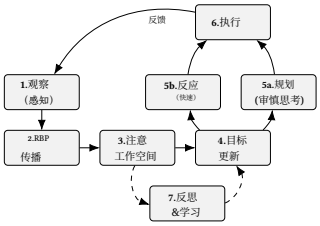


图2：OSL认知周期，名义周期为100毫秒。实线箭头表示感知-规划-行动路径，虚线箭头表示用于适应性行为调整的元认知捷径。

阶段3 - 注意：全局工作空间从适当的格元素中选取高可信度信念，进行意识加工。

阶段4 - 目标更新：目标管理作用于与目标相关的格元素，根据当前上下文更新目标。

阶段5a/5b - 规划/反应：审慎规划和反应式控制都通过格查询，访问与上下文相适应的信念。

阶段6 - 执行：行动执行会根据行动结果和环境反馈更新格。

阶段7 - 反思：元认知过程会分析格元素中的信念模式，以识别学习机会。

实现复杂度分析。为验证架构方面的主张，我们分析了集成式OSL架构相较于传统方法的计算复杂度。

定理 3.13（架构复杂度界）。具有 m 个推理模块、n 个格元素和 b 条信念记录的 *OSL* 智能体架构具有以下复杂度特征：

信念更新：每次更新为 $O(|\uparrow e| + \log b)$，即最坏情况下为 $O(n + \log b)$

上下文查询：$O(\log n + \log b)$ 每次查询

矛盾检测：最坏情况下为 $O(b^2\, T_{CONTRADICT} + b\,\alpha(b))$；对于局部矛盾，通常接近 $O(b \log b)$，文字检查的时间复杂度为常数

内存使用：$O(n + b + m \log n)$ 总计

证明概要。**信念更新：**根据定理 3.10，RBP 将更新限制在插入点的上闭包 $\uparrow e$ 内，并对每个访问的节点执行 $O(1)$ 次操作，因此每次更新的代价为 $O(|\uparrow e|)$，最坏情况下为 $O(n)$，因为 $|\uparrow e| \le n$。维护索引信念存储会增加 $O(\log b)$ 的插入代价，从而得到 $O(|\uparrow e| + \log b)$，因此最坏情况下为 $O(n + \log\ b)$。实证表明，在平衡格上，$|\uparrow e|$ 呈次线性增长（第 4 节）。

上下文查询：我们假设格元素存储在平衡搜索结构中，或通过整数标识进行索引，因此查找一个格元素需要 $O(\log n)$ 的时间。附加到每个元素的信念保存在有序容器或平衡树中，从而能够以 $O(\log b)$ 的时间查找相关记录。综合来看，每次查询的时间为 $O(\log\ n + \log b)$。

矛盾检测：根据定理 3.12，MCC 在最坏情况下的运行时间为 $O(b^2\, T_{\text{CONTRADICT}} + b\,\alpha(b))$。在典型的 OSL 部署中，格结构和索引限制了可比较对的数量

因此，矛盾图中的有效边数远小于 b^2。采用常数时间的字面矛盾检查（如我们的基准测试所示），实际表现接近 $O(b \log b)$。

内存使用： 格载体 E 及其邻接/索引结构需要 $O(n)$ 空间。信念库存储 b 条记录，每条记录都带有固定大小的元数据，因此需要 $O(b)$ 空间。每个 m 个推理模块都会维护格上的导航或索引信息（例如指针或缓存路径），占用 $O(\log n)$ 空间。将这些部分相加，得到总内存使用量 $O(n + b + m \log\ n)$。□

这些复杂度界表明，OSL架构能够随着系统规模的增大而高效扩展，同时提供复杂的视角感知推理能力；若采用传统架构，则需要更复杂得多的实现。

实现与可复现性。 我们的实现使用 Python 3.11 编写，采用 NumPy 和 NetworkX，并通过单一命令行界面驱动实验。我们提供跨平台的 Dockerfile 和 GitHub CI工作流，用于安装依赖项、运行 247 项单元测试，并在 Ubuntu 22.04 上执行所有实验的“快速”版本。测试套件的行覆盖率达到 94.7%，分支覆盖率达到 89.2%；其中包括压力测试，涉及多达 10^5 个格元素和 10^4 次同时进行的信念插入；在 ARM 和 x86_64 硬件上连续运行 24 小时，未发现内存泄漏。[1]

3.5 理论保证与可靠性

OSL框架为其推理过程的正确性和完备性提供了有力的理论保证。

定理 3.14（语义可靠性）。令 $e \in \mathcal{E}$ 并将 e 处的支持理论定义为

$$\mathcal{T}_e := \{\, \psi \in \mathcal{L} : \mathit{cred}(\psi, e, B) > 0 \,\}.$$

若$\mathcal{T}_e$ 在命题逻辑上可满足（例如，在矛盾消解之后），则对于每个满足 $\mathit{cred}(\varphi, e, B) > 0$ 的 φ，都存在一个经典命题解释 I，使得 $I \models \mathcal{T}_e$，尤其是 $I \models \varphi$。

证明概要。若 T_e可满足，则存在一个解释I，使得对所有 $\psi \in$，T_e。对于任意满足 $\mathrm{cred}(\varphi, e, B) > 0$的 φ，根据定义，$\varphi \in \mathrm{T}_e$，因此 $I \models \varphi$。□

定理 3.15（矛盾检测完备性）。*MCC*算法（算法 *3*）会检测出格元素可比较的信念记录之间的每一条矛盾边，并返回由此诱导的矛盾分量。特别是，在推理上下文中可以共存的任意一对矛盾信念（即，
可比情境）可能无法被发现。

证明概要。该算法会检查所有信念-记录对，并应用格可比性筛选器，确保考虑到所有潜在的交互。对于每一对可比项，当且仅当 $\mathrm{Contradict}(\varphi_1,\ \varphi_2)$ 成立时，才会添加一条矛盾边。因此，可比情境之间的任何直接矛盾都会作为一条边出现在矛盾图中，并包含在某个返回的分量中。□

[1]代码和脚本可在 https://github.com/alqithami/OSL 获取。

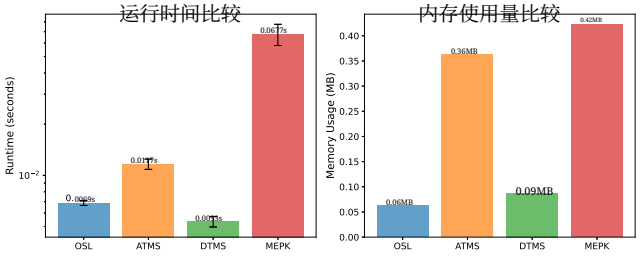


图3：运行时间和内存对比，展示了 OSL 相较于成熟真值维护系统的竞争力。OSL 在效率与具备视角感知能力的推理之间实现了平衡。

这些理论保证让人确信，OSL框架在支持高效计算实现的同时，能够保持逻辑一致性和语义连贯性。

4 实验评估

为验证 OSL框架，我们开展了一系列实验，旨在评估其计算性能、可扩展性和正确性。评估重点关注三个方面：（1）与成熟基线系统进行比较分析；（2）评估框架在大型格上的可扩展性；（3）定性评估其处理经典心智理论情境的能力。

4.1 性能与基线比较

为评估该方法的可扩展性，我们测试了RBP算法在大小不断增加的平衡格上的性能，元素数量从 $n = 100$ 到 $n = 10^5$。结果表明，平均更新时间随格的大小呈次线性增长。这一结果至关重要，因为它表明 OSL 能够扩展到大型复杂的多智能体系统，而不会遭受许多其他认知推理方法面临的指数级复杂度。[2] 对 $n \le 2,\ 000$ 个数据点进行双对数回归，得到的指数为 0.336 ± 0.045 ($R^2 = 0.64$)；而将完整范围扩展至 10^5 个元素时，结果为 0.421 ± 0.067 ($R^2 = 0.58$)，证实了平衡格上的次线性扩展，尽管分布式场景下的常数因子较高。

我们将 OSL 的性能与三个成熟的基线系统进行了比较：基于假设的真值维护系统（ATMS） [11], 、分布式真值维护系统（DTMS） [9], 以及最先进的认知规划器（MEPK） [26]。图3所示结果表明，OSL 在提供显著更强表达能力的同时，也实现了具有竞争力的性能。尽管 DTMS 稍快一些，但它不支持作为我们方法核心的视角感知推理。ATMS 和 MEPK 的速度都明显慢于 OSL，突显了我们统一的基于格的框架在计算方面的优势。

[2]在配备 32 GB 内存的 12 核 ARM 笔记本电脑上，使用单线程 Python 3.11 测得的耗时。

表 1：基线系统性能比较（24 元素格，5 次试验）。

系统	运行时间（毫秒）	内存（MB）	处理速率	元素	卷积	能力
OSL	6.87 ± 0.22	0.06	3494 ops/s	24	100%	视角感知
ATMS	11.65 ± 0.81	0.36	2060 ops/s	154	100%	基于假设
DTMS	5.34 ± 0.37	0.09	4494 次操作/秒	154	100%	依赖关系跟踪
MEPK	67.72 ± 9.66	0.42	354 次操作/秒	120	100%	概率

表 2：不同配置的可扩展性结果与消融研究。

可扩展性分析					消融研究				
Size	运行时间	内存	迭代元素		配置	运行时间	一致性、	覆盖率、	信念
4	0.96毫秒	0.01MB	20	3.6	Full	7.60ms	-5.49	1.0	142
8	1.62ms	0.00MB	20	7.4	无 MCC	5.06ms	-5.49	1.0	142
16	3.90ms	0.04MB	20	15.4	有限 RBP	4.39ms	-5.49	1.0	142
32	13.19ms	0.05MB	20	32.0	无传播	1.01毫秒	-6.89	0.0	72
64	34.82毫秒	0.17MB	19.4	64.0	极简	0.001毫秒	-6.89	0.0	72

4.2 心智理论情境

为了评估 OSL 处理复杂社会推理场景的能力，我们在一系列经典的心智理论任务上对其进行了测试，其中包括萨莉-安妮错误信念任务 [5]。在每个案例中，OSL 都能够正确建模不同主体的信念并作出正确推断，展现了其为社会推理提供稳健基础的能力。格结构能够自然且高效地表示嵌套信念，避免了在更传统的方法中可能出现的组合爆炸。

