arXiv:2611.29738v1 [cs.LO] 22 Nov 2026

tptl-dist：用于验证实时分布式系统的形式化演算与证明自动化

Javier Enríquez Mendoza[1], Sam Speight[2], 和 Vincent Rahli[3]

[1] 伯明翰大学 jxe275@student.bham.ac.uk[2] 伯明翰大学
s.l.speight@bham.ac.uk[3] 伯明翰大学 V.Rahli@bham.ac.uk

摘要。 我们介绍了 tptl-dist，这是一种用于描述和验证实时分布式系统的时序逻辑与标记序列演算。tptl-dist 使用 Agda 进行形式化，以提供经机器检查的正确性和工具支持，其中包括通过在 Agda 内运行的证明搜索实现自动化。该逻辑在定时命题时序逻辑（TPTL）的基础上扩展了过去算子、冻结量化、时间表达式之间的比较、领域专用原子以及时间归纳原理。我们给出了基于世界的语义，构建了区间标注演算，并建立了其相对于该语义的可靠性。为展示其表达能力，我们对标准分布式系统模式进行了编码，并对 Pistis 拜占庭广播协议的一项关键实时性质进行了机械化证明。我们的结果表明，tptl-dist 为推理分布式系统的时序性质提供了一个统一的机械化框架。

关键词： 时序逻辑、实时、分布式系统、拜占庭可靠广播、Agda

1 引言

现代系统，例如智能工厂、电网、高速支付处理系统（如 Visa、PayPal）、超低延迟 5G 网络以及任务关键型空中交通管制系统，都依赖分布式系统。在这些场景中，正确性不仅取决于发生了哪些操作，还取决于这些操作何时发生，因为延迟可能导致故障。尽管 LTL 和计算树逻辑（CTL）等时序逻辑为推理系统随时间变化的行为提供了成熟的基础，但它们不支持实时性质。TPTL [1] 和定时计算树逻辑 [2]引入了时钟和时间约束，而 PRISM [3] 和 TLA+ [4–6]等工具支持对实时系统和分布式系统进行建模与模型检测 [7–15]。然而，现有形式化方法在重要方面不同于我们的方法，并面临一些关键局限。（1）我们的工作采用完整的 Agda形式化，从而能够进行机器检验的推理，并为经过验证的工具支持奠定基础。（2）TPTL 和定时计算树逻辑等逻辑无法表达常见的分布

分布式系统中的事件（例如消息发送/接收）或主体正确性，以及

它们缺乏关于时间的归纳推理。（3）诸如 PRISM 和 TLA+ 之类的工具要求穷举探索系统的所有可能状态，这并不总是可行。（4）现有方法将时钟推理与分布式系统的通信结构分离开来。

为补充当前的研究方法体系，我们介绍 tptl-dist：一种完全形式化的时序逻辑和带标签序列演算，专门用于推理实时容错分布式协议。该逻辑在 TPTL 的基础上扩展了过去算子、冻结量化、针对时间表达式的一般比较算子，以及用于编码消息传输、内部事件和主体正确性的原子谓词。

TPTL-DIST的规则带有区间注释，其语义定义在一个配备时序推理归纳原理的抽象世界结构上。tptl-dist 的核心目标是提供一个统一框架，用于陈述并推理协议的时序和分布式两个方面。所有定义、规则和元理论结果均已在 Agda 中形式化： https://github.com/jaeem006/TPTL-DIST/blob/main/Index2025.lagda以及 https://github.com/jaeem006/TPTL-DIST/blob/main/Index2026.lagda。我们使用符号 链接到该形式化内容。