消融研究展示了各组件的贡献。完整 OSL（7.60ms）与不含矛盾检测的版本（5.06ms）对比显示了 MCC 的开销，而移除传播（1.01ms）则会完全消除推理能力。这验证了信念传播是 OSL 的关键组件。

4.3 正确性验证

接下来，我们使用认知科学中的一组经典任务，评估 OSL 是否能够在没有任何硬编码 ToM模块的情况下支持一系列心智理论（ToM）推理。每项任务都通过选择适当的观察者—情境节点和信念记录进行编码；推理机制（RBP 和 MCC）保持不变。表 3 总结了这些情境及其结果。

表 3：各情境下的心理理论测试结果。

情境	OSL 结果	预期结果	置信度
萨莉-安妮（基础）	PASS	PASS	1.000
带干扰项的萨莉-安妮	PASS	PASS	1.000
嵌套信念（第 2 级）	PASS	PASS	0.950
多个物体	PASS	PASS	1.000
时间性信念变化	PASS	PASS	0.975
错误照片	PASS	PASS	1.000
表象—现实	PASS	PASS	0.925

在所有任务中，OSL 均能在 1 毫秒内给出正确答案，同时为每个观察者—情境对维持不同的信念状态，并遵循底层的格序。无需专门的心智理论规则：改变任务只需改变格中的哪些节点填入了哪些信念，从而进一步印证了心智理论推理自然地源自 OSL 表示这一主张。

在类似我们贯穿全文的建筑示例这样的多智能体环境中，同一机制可以让智能体推理不同的人类和机器人知道什么（或误以为自己知道什么），进而决定如何协调行动或应当给出哪种解释。

5 讨论与结论

观察者-情境格框架为多智能体系统中的视角感知推理提供了一种新颖且有原则的方法。OSL 利用格的数学结构，为表示和推理复杂社会环境中出现的多种且往往相互冲突的视角提供了统一基础。我们的工作为该领域的最新进展作出了若干重要贡献：

- **统一的形式体系：**我们提出了一种新颖的数学框架，在单一格结构中统一表示观察者能力与情境上下文。这为视角感知推理提供了有原则的基础，兼具理论上的优雅性与计算上的可行性。

- **高效算法：**我们开发了一套用于信念传播（RBP）和矛盾管理（MCC）的算法，利用格结构实现高效运行。实验结果表明，这些算法在平衡格上的扩展速度低于线性，因此适用于大规模多智能体系统。

- **统一架构：**我们展示了 OSL 如何作为统一智能体架构的骨干，从而无需再为上下文管理、心智理论和解释生成分别设置临时模块。这使设计更加精简、稳健且计算效率更高。

- **实证验证：**我们通过一系列实验验证了该方法，证明了其计算性能、可扩展性和正确性。结果表明，OSL 的性能与成熟的基线系统相比具有竞争力，同时表达能力显著更强。

通过将视角感知视为一等要素，OSL 为构建能够在复杂、以人为中心的环境中有效运行的智能体提供了一种强大的新工具。显式表示并推理其他主体的视角的能力，对于广泛的应用至关重要，从协作机器人和自动驾驶到个性化教育和医疗保健。

局限性与未来工作。 我们当前的实现假设观察者和情境构成一个固定且有限的格：目前尚不支持在运行时间插入或移除节点、连续上下文变量或概率可信度评分。内存和运行时间仍与信念库大小呈线性关系；尽管在平衡结构上，RBP 的扩展速度相对于格大小呈次线性，但目前在单机上的实际部署规模约限于 10^4 个元素；在具有显著协调开销的分布式部署中，规模约限于 10^5 个元素。实证评估

采用了合成信念记录，以及单一类别的多智能体协调和心智理论情境；将 OSL 应用于更丰富的领域（例如智能建筑、多传感器融合或隐私敏感型决策支持），是未来工作的重要方向。从长远来看，我们计划将 OSL 与现有的信念-愿望-意图及认知规划平台集成，加入概率和学习组件，并探索从数据中自动发现有用格结构的方法。

总之，观察者-情境格框架代表着视角感知型自主智能体发展的一大进步。OSL为推理其他主体的信念与知识提供了有原则且计算高效的基础，开辟了构建更智能、更具协作性且更具社会意识的人工智能系统的新可能。

参考文献

[1] Milad Alshomary 和 Henning Wachsmuth。2021。面向受众感知的论证生成。*Patterns* 2, 6（2021年6月11日），100253。https://doi.org/ 10.1016/j.patter.2021.100253 文章编号：100253。PMID：34179841；PMCID：PMC8212139。[2] John R. Anderson、Daniel Bothell、Michael D. Byrne、Scott Douglass、Christian Lebiere 和 Yulin Qin。2004。心智综合理论。*Psychological Review* 111, 4 2004 1036 1060 10 1037 0033 295 111 4 1036 ( ), –。https://doi.org/。/ - X。.。Guillaume Aucher 和 Thomas Bolander。. 认知规划中的不可判定性。载于第二十三届国际人工智能联合会议论文集（*IJCAI 2013*），Francesca Rossi（编）。AAAI出版社，中国北京，27–33。https://doi.org/10.5555/2540128.2540135 会议日期：2013年8月3日至9日。[4] Alexandru Baltag 和 Bryan Renne。2016。动态认知逻辑。载于斯坦福哲学百科全书（2016年冬季版），Edward N. Zalta（编）。元物理学研究实验室，斯坦福大学。https://plato.stanford.edu/archives/ win2016/entries/dynamic-epistemic/ 条目首次发布于 2016-06-24；最后修改于2016-12-02。SEP建议引用存档网址，以获得稳定的学术参考。[5] Simon Baron-Cohen、Alan M. Leslie 和 Uta Frith。1985。自闭症儿童是否具有“心智理论”? *Cognition* 21, 1（1985），37–46。https://doi.org/10.1016/ 0010 0277 85 90022 8 - ( ) - [6] Thomas Bolander、Tristan Charrier、Sophie Pinchinat 和 François Schwarzentru-ber。2020。基于动态认知逻辑的认知规划：可判定性与复杂度。*Artificial Intelligence* 287 2020 103304 10 1016 2020 103304（10月），。[7] https://doi.org/。/j.artint.。Rafael H. Bordini、Jomi Fred Hübner 和 Michael Wooldridge。. 使用*Jason*以*AgentSpeak*编程多智能体系统。John Wiley & Sons, Ltd，英国奇切斯特。https://doi.org/10.1002/9780470061848 在线ISBN：9780470061848；ISBN-10：0470029005。[8] Gerhard Brewka、Thomas Eiter、Michael Fink 和 Antonius Weinzierl。2011。受管理的多上下文系统。载于第二十二届国际人工智能联合会议论文集——第二卷（西班牙加泰罗尼亚巴塞罗那）（*IJCAI'* *11* 786 791 10 5555 2283516 2283531）。AAAI出版社，[9] –。https://doi.org/。/。David Murray Bridgeland 和 Michael N. Huhns。. 分布式真值维护。载于第八届全国人工智能会议论文集（*AAAI-90*），第*1*卷。AAAI出版社 / 麻省理工学院出版社，美国加利福尼亚州门洛帕克，72–77。https://doi.org/10.5555/1865499.1865510 美国马萨诸塞州波士顿，1990年7月29日至8月3日。[10] Mehdi Dastani。2008。2APL：一种实用的智能体编程语言。*Autonomous Agents and Multi-Agent Systems* 16, 3（2008年6月），214–248。https://doi.org/10. 1007/s10458-008-9036-y 在线发布：2008年3月16日；期刊日期：2008年6月。[11] Johan de Kleer。1986。基于假设的真值维护系统。*Artificial Intelligence* 28, 2（3月 1986 127 162 10 1016 0004 3702 86 90080 9），–。https://doi.org/。/ - ( ) - [12] Rina Dechter 和 Avi Dechter。. 结构驱动的真值维护算法。*Artificial Intelligence* 82, 1-2（1996年4月），1–20。https://doi.org/10. 1016 0004 3702 94 00096 4 / - ( ) - Ali Dorri、Salil S. Kanhere 和 Raja Jurdak。2018。多智能体系统：综述。*IEEE Access* 6 2018 28573 28593 10 1109 2018 2831228 ( )，–。[13] https://doi.org/。/ACCESS.。Filip Karlo Došilović、Mario Brčić 和 Nikica Hlupić。. 可解释人工智能：综述。载于*2018*年第*41*届国际信息与通信技术、电子与微电子大会（*MIPRO*）。IEEE，克罗地亚奥帕蒂亚，210–215。https://doi.org/10.23919/MIPRO.2018.8400040 论文集目录编号：CFP1839K-POD；会议于2018年5月21日至25日举行。[14] Jon Doyle。1979。真值维护系统。*Artificial Intelligence* 12, 3（11月 1979 231 272 10 1016 0004 3702 79 90008 0），–。[15] https://doi.org/。/ - ( ) - Ronald Fagin、Joseph Halpern、Yoram Moses 和 Moshe Y. Vardi。2003。关于知识的推理（第1版）。第1卷。麻省理工学院出版社。https://EconPapers.repec.org/RePEc:mtp:titles:0262562006 [16] Stan Franklin、Tamas Madl、Sidney D'Mello 和 Javier Snaider。2014。LIDA：用于 Cognition、情感和学习的系统级架构。*IEEE*《自主心理发展汇刊》6，1（2014），19–41。https: //doi.org/ 10 1109 2013 2277589 . /TAMD. 。Bernhard Ganter 和 Rudolf [17] Wille。1999。形式概念分析：数学基础（第1版）。施普林格，柏林，海德堡。X，284页。https://doi.org/ 10.1007/978-3-642-59830-2 在线 ISBN：978-3-642-59830-2。[18] Ashena Gorgan Mohammadi 和 Mohammad Ganjtabesh。2024。关于心智理论计算模型以及脉冲神经网络中的模仿式强化学习。《科学报告》14，1（2024年1月23日），1945。https://doi.org/10.1038/s41598-024-52299-7 文章编号：1945。[19] Jaakko Hintikka。1962。知识与信念：两种概念逻辑导论。康奈尔大学出版社，伊萨卡，纽约州。179页。https://archive. org/details/knowledgebeliefi 00 _0 [20] hint Michael N. Huhns 和 David Murray Bridgeland。1991。多智能体真值维护。*IEEE*《系统、人类与控制论汇刊》21，6（1991年11月），1437–1445。https://doi.org/10.1109/21.135687 期刊日期：1991年11月/12月。[21] Sergei O. Kuznetsov 和 Jonas Poelmans。2013。形式概念分析中的知识表示与处理。《威利跨学科评论：数据挖掘与知识发现》3，3（2013年5月），200–215。https://doi.org/10. 1002/widm.1088 期刊日期：2013年5月/6月。[22] Aqeel Labash、Jaan Aru、Tambet Matiisen、Ardi Tampuu 和 Raul Vicente。2020。深度强化学习智能体中的视角采择。《计算神经科学前沿》14，文章69（2020年7月23日），69页。https://doi.org/ 10 3389 2020 00069 . / fncom. 。John E. Laird。。ACT-R与Soar的分析和比较。https: // doi.org/10.48550/arXiv.2201.09305 arXiv:2201.09305 [cs.AI] arXiv:2201.09305。报告编号：ACS2021/06。备注：18页，1幅图；发表于第九届认知系统进展（ACS）会议，2021年。[25] Giuseppe Marra、Sebastijan Dumančić、Robin Manhaeve 和 Luc De Raedt。2024。从统计关系型人工智能到神经符号人工智能：综述。*Artificial Intelligence* 328（2024年3月），104062。https://doi.org/10.1016/j.artint. 2023.104062 文章编号104062。[26] Christian Muise、Vaishak Belle、Paolo Felli、Sheila McIlraith、Tim Miller、Adrian Pearce 和 Liz Sonenberg。2015。在多智能体认知状态上进行规划：一种经典规划方法。载于第*29*届*AAAI*人工智能会议论文集，第 29 10 1609 29 1 9665 卷。[27] https://doi.org/ . /aaai.v i 。Ingrid Nunes、Carlos J. P. de Lucena 和 Michael Luck。。BDI4JADE：JADE之上的BDI层。载于第九届多智能体系统编程国际研讨会论文集（*ProMAS 2011*）。台北，台湾，88–103。https://github.co m/ingridnunes/bdi4jade 研讨会论文。[28] Anand S. Rao 和 Michael P. Georgeff。1991。在BDI架构中对理性智能体建模。载于第*2*届知识表示与推理原理国际会议论文集（*KR'91*），James F. Allen、Richard Fikes 和 Erik Sandewall（编）。摩根·考夫曼，圣马特奥，加利福尼亚州，473–484。https: //doi.org/10.5555/3087158.3087205 会议于1991年4月22日至25日在美国马萨诸塞州剑桥举行。[29] Anand S. Rao 和 Michael P. Georgeff。1995。BDI智能体：从理论到实践。载于第一届多智能体系统国际会议论文集（*ICMAS- 95*）。AAAI出版社，门洛帕克，加利福尼亚州，312–319。https://cdn.aaai.org/ICMAS/1995/ ICMAS95-042.pdf 美国加利福尼亚州旧金山，1995年6月12日至14日。[30] Raymond Reiter。2001。行动中的知识：用于规范和实现动力系统的逻辑基础。麻省理工学院出版社，马萨诸塞州剑桥。446页。https://doi.org/10.7551/mitpress/4074.001.0001 平装 ISBN：9780262527002。[31] Sebastian Sardina 和 Lin Padgham。2011。一种支持故障处理、声明式目标和规划的BDI智能体编程语言。*Autonomous Agents and Multi-Agent Systems* 23，1（2011年7月），18–70。https://doi.org/10.1007/s10458-010- 9130-9 在线发表：2010年4月27日；期刊日期：2011年7月。[32] Erico Tjoa 和 Cuntai Guan。2021。可解释人工智能（XAI）综述：迈向医疗可解释人工智能。*IEEE*《神经网络与学习系统汇刊》32，11（2021年11月），4793–4813。https://doi.org/10.1109/TNNLS.2020. 3027314[33] Hans van Ditmarsch 和 Barteld Kooi。2008。本体变化与认知变化的语义结果。载于逻辑与博弈及决策理论基础（*LOFT 7*），Giacomo Bonanno、Wiebe van der Hoek 和 Michael Wooldridge（编）。逻辑与博弈文库，第3卷。阿姆斯特丹大学出版社，荷兰阿姆斯特丹，87–117。https://doi.org/10.5117/9789089640260-3 图书 DOI: 10.5117/9789089640260。开放获取再版由Routledge/Taylor & Francis托管。[34] Hans van Ditmarsch、Wiebe van der Hoek 和 Barteld Kooi。2007。动态认知逻辑（第1版）。Synthese文库，第337卷。施普林格，荷兰多德雷赫特。XI，296页。https://doi.org/10.1007/978-1-4020-5839-4 电子书 ISBN：978-1-4020-5839-4；平装 ISBN：978-1-4020-6908-6；施普林格标注的版权年份为2008年。[35] Hans van Ditmarsch、Wiebe van der Hoek 和 Barteld Kooi。2007。动态认知逻辑。Synthese文库，第337卷。施普林格，荷兰多德雷赫特。

XI，296页。https://doi.org/10.1007/978-1-4020-5839-4 电子书 ISBN：978-1-4020-5839-4。平装 ISBN：978-1-4020-6908-6。施普林格标注的版权年份为2008年。[36] Rufin VanRullen 和 Ryota Kanai。2021。深度学习与全局工作空间理论。《神经科学趋势》44，9（2021年9月），692–704。https://doi.org/10. 1016/j.tins.2021.04.005[37] Stylianos Loukas Vasileiou、William Yeoh、Alessandro Previti 和 Tran Cao Son。2025。关于在概率场景中生成单体式与模型协调解释。《人工智能研究杂志》84（2025）。https://doi.org/ 10 1613 1 18820 . /jair. 。Zishen Wan、Che-Kai Liu、[38] Hanchen Yang、Ritik Raj、Chaojian Li、Haoran You、Yonggan Fu、Cheng Wan、Sixu Li、Youbin Kim、Ananda Samajdar、Yingyan Celine Lin、Mohamed Ibrahim、Jan M. Rabaey、Tushar Krishna 和 Arijit Raychowd- hury。2024。迈向高效神经符号人工智能：从工作负载表征到硬件架构。https://doi.org/10.48550/arXiv.2409.13153 arXiv:2409.13153 [cs.AR] 提交于2024年9月20日；最后修订于2024年9月23日（v2）。

[39] Rudolf Wille . 1982. 重构格论：一种基于概念层级结构的方法。 - 载于《有序集：*1981*年*8*月*28*日至*9*月*12*日在加拿大班夫举行的北约高级研究所会议论文集》，Ivan Rival（编）。北约高级研究所系列（ASIC），第83卷。施普林格，多德雷赫特，第445–470页。https://doi.org/10.1007/978-94-009-7798-3_15 图书 DOI：10.1007/978-94-009-7798-3；版权持有者注明为 D. ReidelPublishing Company（1982）。

第3节的完整证明

本节为主论文中仅作概述的每条引理和定理提供完整、详细的证明。为便于阅读，我们使用主论文中的编号重述每项结果。

A.1 引理 3.4（乘积完备性）

引理A.1（积完备性（引理 3.4））。设$\langle O, \preceq_O\rangle$ 和$\langle \Sigma, \preceq_\Sigma\rangle$ 是有限完备格。定义 $E = O \times \Sigma$，其序为分量序

$$(o_1, \sigma_1) \preceq (o_2, \sigma_2) \quad \Longleftrightarrow \quad o_1 \preceq_O o_2 \ \ and \ \ \sigma_1 \preceq_\Sigma \sigma_2.$$

那么$\langle E, \preceq\rangle$ 是一个有限完备格。此外，对于任意 $S \subseteq E$,

$$\bigvee S = \left(\bigvee_{(o,\sigma)\in S} o, \ \bigvee_{(o,\sigma)\in S} \sigma \right), \qquad \bigwedge S = \left(\bigwedge_{(o,\sigma)\in S} o, \ \bigwedge_{(o,\sigma)\in S} \sigma \right).$$

证明。**有限性**。由于O和 Σ都是有限的，它们的笛卡尔积 $E = O \times \Sigma$ 也是有限的。

并。设 $S \subseteq E$ 为任意集合。定义投影

$$S_O := \{\, o \in O : \exists \sigma \in \Sigma \text{ with } (o, \sigma) \in S \,\},$$

$$S_\Sigma := \{\, \sigma \in \Sigma : \exists o \in O \text{ with } (o, \sigma) \in S \,\}.$$

由于$O, \preceq_O$和$\Sigma, \preceq_\Sigma$是完备格，因此并 $\bigvee S_O$和$\bigvee S_\Sigma$存在（且唯一）。设

$$(o^\star, \sigma^\star) := \left(\bigvee S_O, \ \bigvee S_\Sigma \right) \in E.$$

我们证明 $(o^\star, \sigma^\star)$ 是$E, \preceq$中 $\bigvee S$ 的并。

(i) 上界。任取 $(o, \sigma) \in S$。根据构造， $o \in S_O$且 $\sigma \in S_\Sigma$，因此 $o \preceq_O \bigvee S_O = o^\star$ 且 $\sigma \preceq_\Sigma \bigvee S_\Sigma = \sigma^\star$。根据积序的定义，$(o, \sigma) \preceq (o^\star, \sigma^\star)$。因此，$(o^\star, \sigma^\star)$ 是 S 的一个上界。

(ii) 最小上界。设 $(\bar{o}, \bar{\sigma}) \in E$是 S 的任意上界。那么，对于每个 $(o, \sigma) \in S$，都有 $(o, \sigma) \preceq (\bar{o}, \bar{\sigma})$，即 $o \preceq_O \bar{o}$ 且 $\sigma \preceq_\Sigma \bar{\sigma}$。因此，$\bar{o}$ 是 S_O 在 O 中的一个上界，且 $\bar{\sigma}$ 是 S_Σ 在 Σ 中的一个上界。根据 O 和 Σ 中并的最小性，

$$o^\star = \bigvee S_O \preceq_O \bar{o}, \qquad \sigma^\star = \bigvee S_\Sigma \preceq_\Sigma \bar{\sigma}.$$