贡献 我们的贡献如下，展示了 tptl-dist 为实时分布式系统的推理提供了有原则且机械化的基础：

– TPTL-DIST，一种面向实时分布式系统的新逻辑，在 TPTL 的基础上扩展了过去算子、量词、时间表达式比较以及容错分布式系统算子。– 一个针对 TPTL-DIST 的序列演算，支持使用区间进行推理，并且对于预期语义是 sound 的。– 一个基于有序世界的通用语义框架，从而能够为无界时间推理提供有原则的归纳规则。– 在 Agda 中实现的机械化，涵盖逻辑、演算和元理论。– Pistis 实时可靠广播协议的模块化设计 [16]。– 对 Pistis 及时性属性的机械化证明，展示了 TPTL-DIST 对实时约束、消息传递模式和归纳的支持。– 我们通过一种在 Agda 内运行、且其结果由 Agda 类型检查器检查的证明搜索，自动化 TPTL-DIST 中的证明。

虽然该演算及其在 Pistis 协议中的应用已在会议论文中介绍 [17],第5节所介绍的自动化方法是全新的。

2 相关工作

已有多种证明系统被提出，用于处理类线性时序逻辑。然而，直到算子兼具全称性和存在性这一双重特性，因此一直被认为较难处理。有人在 [18]中提出了一种带标签自然演绎系统来应对这些困难。在简化并推广该系统后（见第 3节），我们提出了一种带标签序列演算，使得能够以统一的方式对 TPTL 的时间算子，尤其是直到算子和冻结算子，进行推理。

TPTL TPTL 的语法如下，其中 $\mathbf{U}$ 和 $\bigcirc$ 是通常的“直到”和“下一时刻”线性时序逻辑算子； p是一个原子命题； x 是一个时钟变量； c是一个时钟表达式； $\sim \in \{\leq, <, =, >, \geq\}$ 允许将时钟变量与时钟表达式进行比较；冻结公式 $x \cdot \varphi$ 则将当前时间绑定到 x 中的 φ:

$$\varphi ::= p \mid \varphi \wedge \varphi \mid \neg\varphi \mid \varphi \,\mathbf{U}\, \varphi \mid \bigcirc \varphi \mid x \sim c \mid x \cdot \varphi$$

最终 和 始终 算子按通常方式定义： $\Diamond\varphi \coloneqq \top \mathbf{U}\, \varphi$ 和$\Box\varphi \coloneqq \neg\Diamond\neg\varphi$；而冻结算子则允许捕获 φ 最终成立的时刻，其时间不晚于 c： $x \cdot \Diamond(x \leq c \wedge \varphi)$。为理解这一点，请考虑 TPTL 在 $\mathbb{N}$给出的离散时间概念下的语义。公式的语义由映射 ρ 给出，该映射从 $\mathbb{N}$ 映射到状态、一个时间点 $i \in \mathbb{N}$以及一个变量赋值 v; 其中状态 ρ_n 是原子赋值 σ_n 与时间戳 τ_n组成的有序对:

$\rho, i, v \models p \iff p \in \sigma_i$ $\rho, i, v \models \varphi \wedge \psi \iff \rho, i, v \models \varphi$和 $\rho, i, v \models \psi\rho, i,$
$v \models \neg\varphi \iff \rho, i, v \not\models \varphi$ $\rho, i, v \models \bigcirc\varphi \iff \rho, i+1, v \models \varphi\rho, i, v \models x \sim c \iff \tau_i - v(x)$
$\sim c$ $\rho, i, v \models x \cdot \varphi \iff \rho, i, v[x \mapsto \tau_i] \models \varphi\rho, i, v \models \varphi U\ \psi \iff \exists j \geq i.(\rho, j, v \models \psi) \wedge \forall k \in [i, j).$
$(\rho, k, v \models \varphi)$

“φ在时间 c之前最终成立”的语义，即相对于 ρ， i， v， $x \cdot \Diamond(x \leq c \wedge \varphi)$表示存在某个时间 $\tau_j \leq \tau_i + c$ ，使得对于某个 $j \geq i$， φ 为真。实现方式是通过在变量赋值 v中将 x 映射到 τ_i ，从而“冻结”当前时间 τ_i；借助 $\Diamond$ 算子跳转到更晚的时间τ_j ；最后，由约束 $x \leq c$ 保证 $\tau_j \leq \tau_i + c$。