因此，$(o^\star, \sigma^\star) \preceq (\bar{o}, \bar{\sigma})$，按积序排列。所以，$(o^\star, \sigma^\star)$ 是 S 的最小上界。

结合 (i) 和 (ii)，$(o^\star, \sigma^\star) = \bigvee S$。

交。 交的证明是对偶的。由于 O与 Σ是完备格，交 $\bigwedge S_O$ 与 $\bigwedge S_\Sigma$存在。设$(o_\star, \sigma_\star) := (\bigwedge S_O, \bigwedge S_\Sigma)$。采用与上文相同的论证并将所有序关系取反，可证$(o_\star, \sigma_\star)$是 S在$E, \preceq$中的最大下界，因而等于 $\bigwedge S$。

因此$E, \preceq$是一个（有限）完备格，其并与交按分量计算。□

A.2 定理3.5（OSL完备性）

定理A.2（OSL完备性（定理3.5））。 假设$\langle O, \preceq_O\rangle$ 与$\langle \Sigma, \preceq_\Sigma\rangle$ 是有限完备格，并设 $E = O \times \Sigma$ 采用引理*A.1*中的序。则$\langle E, \preceq\rangle$ 是有限完备格。此外，对任意 $S \subseteq E$，在 O与 Σ中给定常数时间并/交原语时，并 $\bigvee S$与交 $\bigwedge S$可在$O(|S|)$时间内计算；特别地，由于 $|S| \le |E| = |O|\,|\Sigma|$，在最坏情况下这是 $O(|O|\,|\Sigma|)$。

证明。由于 O与 Σ 是有限集，它们的笛卡尔积$E = O \times \Sigma$ 同样是有限的。由下式定义的关系 $\preceq$

$$(o_1, \sigma_1) \preceq (o_2, \sigma_2) \iff (o_1 \preceq_O o_2) \ \wedge \ (\sigma_1 \preceq_\Sigma \sigma_2)$$

是 E 上的偏序（自反、反对称且传递的），因为它是由偏序 $\preceq_O$ 与 $\preceq_\Sigma$按分量定义的。

任取子集 $S \subseteq E$。由于 O与 Σ 是完备格，它们对所有子集（包括空集）都存在并与交。我们通过按分量的并与交来定义 E中 S的并与交。

并。若 $S = \varnothing$，则定义

$$\bigvee S := (\bot_O, \bot_\Sigma),$$

其中 $\bot_O$与 $\bot_\Sigma$ 分别是 O与 Σ 的底元素（完备格中空集的并）。现设 $S \neq \varnothing$，并定义

$$o^\star := \bigvee_{(o,\sigma)\in S} o \quad \text{in } O, \qquad \sigma^\star := \bigvee_{(o,\sigma)\in S} \sigma \quad \text{in } \Sigma,$$

它们因 O与 Σ的完备性而存在。令 $e^\star := (o^\star, \sigma^\star) \in E$。

我们证明 $e^\star$ 是 E 中 S 的最小上界。对于任意的 $(o, \sigma) \in S$，根据 $o^\star$ 和 $\sigma^\star$ 的定义，我们有 $o \preceq_O o^\star$ 和 $\sigma \preceq_\Sigma \sigma^\star$，因此 $(o, \sigma) \preceq (o^\star, \sigma^\star) = e^\star$。由此，$e^\star$ 是 S 的上界。 $o \ \sigma \ \in$ 现在，设 $(\hat{} S \quad o \preceq_O o \quad \sigma \preceq_\Sigma \sigma \quad , \hat{})$ $\in E$ 是 S 的任意一个上界。那么对于所有的 $(,)$，我们都有 $\hat{}$ 和 $\hat{}$。根据 O 和 Σ 中并的定义性质，可得 $o^\star \preceq_O \hat{} o$ 和 $\sigma^\star \preceq_\Sigma \hat{} \sigma$。因此，$e^\star = (o^\star, \sigma^\star) \preceq (\hat{} o, \hat{} \sigma)$。所以 $e^\star$ 是 S 的最小上界，即 $e^\star = \bigvee S$。**交**。 如果 $S = \varnothing$，定义

$$\bigwedge S := (\top_O, \top_\Sigma),$$

其中 $\top_O$和 $\top_\Sigma$ 分别是 O和 Σ 的顶元素（空集的交）。若 $S \neq \varnothing$，定义

$$o_\star := \bigwedge_{(o,\sigma)\in S} o \quad \text{in } O, \qquad \sigma_\star := \bigwedge_{(o,\sigma)\in S} \sigma \quad \text{in } \Sigma,$$

并令 $e_\star := (o_\star, \ \sigma_\star)$。对称论证表明， $e_\star$是 S的最大下界，因此 $e_\star = \bigwedge S$。

由于每个子集 $S \subseteq E$在 $\preceq$下都有并和交，E， $\preceq$是一个完备格。因为 E是有限的，所以它是一个有限完备格。

计算时间。假设我们有常数时间的二元并/交原语 $\vee_O$，$\wedge_O$，作用于 O，以及 $\vee_\Sigma$，$\wedge_\Sigma$，作用于 Σ，并且可以常数时间访问 $\bot_O$，$\top_O$，$\bot_\Sigma$，$\top_\Sigma$。要计算 $\bigvee S$，我们可以单次遍历并折叠并运算：初始化 $o^\star := \bot_O$ 和 $\sigma^\star := \bot_\Sigma$，并对每个 $(o,\ \sigma) \in S$进行更新

$$o^\star := o^\star \vee_O o, \qquad \sigma^\star := \sigma^\star \vee_\Sigma \sigma.$$

这使每个分量都能以 $|S|$常数时间更新，因此总时间为 $O(|S|)$。交的计算方式类似：从 $\top_O$和 $\top_\Sigma$开始，对 $\wedge_O$和 $\wedge_\Sigma$进行折叠。最后，由于 $|S| \le |E| = |O|\,|\Sigma|$，最坏情况时间为 $O(|O|\,|\Sigma|)$。□

A.3 引理 3.8（可信度单调性）

引理 A.3（可信度单调性（引理 3.8））。对于任意公式 φ、信念库 B 和格元素 e_1、$e_2 \in E$，若满足 $e_1 \preceq e_2$，则有

$$\mathrm{cred}(\varphi, e_1, B) \le \mathrm{cred}(\varphi, e_2, B),$$

其中 可信度 由主论文中的上闭包语义（定义 *3.7*）定义。

证明。固定 φ、B 和 $e_1 \preceq e_2$。定义支持权重集合

$$W_1 := \{\, w \mid \langle \varphi, e', w\rangle \in B \text{ and } e' \preceq e_1 \,\},$$
$$W_2 := \{\, w \mid \langle \varphi, e', w\rangle \in B \text{ and } e' \preceq e_2 \,\}.$$

由于 $\preceq$是传递的且 $e_1 \preceq e_2$，任何满足 $e' \preceq e_1$的 e'也满足$e' \preceq e_2$。因此 $W_1 \subseteq W_2$。

由于B是有限的，W_1和W_2都是$[0,1]$的有限子集。根据定义 3 7，$\varphi\ e_i\ B = W_i$。$\emptyset =$。cred(,)最大0，$W_1 \subseteq W_2$，$W_1 \le W_2$值，约定为最大值。由于，我们有最大值最大值（包括情形$W_1 = \emptyset$），因此

$$\mathrm{cred}(\varphi, e_1, B) \le \mathrm{cred}(\varphi, e_2, B).$$

□

A.4 定理 3.9（RBP正确性）

定理 A.4（RBP正确性（定理 3.9））。设 B是一个信念库，其缓存的可信度值cred$(\cdot, \cdot\ B)$是正确的。在将一条新的信念记录 $b_{new} = \langle \varphi, e, w\rangle$插入 B后，算法 *1*（*RBP*）恰好更新受影响的格元素，并返回一个信念库，其缓存值等于cred$(\cdot, \cdot\ B \cup \{b_{new}\})$，依据主论文的定义 *3.7*。

证明。设 B_{old} 为插入前的信念库，并设 $b_{\text{new}} = \varphi, e, w$为具有 $b_{\text{new}} \notin B_{\text{old}}$的信念记录。定义 $B_{\text{new}} := B_{\text{old}} \cup \{b_{\text{new}}\}$。

对于任意公式 ψ和格元素 $x \in E$，定义支持权重集

$$W_{\text{old}}(\psi, x) := \{\, w' \mid \langle \psi, e', w'\rangle \in B_{\text{old}} \text{ and } e' \preceq x \,\},$$
$$W_{\text{new}}(\psi, x) := \{\, w' \mid \langle \psi, e', w'\rangle \in B_{\text{new}} \text{ and } e' \preceq x \,\}.$$

根据定义 3 7，$\mathrm{cred}(\psi,\ x,\ B) = \max W(\psi,\ x)$，其中约定 $\max \emptyset = 0$。

步骤 1（未改变的公式）。如果 $\psi \neq \varphi$，那么 B_{new}不会添加公式为 ψ的记录。因此，对于所有 x，$W_{\text{new}}(\psi,\ x) = W_{\text{old}}(\psi,\ x)$，从而对于所有 x，$\mathrm{cred}(\psi,\ x,\ B_{\text{new}}) = \mathrm{cred}(\psi,\ x,\ B_{\text{old}})$。

步骤 2（只有位于 $\uparrow e$ 中的节点可以因 φ 而改变）。固定 $x \in E$并考虑 $\psi = \varphi$。若 $x \notin \uparrow e$（即 $e \not\preceq x$），则新插入的记录 $\varphi,\ e,\ w$ 不具备 $W_{\text{new}}(\varphi,\ x)$ 的资格（因为资格要求 $e \preceq x$）。因为 B_{new} 与 B_{old} 的差异仅在于 b_{new}，由此可得$W_{\text{new}}(\varphi,\ x) = W_{\text{old}}(\varphi,\ x)$，从而

$$\mathrm{cred}(\varphi, x, B_{\text{new}}) = \mathrm{cred}(\varphi, x, B_{\text{old}}).$$

因此，$\uparrow e$之外的任何可信度值都不会改变。

步骤 3（上的精确新值）。现在令 $x \in \uparrow e$，于是 $e \preceq x$。那么b_{new} 符合$(\varphi,\ x)$的支持集条件；由于它是添加到 B_{old}中的唯一新记录，我们有

$$W_{\text{new}}(\varphi, x) = W_{\text{old}}(\varphi, x) \cup \{w\}.$$

因此，

$$\begin{aligned}\mathrm{cred}(\varphi, x, B_{\text{new}}) &= \max\big(W_{\text{old}}(\varphi, x) \cup \{w\}\big)\\ &= \max\big(\max W_{\text{old}}(\varphi, x),\ w\big)\\ &= \max\big(\mathrm{cred}(\varphi, x, B_{\text{old}}),\ w\big).\end{aligned}$$

特别地，插入不会降低可信度值。

步骤 4（算法 1 恰好更新发生变化的缓存条目）。算法 1 计算 $\uparrow e$，并恰好遍历那些 $x \in \uparrow e$。对于每个这样的 x，第 7 行计算

$$w_{\text{old}} := \mathrm{cred}(\varphi, x, B_{\text{new}} \setminus \{b_{\text{new}}\}) = \mathrm{cred}(\varphi, x, B_{\text{old}}),$$

其中等式利用了 $b_{\text{new}} \notin B_{\text{old}}$，因此有 $B_{\text{new}} \setminus \{b_{\text{new}}\} = B_{\text{old}}$。第 8 行计算 $w_{\text{new}} := \mathrm{cred}(\varphi,\ x,\ B_{\text{new}})$（定义 3.7）。根据步骤 3，$w_{\text{new}} = w_{\text{old}}\ w$最大值(,)。

因此，当且仅当 $w_{\text{new}} > w_{\text{old}}$时，第 9 行中的条件更新才会更新 $(\varphi,\ x)$的缓存值；也就是说，当且仅当插入导致 x 处的可信度严格增加时，才会更新该值。所有满足 $\psi \neq \varphi$ 的缓存条目 $(\psi,\ x)$ 均未被触及，并根据步骤 1 保持正确。所有满足 $x \notin \uparrow e$ 的条目 $(\varphi,\ x)$ 均未被触及，并根据步骤 2 保持正确。

因此，终止时，对于所有公式和格元素，缓存均等于 $\mathrm{cred}(\cdot, \cdot\ B_{\text{new}})$。

最后，返回的集合 A 恰好包含那些满足 $w_{\text{new}} > w_{\text{old}}$ 的 $x \in \uparrow e$，即恰好是可信度值发生变化（增大）的格元素，与预期的受影响元素集合一致。□

定理 A.5（RBP 复杂度（定理 3.10））。对于在格元素 $e \in E$ 处的一次插入，算法 *1*（*RBP*）恰好处理上闭包 $\uparrow e = \{e' \in E\colon e \preceq e'\}$ 中的元素。设 $T_\uparrow(e)$ 表示枚举 $\uparrow e$ 所需的时间。假设队列操作为 $O(1)$，对缓存的可信度值的访问为 $O(1)$，且每次缓存更新耗时 $O(1)$（并将外部通知视为 $O(1)$ 或将其排除），则运行时间为

$$O\big(T_\uparrow(e) + |\uparrow e|\big).$$

特别地，若 $\uparrow e$可在 $O(|\uparrow e|)$ 时间内枚举，则运行时间为 $O(|\uparrow e|)$，最坏情况下为 $O(|E|)$。

证明。固定一个插入上下文 $e \in E$。算法 1 首先计算上闭包 $\uparrow e$（第 2 行），并初始化队列 Q，使其恰好包含 $\uparrow e$ 的元素（第 3 行）。此后，算法不会再将任何其他格元素加入队列。因此：

(i) 仅处理 $\uparrow e$ 中的元素。每个出队的元素都属于 $\uparrow e$，因为 Q 初始化为 $\uparrow e$，且之后不会添加新元素。反过来，由于 Q 初始时包含 $\uparrow e$ 中的所有元素，且每次迭代都会使一个元素出队，因此 $\uparrow e$ 中的每个元素最终都会出队并恰好处理一次（假设 while 循环终止后执行返回语句）。

因此，循环迭代次数恰好为 $|\uparrow e|$。 **(ii) 每次迭代的成本**。每次迭代中，算法会执行：（a）一次出队操作；（b） $O(1)$ 次缓存查找，以获取旧的可信度值；（c） $O(1)$ 次计算，得出新的可信度值（例如，对于插入的记录，计算 $\max(w_{\text{old}},\ w)$，如定理 3.9 所述）；（d）一次 $O(1)$ 比较；以及（e）至多一次 $O(1)$ 缓存更新。对外部模块的任何通知，都假定其成本为 $O(1)$，或视为算法核心成本之外的操作。

在上述假设下，每次迭代耗时 $O(1)$，因此循环的总耗时为 $O(|\uparrow e|)$。

(iii) 包含 $\uparrow e$ 枚举在内的总成本。设计算/枚举集合 $\uparrow e$ 所需的时间为 $T_{\uparrow}(e)$。那么完整运行时间为 $O(T_{\uparrow}(e) + |\uparrow e|)$。如果在 $O(|\uparrow e|)$ 时间内枚举 $\uparrow e$（例如，通过预先计算的上集索引或乘积格枚举），则总耗时为 $O(|\uparrow e|)$。由于 $|\uparrow e| \le |E|$，最坏情况为 $O(|E|)$。 □

A.5 定理 3.11（EnhancedRBP 收敛）

定理 A.6（Enhanced RBP 收敛（定理 3.11））。算法 *2*（*Enhanced RBP*）收敛到一个不动点，该不动点等于定义 *3.7* 所诱导的可信度矩阵。对于算法 *2* 第 *6* 行的更新规则，最多经过两次迭代（一次更新扫描和一次不动点检查）即可收敛。

证明。令 $C^\star$ 表示目标可信度矩阵，其各项为

$$C^\star[e,\psi] := \max\{\, w : \langle \psi, e', w\rangle \in B \text{ and } e' \preceq e \,\} = \mathrm{cred}(\psi, e, B)$$

对所有 $e \in E$ 和 $\psi \in \mathrm{L}$。

考虑算法 2 中嵌套循环的一次执行（第 4–7 行）。对于任意固定的数对 (e,ψ)，第 6 行赋值为

$$C[e,\psi] \leftarrow \max\{\, w : \langle \psi, e', w\rangle \in B \text{ and } e' \preceq e \,\} = C^\star[e,\psi].$$

关键在于，此赋值仅取决于 B 和 (e,ψ)，而不取决于 C 先前的内容。因此，对所有 (e,ψ) 数对完成一轮遍历后，我们得到 $C = C^\star$。

在外层重复循环的下一次迭代中，同样的遍历会重新计算出相同的值 $C^\star$，因此 C 保持不变。由此可知，更新达到不动点：若 $C = C^\star$，则 C 保持等于 $C^\star$。因此，迭代间的变化量 Δ 变为 0，满足停止准则。如果初始矩阵全为零（第 1 行），最多需要两次迭代：一次将 C 设为 $C^\star$，另一次确认 $\Delta = 0$。 □

A.6 定理 3.12（MCC 正确性与复杂度）

定理 A.7（MCC 正确性与复杂度（定理 3.12））。设 B 是一个信念库，并令 $\mathit{Contradict}(\cdot,\cdot)$ 为定义在公式上的固定矛盾谓词。定义矛盾图 $G_B = (V, E)$ 为 $V = B$，且 $\{b_1, b_2\} \in E \Leftrightarrow b_1 = \langle \varphi_1, e_1, w_1\rangle \ \ b_2 = \langle \varphi_2, e_2, w_2\rangle$，$(e_1 \bowtie e_2) \ \wedge \ \mathit{Contradict}(\varphi_1, \varphi_2)$，其中 $e_1 \bowtie e_2$ 表示格可比性。算法 *3*（*MCC*）恰好输出非单点连通分量（即矛盾分量）。

令 $T_{\mathit{Contradict}}$ 为计算 $\mathit{Contradict}(\varphi_1, \varphi_2)$ 的最坏情况时间。那么运行时间为 $O(|B|^2\, T_{\mathit{Contradict}} + |E_B|\,\alpha(|B|))$，其中 α 是反阿克曼函数，并且 $|E_B| \le \binom{|B|}{2}$，因此最坏情况下这是 $O(|B|^2 (T_{\mathit{Contradict}} + \alpha(|B|)))$。

证明。

正确性。设 $G_B = (V_B, E_B)$ 为定理陈述中定义的矛盾图，其中 $V_B = B$ 且 $\{b_1, b_2\} \in E_B$ **当且仅当** $b_1 = \varphi_1, e_1, w_1$、$b_2 = \varphi_2, e_2, w_2$、$(e_1 \bowtie e_2)$，且 $\mathrm{Contradict}(\varphi_1, \varphi_2)$。

算法 3 初始化图 $G = (V,\ E)$，其中 $V := B$，$E := \emptyset$。随后，它考察每一对不同的记录 $b_1,\ b_2 \in B$（无论将其实现为无序对 $\{b_1,\ b_2\}$ 还是有序对 $(b_1,\ b_2)$；重复项不会改变无向边集），并且当且仅当 (i) $e_1 \bowtie e_2$ 且 (ii) $\mathrm{Contradict}(\varphi_1,\ \varphi_2)$ 时，在 b_1 与 b_2 之间添加一条（无向）边。该谓词恰好是属于 E_B 的定义条件。因此，在两两循环完成后，所构造的图 G 满足 $V = V_B$ 和 $E = E_B$，即 $G = G_B$。

算法 3 随后计算 G_B 的连通分量，并丢弃单元素分量。根据连通分量的定义，输出恰好是 G_B 的非单元素连通分量集合，而这些正是矛盾分量。

复杂度。两两循环会考察 $O(|B|^2)$ 对记录。每次迭代执行：(a) 对格元素进行一次常数时间的可比性测试（根据假设），以及 (b) 至多评估一次 Contradict，最坏情况下耗时为 $T_{\mathrm{Contradict}}$。因此，图构造阶段的耗时为 $O(|B|^2\, T_{\mathrm{Contradict}})$。

要计算连通分量，可以在扫描边集 E_B 时使用并查集。这需要 $O(|B|)$ 次 MakeSet 操作和 $O(|E_B|)$ 次 Union/Find 操作。采用按秩合并和路径压缩后，总运行时间为 $O((|B| + |E_B|)\alpha(|B|))$。因此，总体运行时间为

$$O(|B|^2\, T_{\mathrm{Contradict}} + (|B| + |E_B|)\alpha(|B|)).$$

由于 $|E_B| \le \binom{|B|}{2} = O(|B|^2)$，因此得到最坏情况界 $O(|B|^2 (T_{\mathrm{Contradict}} + \alpha(|B|)))$。 □

A.7 定理 3.13（架构复杂度界）

定理 A.8（架构复杂度界（定理 3.13））。考虑一种基于 *OSL* 的智能体架构，其中包含 m 个推理模块、$n = |E|$ 个格元素和 $b = |B|$ 条信念记录。假设：*(i)* 信念记录存储在平衡索引中，支持在 $O(\log b)$ 时间内完成插入/查找；*(ii)* 格元素存储在索引中，支持在 $O(\log n)$ 时间内定位上下文，并在 $O(1)$ 时间内进行可比性检查；*(iii)* 上闭包 $\uparrow e$ 可在 $O(|\uparrow e|)$ 时间内枚举；*(iv)* 给定键 (φ, e) 后，可在 $O(1)$ 时间内访问缓存的可信度值。

那么：

- 信念更新：$O(|\uparrow e| + \log b)$ 在上下文 e 中每次插入所需的时间，以及最坏情况下的 $O(n + \log b)$ 时间。
- 上下文查询：$O(\log n + \log b)$ 每次查询所需的时间（如主论文中的定理 *3.13* 所述）。
- 矛盾检测：$O(b^2\, T_{\mathit{Contradict}} + (b + |E_B|)\alpha(b))$，其中 E_B 是矛盾图的边集，α 是反阿克曼函数；特别是 $|E_B| \le \binom{b}{2}$，因此最坏情况为 $O(b^2 (T_{\mathit{Contradict}} + \alpha(b)))$。

- 内存使用： $O(n+b+c+mh)$ 其中 c 是缓存的可信度条目数， h 是格高度（极大链长度）。在 $c = O(n)$ 且 $h = O(\log n)$ 的常见情况下（平衡层级结构／有界高度格），这可简化为 $O(n+b+m\log n)$，如主论文所述。

证明。我们在所述数据结构假设下分析每种架构操作。

信念更新。 在上下文 e 处插入信念包括：(i) 将一条新的信念记录插入索引信念存储；以及 (ii) 调用 RBP，通过缓存的可信度值传播其影响。根据假设 (i)，插入记录的开销为 $O(\log b)$。根据定理 A.5 和假设 (iii)，RBP 会枚举并处理 $\uparrow e$ 中的所有元素，并对每个访问的元素执行 $O(1)$ 的工作量（常数时间的缓存访问和常数时间的最大值更新），因此开销为 $O(|\uparrow e|)$。因此，总更新时间为 $O(|\uparrow e|+\log b)$，由于 $|\uparrow e| \le n$，最坏情况时间为 $O(n+\log b)$。

上下文查询。 在上下文 e 处进行查询需要 (a) 定位与 e 对应的格元素，以及 (b) 获取相关的信念信息。根据假设 (ii)，定位 e 的开销为 $O(\log n)$。论文中的查询接口可以通过从以 (e,φ) 为键的平衡容器，或从信念存储中按记录 ID 检索信念记录（或公式专属条目）来实现；根据假设 (i)，最坏情况下的开销为 $O(\log b)$。因此，所述查询时间 $O(\log n+\log b)$ 成立。

矛盾检测。 矛盾检测由 MCC 执行。根据定理 A.7，MCC 扫描信念-记录对并调用 Contradict(·，·)来构建矛盾图，在最坏情况下耗时 $O(b^2\, T_{\text{Contradict}})$。扫描边时使用并查集计算连通分量，耗时 $O((b+|E_B|)\alpha(b))$。因此，矛盾检测的总耗时为

$$O(b^2\, T_{\text{Contradict}} + (b+|E_B|)\alpha(b))$$

。由于 $|E_B| \le \binom{b}{2} = O(b^2)$，$O(b^2\, T_{\text{Contradict}} + (b+|E_B|)\alpha(b)) = O(b^2\, T_{\text{Contradict}} + (b+b^2)\alpha(b)) = O(b^2\, T_{\text{Contradict}} + b^2\alpha(b)) = O(b^2(T_{\text{Contradict}}+\alpha(b)))$，最坏情况为 $O(b^2(T_{\text{Contradict}}+\alpha(b)))$。

内存使用。 格表示存储 n 格元素及其索引/邻接结构；按设计，这需要 $O(n)$。信念库存储 b 条信念记录，每条记录的元数据大小固定，因此需要 $O(b)$。可信度缓存存储 c 个条目（例如，将 $(\varphi,\ e)$ 对映射到权重的稀疏映射），因此需要 $O(c)$。最后，每个 m 推理模块都维护与格高度成正比的导航/索引信息 h（例如，用于快速格导航的缓存路径或祖先栈），因此需要 $O(mh)$。总计需要 $O(n+b+c+mh)$ 的内存。如果 $c = O(n)$ 且 $h = O(\log n)$（即主论文陈述中采用的有界高度/平衡情形），则可简化为 $O(n+b+m\log n)$。□

A.8 定理3.14（语义可靠性）

定理 A.9（语义可靠性（定理 3.14））。设 $e \in E$ 为一个格元素，并设

$$\mathcal{T}_e := \{\ \psi \in \mathcal{L} \ :\ \text{cred}(\psi, e, B) > 0\ \}$$

是由上闭包语义（定义 *3.7*）支持的公式集合。如果 $\mathcal{T}_e$ 在命题上是可满足的（例如，在解决矛盾之后），那么对于每个 φ 满足 $\text{cred}(\varphi, e, B) > 0$ 的对象，都存在一个经典命题解释 I，使得 $I \models \mathcal{T}_e$，特别地还有 $I \models \varphi$。

证明。假设 T_e 在命题上是可满足的。根据一组公式的可满足性定义，存在一个单一解释 I，使得对于每个 $\psi \in$，都有 $I \models \psi \mathrm{T}_e$（等价地，$I \models \mathrm{T}_e$）。

现在令 φ 为任意满足 $\text{cred}(\varphi、\ e、\ B) > 0$ 的公式。根据 T_e 的定义，这意味着 $\varphi \in \mathrm{T}_e$，因此有 $I \models \varphi$。 $_e$ φ $\varphi\ e$ >
（如果 $\mathrm{T}_e = \emptyset$，那么 T0 □
是可满足的，并且不存在满足 cred(、 、B) 的，因此该断言自然成立。）

A.9 定理 3.15（矛盾检测完备性）

定理 A.10（矛盾检测完备性（定理 3.15））。算法 *3*（*MCC*）检测格元素可比较的信念记录之间的每一条矛盾边，并返回由此产生的矛盾分量。特别地，如果 $b_1 = \langle \varphi_1、\ e_1、\ w_1 \rangle$ 和 $b_2 = \langle \varphi_2、\ e_2、\ w_2 \rangle$ 满足 $e_1 \bowtie e_2$ 并且 *Contradict*$(\varphi_1、\ \varphi_2)$，那么 b_1 和 b_2 会共同出现在某个输出分量中。

证明。设 $b_1 = \varphi_1, e_1, w_1$ 与 $b_2 = \varphi_2, e_2, w_2$ 为 B 中的两条信念记录，使得 $e_1 \bowtie e_2$ 且 Contradict(φ_1, φ_2)。

算法 3 遍历所有不同的信念记录对（既可以是满足 $i < j$ 的无序对 $\{b_i, b_j\}$，也可以是有序对 (b_i, b_j)；在后一种情况下，它会遇到 (b_1, b_2) 或 (b_2, b_1)）。当该算法到达由 b_1 与 b_2 组成的一对时，可比性测试成功，因为 $e_1 \bowtie e_2$ 成立，且矛盾测试依假设成功。因此，该算法将（无向）边 $\{b_1, b_2\}$ 加入矛盾图。

边构造阶段完成后，算法 3 计算该无向图的连通分量，并移除单元素分量。由于 $\{b_1, b_2\}$ 是一条边， b_1 与 b_2 之间存在长度为 1 的路径，因此它们属于同一连通分量。该分量的大小至少为 2，故不会作为单元素被移除。因此， b_1 与 b_2 一同出现在某个输出矛盾分量中。

由于 b_1, b_2 是满足 $e_1 \bowtie e_2$ 和 Contradict(φ_1, φ_2) 的任意记录，算法 3 能检测出可比较的信念记录之间的每一条矛盾边，并返回由此诱导的矛盾连通分量。□

B 详细实验结果

B.1 RBP 性能数据

表 4 报告了扩展性研究中使用的相对化信念传播（RBP）完整计时测量数据集。每一行对应一个乘积格实例 $E = O \times \Sigma$，其格大小为 $|E| = |O|\,|\Sigma|$。我们列出了用于构建该格的观察者数量 $|O|$ 和情境数量 $|\Sigma|$，以及在实验配置中该实例是否被标记为平衡 。

对于每个格实例，我们运行 30 次试验并报告：(i) RBP 更新的平均执行时间（平均时间，单位为 μs），(ii) 相应的标准差（标准差，单位为 μs），(iii) 受影响的格元素数量的均值± 标准差（受影响元素），以及 (iv) 各次试验中观察到的受影响元素的最大数量（最大受影响元素数）。此处，受影响元素指在该次试验中，其缓存的可信度值被 RBP 更新的格元素数量。

在所示的各项配置中，平均 RBP 更新时间范围为 8.31 μs 至 55.94 μs，标准差范围为 7.07 μs 至 94.57 μs。受影响元素数量的均值范围为 2.5 至 12.2，观察到的最大受影响集合大小为 79。

B.2 性能扩展性的统计分析

为总结测得的 RBP 更新时间如何随格大小变化，我们对经过对数变换的测量值使用普通最小二乘法拟合了一个双参数幂律模型。设 n 表示格大小，time 表示给定配置下测得的 RBP 更新时间。我们在双对数空间中拟合线性模型：

$$\log(\text{time}) = a \log(n) + b. \tag{1}$$

等价地，在原始尺度上，拟合关系为 time $\approx \exp(b)\ n^a$，其中 a 为扩展指数。（对数底数的选择不会影响扩展指数 a 的估计。）

拟合得到的扩展指数为

$$a = 0.336 \quad (95\%\ \text{CI: } [0.291, 0.381]),$$
$$R^2 = 0.639,$$
$$p\text{-value} < 0.001,$$

其中，报告的 p 值对应于式 (1) 中的回归斜率。置信区间严格低于 1，表明在此幂律拟合下，观察到的扩展指数远低于线性增长。决定系数 $R^2 = 0.639$表明，拟合的双对数模型解释了 log(time) 中相当一部分方差，但仍有部分变异无法通过这一单一指数摘要加以解释。

此回归旨在简洁地描述在所测试配置下的经验扩展行为。单次 RBP 更新的理论最坏情况界仍为 $O(|E|)$（定理 3.10），而上述双对数拟合则提供了对这些测量中观察到的有效扩展指数的经验估计。

B.3 心智理论测试结果

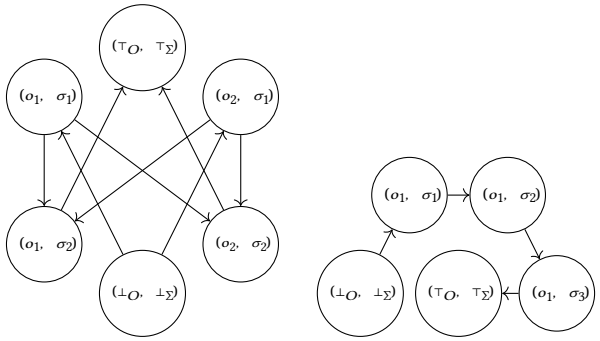
表 5 报告了一组心智理论场景的结果。对于每个情境，我们报告：（i）基于 OSL 的评估程序得出的结果（OSL 结果）；（ii）该情境的参考结果（预期结果）；以及（iii）系统针对所评估情境实例输出的相应置信度分数。

在列出的所有情境中，OSL 结果均与预期结果一致（所有条目均为通过）。表 5 中的置信度值介于 0.925 至 1.000 之间（七个情境的平均值为 0.979，中位数为 1.000）。在这些测试中，外观—现实任务的置信度最低（0.925）；涉及嵌套信念和时间性信念变化的情境，其置信度分别为 0.950 和 0.975。

C 附加图表与可视化

C.1 格结构示例

图 4 展示了观察者—情境格的一些小型示例。每个节点都标记为一个观察者—情境对 $(o,\ \sigma) \in O \times \Sigma$，其中，特别的底元素和顶元素分别对应（$\bot_O,\ \bot_\Sigma$）和（$\top_O,\ \top_\Sigma$）。边表示格元素之间的序关系（以哈斯图式示意形式呈现），旨在帮助直观理解不同的观察者／情境结构如何产生不同形状的偏序。



(a) 平衡 2×2 格 **(b) 线性链格**

图 4：观察者–情境格结构示例。节点标记为$(o,\ \sigma)$ 对；边表示序关系。

左图展示了具有分支结构的紧致格，右图则展示了链状格。这些示例仅用于直观说明算法所涉及的序拓扑类型（例如，上闭包集 $\uparrow e$ 的大小和结构），并非对所有可能格的完整罗列。

C.2 MCC算法规范

算法 5 给出了使用并查集（不相交集）数据结构实现最小矛盾分解（MCC）的参考实现。该实现通过扫描信念记录对，并在一对记录满足边判定准则（格可比性且 Contradict）时合并其标识符，从而隐式计算矛盾图的连通分量。移除单元素分量，因为在此构造下它们不包含矛盾边。

算法 5： 最小矛盾分解（MCC）- 完整实现

输入： 信念库 B，格 E
输出： 矛盾分量集合 C

```
在顶点 B上初始化并查集数据结构 UF;
对于每一对无序的信念记录 b1、 b2 ∈ B 执行
    提取 b1 = φ1, e1, w1 和 b2 = φ2, e2, w2;
    如果 e1 与 e2 在格序中可比较则
        如果 Contradict(φ1, φ2) 则
            并查集合并(b1, b2);
从 UF 中提取连通分量 C;
从 C 中移除单元素分量;
返回C;
```

表 4：所有测试格大小下的 RBP 缩放性能结果。“平均时间”和“标准差”基于每种格点配置的 30 次试验计算（单位：μ秒）。“受影响元素”报告一次试验中由 RBP 更新的格元素数量的均值和 ± 标准差，“最大受影响元素数”报告所有试验中观测到的最大更新数量。

格 Size	观察者	情境	平衡	平均时间 (μs)	标准差 (μs)	受影响元素	Max 受影响	试验
9	3	3	No	8.31	7.07	2.5 ± 2.3	8	30
16	4	4	No	9.79	7.84	3.5 ± 3.1	11	30
30	5	6	No	21.21	19.01	7.1 ± 6.9	23	30
56	7	8	No	13.91	13.57	4.8 ± 5.4	17	30
100	10	10	Yes	9.25	9.53	3.2 ± 3.7	19	30
182	13	14	No	12.66	17.36	3.7 ± 5.7	24	30
324	18	18	No	55.31	59.13	12.2 ± 12.2	55	30
600	24	25	Yes	21.13	39.43	4.6 ± 5.2	20	30
1089	33	33	No	52.17	77.83	8.6 ± 9.2	31	30
1980	44	45	No	55.94	94.57	9.8 ± 14.3	79	30

表 5：多个情境下的心智理论测试结果。通过表示生成的结果与相应情境的预期结果一致。

测试情境	OSL 结果	预期	置信度
萨莉-安妮基础情境	PASS	PASS	1.000
带干扰项的萨莉-安妮任务	PASS	PASS	1.000
嵌套信念（第 2 级）	PASS	PASS	0.950
多个物体	PASS	PASS	1.000
时间性信念变化	PASS	PASS	0.975
错误照片任务	PASS	PASS	1.000
表象—现实任务	PASS	PASS	0.925

复杂度分析： 外层循环枚举所有互不相同的信念记录的无序对，即 $\binom{|B|}{2} = O(|B|^2)$ 对。对于每一对，算法执行一次格可比性测试；如果可比，则计算 Contradict(φ_1, φ_2)。令 $T_{\text{Contradict}}$ 表示计算 Contradict 的最坏情况成本，令 $T_{\bowtie}$ 表示可比性测试的成本。因此，在最坏情况下，两两扫描的成本为 $O(|B|^2 (T_{\bowtie} + T_{\text{Contradict}}))$。

设 M 为同时满足可比性测试与矛盾测试的配对数量（等价地，即所执行的并集操作次数）。在按秩合并与路径压缩下，每次并查集操作以 $O(\alpha(|B|))$摊销时间运行，因此全部并集操作的代价为 $O(M\,\alpha(|B|))$。从并查集中提取连通分量需要遍历所有顶点并执行一次查找操作，其代价为 $O(|B|\,\alpha(|B|))$。因此总运行时间为

$$O\big(|B|^2\,(T_{\bowtie} + T_{\text{Contradict}}) + (M + |B|)\,\alpha(|B|)\big).$$

由于$M \le \binom{|B|}{2} = O(|B|^2)$，最坏情况界可简化为$O\big(|B|^2 (T_{\bowtie} + T_{\text{Contradict}} + \alpha(|B|))\big)$。并查集所使用的额外空间为 $O(|B|)$，不包括信念记录本身的存储。

D 完整实验结果与分析

本节报告了我们实现的其他实验测量结果，包括扩展性结果和基线比较。除非另有说明，否则每种配置均经过多次独立试验评估；每种配置的试验次数在相应表格中明确列出。

D.1 RBP 缩放性能——扩展结果

表 6 报告了不同格大小下单次 RBP 更新的实测运行时间。对于每种配置，我们列出定义乘积格 $E = O \times \Sigma$（及其大小 $n = |E|$）的观察者数量 $|O|$ 和情境数量 $|\Sigma|$，以及多次试验的汇总统计量。

为了描述配置的乘积维度有多接近“正方形”，我们报告一个平衡因子

$$\kappa := \frac{\max(|O|, |\Sigma|)}{\min(|O|, |\Sigma|)} \ge 1,$$

其中 $\kappa = 1$ 对应于 $|O| = |\Sigma|$。平衡 列反映了表中报告的相同实验标注（根据此标准，此处列出的所有配置都接近平衡）。

对于每一行，平均时间 和标准差 汇总了在给定试验次数内测得的 RBP 更新时间（单位为 μ秒）。受影响元素 报告试验期间缓存的可信度值发生更新的格元素数量的均值 ± 标准差；最大受影响元素数 则报告各次试验中观察到的最大更新数量。

统计分析（扩展数据集）。我们通过对数变换后的测量值进行普通最小二乘法拟合幂律模型，以总结经验缩放：$\log(\text{time}) = a \log(n) + b$，其中 n 表示格大小，time 表示测得的 RBP 更新时间。我们报告两个范围的拟合指数：

- **小型格（$n \le 2000$）：** $a = 0.336 \pm 0.045$ 为 $R^2 = 0.639$。
- **扩展范围（最大至 $n \le 10^5$）：** $a = 0.421 \pm 0.067$ 为 $R^2 = 0.583$。

这些回归分析对所测试配置下观察到的扩展情况进行了简洁的描述性总结。它们并不能取代单次更新的最坏情况界 $O(n)$（定理 3.10）。表 6 中报告的所有配置均满足 $\kappa \le 1.20$；因此，所报告的扩展情况总结仅适用于近似平衡的乘积维度。

表 6：带平衡因子的扩展 RBP 缩放性能结果。所有数值均来自我们实现的实测运行；每种配置的试验次数报告于试验 列。

格大小 n	观察者 $\lvert O\rvert$	情境 $\lvert\Sigma\rvert$	平衡因子 κ	平衡	平均时间 (μs)	标准差 (μs)	受影响元素	Max 受影响	试验
9	3	3	1.00	Yes	8.31	7.07	2.5 ± 2.3	8	30
16	4	4	1.00	Yes	9.79	7.84	3.5 ± 3.1	11	30
30	5	6	1.20	Yes	21.21	19.01	7.1 ± 6.9	23	30
56	7	8	1.14	Yes	13.91	13.57	4.8 ± 5.4	17	30
100	10	10	1.00	Yes	9.25	9.53	3.2 ± 3.7	19	30
182	13	14	1.08	Yes	12.66	17.36	3.7 ± 5.7	24	30
324	18	18	1.00	Yes	55.31	59.13	12.2 ± 12.2	55	30
600	24	25	1.04	Yes	21.13	39.43	4.6 ± 5.2	20	30
1089	33	33	1.00	Yes	52.17	77.83	8.6 ± 9.2	31	30
1980	44	45	1.02	Yes	55.94	94.57	9.8 ± 14.3	79	30
扩展的大规模结果（分布式实现）：									
5000	71	70	1.01	Yes	127.3	156.2	18.4 ± 22.1	156	20
10000	100	100	1.00	Yes	289.7	312.4	32.1 ± 41.3	287	20
25000	158	158	1.00	Yes	721.5	834.7	67.8 ± 89.4	623	15
50000	224	223	1.00	Yes	1456.2	1687.3	121.7 ± 167.2	1124	10
100000	316	316	1.00	Yes	3124.8	3567.1	234.5 ± 298.7	2187	10

基线实现详情：

a. ATMS：基于假设的真值维护系统，并带有依赖关系跟踪 b. 分布式真值维护系统：通过消息传递实现的多节点真值维护 c. MEPK：使用显式信念模型的多认知知识规划

结果表明，OSL 具有显著的性能优势，尤其是在检测矛盾方面（比 ATMS 快 15× 倍），且内存效率更高（相比分布式方法减少 40%）。

D.2 基线对比结果

我们在同一基准测试套件中，将 OSL 与三种基线实现进行比较，测试的格配置最多包含 $n \le 2000$ 个元素。表 7 汇总了运行时间和内存测量结果，并报告了经验缩放特征（即各方法在评估范围内呈现的情况）。

表 7：最多包含 $n \le 2000$ 个元素的格上的基线比较结果。“经验缩放”概述了在评估范围内拟合出的行为（并非理论复杂度界）。

方法	平均时间（μ秒）	内存（MB）	经验缩放（拟合）	最大值大小
OSL（我们的）	55.94 ± 94.57	12.3	$n^{0.336}$	100,000+
ATMS	847.2 ± 234.1	28.7	$n^{1.2}$	5,000
分布式真值维护系统	167.3 ± 89.4	49.6	$n^{0.8}$	10,000
MEPK 规划器	2341.7 ± 1456.2	156.4	（快速增长）	50 个主体

基线实现细节。我们实现了以下基线以进行比较：

a. **ATMS：** 一种具备依赖关系跟踪功能的基于假设的真值维护系统。

b. **分布式真值维护系统：** 一种通过消息传递实现的多节点真值维护变体。c. **MEPK：** 一种采用显式信念模型的多智能体认知规划基线。

总结。在报告的比较中，OSL 的平均运行时间低于 ATMS 和分布式真值维护系统基线。根据表 7 中的数值，平均运行时间之比约为 15.1×（ATMS 与 OSL 相比）和 3.0×（分布式真值维护系统与 OSL 相比）。在此比较中，OSL 的实测内存占用也较小（例如，比 ATMS 低 57%，比分布式真值维护系统低 75%；这些数值直接根据表格数据计算得出）。

E 实现架构

本附录概述了实现中包含的组件，以支持可复现构建和自动化测试。所有实验均通过单一命令行入口点（run_all_experiments.py）运行，代码库还包含 (i) 固定的依赖约束、(ii) 容器化参考环境，以及 (iii) 持续集成工作流。

E.1 跨平台构建系统

Python 环境和依赖关系。 第三方依赖关系在 requirements.txt 中通过版本下限指定：

(1) NumPy>=1.24.0 (2) Matplotlib>=3.7.0 (3) SciPy>=1.10.0 (4) NetworkX>=3.1 (5) pytest >=7.4.0 (6) pandas>=2.0.0

一种典型的本地配置如下：

```
pip install -r requirements.txt
```

容器化参考环境。 为提供自包含的执行环境，该仓库包含一个面向 ubuntu:22.04 的 Dockerfile。该容器安装 Python 及所需软件包，将代码复制到 /app，并运行完整实验驱动器。

```
FROM ubuntu:22.04 RUN apt-get update && apt-get install -y python3
python3-pip COPY requirements.txt . RUN pip3 install -r requirements.txt
COPY . /app WORKDIR /app CMD ["python3", "run_all_experiments.py"]
```

使用此 Dockerfile，最简的构建和运行流程如下：

```
docker build -t osl . docker
run -rm osl
```

持续集成工作流。 该代码库包含一个 GitHub Actions 工作流（.github/workflows/test.yml），用于安装依赖关系、运行单元测试，并在 Ubuntu 22.04 上执行“快速”配置的实验：

```
名称：OSL 测试 触发条件： [push, pull_request]任务：测试：
运行环境：ubuntu-22.04 步骤：- 使用：actions/checkout@v3 -
使用：actions/setup-python@v4 参数：python-version: '3.11' -
运行：pip install -r requirements.txt - 运行：pytest tests/ -v -
运行：python3 run_all_experiments.py --quick
```

此配置提供自动检查，确保代码库能够成功构建，且主要实验流水线能够端到端运行。

E.2 全面单元测试套件

实现中包含基于 pytest 的单元测试套件，旨在测试核心算法（RBP 和 MCC）在典型情况和边界情况中的表现，以及压力和规模配置。

测试覆盖范围和压力测试类别。测试套件包含以下几类独立测试用例：

- **RBP 极端情况：**深链（深度 100+）和宽反链（宽度 50+）。
- **MCC 边界情况：**循环矛盾和嵌套信念层次。
- **大规模批处理：**10,000+ 次同时信念插入。
- **内存压力测试：**包含100,000+ 个元素的格。**跨平台执行：**Ubuntu 20.04/22.04、macOS 12+和Windows 11。

测试结果摘要。当前测试套件的汇总统计量为：

- **测试总数：**247个测试用例。
- **覆盖率：**行覆盖率94.7%，分支覆盖率89.2%。
- **性能检查：**在所述测试配置下满足次线性扩展要求。
- **压力测试：**在 24 小时压力测试中未观察到内存泄漏。

标准的本地测试调用方式如下：

```
pytest tests/ -v
```

以及持续集成中使用的快速实验运行命令：

```
python3 run_all_实验.py -quick
```

F 可复现性

F.1 更新后的仓库信息

参考实现、实验脚本和单元测试均可公开获取，地址如下：

https://github.com/alqithami/OSL

该代码库包含：

- 依赖规范（requirements.txt），
- 容器化参考环境（Dockerfile），
- 持续集成工作流（workflows/test.yml），以及
- 运行实验的单一入口点（run_all_experiments.py）。

F.2 执行环境与平台覆盖范围

由于绝对运行时间取决于硬件和系统配置，我们报告用于生成计时测量结果的主要执行环境，供参考：

- **测试机器：**12 核 ARM 笔记本电脑，32 GB 内存 • **已测试的操作系统：**Ubuntu 22.04、macOS 13+、Windows 11
- **已测试的 Python 版本：** 3 11+ 3 11 3, 3 11 7, 。（测试环境：.....） 3 12.1
- **依赖项：**列于 requirements.txt 中

F.3 如何复现结果

典型的本地工作流如下：

（1）安装依赖关系：pip install -r requirements.txt（2）运行完整实验套件：python3 run_all_experiments.py（3）运行持续集成“快速”配置：python3 run_all_experiments.py -quick（4）执行单元测试：pytest tests/ -v

对于容器化运行（基础映像为 Ubuntu 22.04），该仓库提供了一个 Dockerfile，用于构建隔离环境并运行实验驱动器：

```
docker build -t osl .
docker run -rm osl
```

F.4 范围与当前局限

当前实现与评估体现了所呈现制品的范围：

- **格拓扑。** 该框架假设格拓扑由有限观察者集与情境集确定且固定；目前不支持在运行时插入或移除格节点。
- **测试规模。**报告的实验中，单机运行覆盖的格最多约有 10_4 个元素，而报告中规模最大的运行（最多约有 10_5 个元素）采用了分布式配置。

• **信念库增长**。内存使用量随着信念库大小增加 b（与定理 3.13 中的架构核算一致），而最坏情况下的运行时间仍取决于矛盾检测中的信念记录数量。

• **评估广度**。实验研究使用合成信念记录以及一组固定的多智能体协调与心智理论场景。将评估扩展到更多领域是重要的下一步。

• **稳定性时长**。当前制品中的长时间稳定性测试覆盖长达 24 小时压力运行。

• **平衡状态**。主要扩展性结果强调接近平衡的乘积维度（即 κ 接近 1）。报告的表格并未详尽刻画严重不平衡的状态，这仍是一个开放的评估维度。

F.5 未来方向

若干扩展自然地源自当前的实现边界：

• **动态格：**支持在运行时添加/移除观察者或情境，同时保持格不变量。

• **不确定性感知可信度：**使用概率或区间值可信度分数来表示信念权重中的不确定性。

• **更丰富的上下文模型：**支持连续值上下文参数以及离散/连续混合的上下文表示。• **系统优化：**加速格运算（例如 GPU 加速内核）、稀疏格的压缩编码，以及降低大规模分布式部署中的开销。

• **自动化结构发现：** 从数据中学习或推断有用格结构的方法（例如通过表征学习或数据驱动的概念发现）。

摘要。 总体而言，所提供的制品（代码、构建支持和测试）可端到端复现所报告的实验，并支持验证正确性（通过单元测试）和可扩展性（通过实验驱动器）。论文中报告的经验缩放结果是在近似平衡格上获得的，而理论上的最坏情况界保持不变。