尽管 TPTL 表达能力强大，但它无法形式化描述分布式系统的性质，例如与 Pistis 实时拜占庭容错可靠广播协议相关的性质 [16], ，因为它不支持对主体、消息、通信等进行推理。TPTL-DIST旨在解决这些问题，同时弥补 TPTL 在重要的时间相关推理机制方面的不足，例如，除推理未来性质外，还推理过去性质，以及对时间进行归纳推理。

请注意，tptl-dist 旨在支持对需要及时执行关键操作的系统进行规格说明和验证。因此，除了真实或物理的定量时间概念外，它还允许对逻辑的定性时间概念进行建模 [19, 20]。

文献中已提出其他具有类似动机的方法，即形式化验证系统的定时性质，但采用了不同的技术。下面我们将说明我们的方法与这些方法的关联及差异。

TLA+: 动作时序逻辑 与 TLA+ 的主要区别在于对时间的处理方式。我们的系统通过冻结和比较运算符，将时序构造直接整合到语言的语法中。这种时间表示方式能够简单直观地对时序条件进行建模。相比之下，TLA+ 依靠“总是”和“最终”等时序运算符来表达依赖时间的性质。尽管可以在 TLA+ 中使用表示时间的变量来捕捉定量的时间概念，但这些变量会被视为普通的状态变量，并未与

时序运算符建立内在联系。因此，虽然这种方式具有很大的灵活性，但正确管理时间的责任落在用户身上，需要用户显式更新这些变量，而不是由 tptl-dist 自动管理时间。

PRISM：概率模型检测 PRISM主要用于验证概率性质。它基于概率时态逻辑，例如 PCTL（概率计算树逻辑） [21, 22] 和 CSL（连续随机逻辑） [23, 24]。除了包含能够推理时间相关性质的时态机制外，PRISM 的核心优势还在于能够处理具有随机性、不确定性和非确定性特征的系统。TPTL-DIST 以补充 PRISM 的方法。PRISM 的模型检测引擎对可验证的实时系统和性质施加了一些限制，而 tptl-dist 则通过其演算为用户提供了更大的灵活性。

总结 TLA+ 允许用户通过变量灵活地管理时间，但这需要谨慎处理；而 PRISM 则采取更受限的方法，以实现有效的模型检测。tptl-dist 提供了无需谨慎管理变量的原语，同时在表述时间相关性质时仍保留了一定的灵活性，并提供了用于证明这些性质的演算。

3 tptl-dist

现在我们介绍 tptl-dist，这是 TPTL + Past [25], 的扩展，加入了量词、更一般的比较算子、“离散”语义和序列演算。

3.1 语法

为了推理由节点（或主体）组成、通过交换消息进行通信的分布式系统，我们固定一些类型（T），以便对主体（**主体**）、主体集合（$\textsc{Agents}$）和数据（$\textsc{Data}$）进行量化，其中$\textsc{Data}$ 用于表示主体之间交换的信息 ⚙:

$$T \in \mathsf{Type} ::= \textsc{Agent} \mid \textsc{Agents} \mid \textsc{Data}$$

令 v_a 遍历一个 $\mathcal{V}_a$ 主体类型 $\textsc{Agent}$ 变量集合，并令 c_a 遍历一个 $\mathcal{C}_a$ 主体类型 $\textsc{Agent}$常量集合；令 v_A 遍历一个 $\mathcal{V}_A$ 类型为 $\textsc{Agents}$ 的变量集合；并令 v_d 遍历一个 $\mathcal{V}_d$ 数据类型 $\textsc{Data}$ 变量集合，以及令 c_d遍历一个 $\mathcal{C}_d$ 数据类型 $\textsc{Data}$常量集合。类型为 $\textsc{Agent}$ ⚙、$\textsc{Agents}$ ⚙ 和 $\textsc{Data}$ ⚙ 的表达式定义如下:

$$i, j \in \mathsf{Agent} ::= v_a \mid c_a \qquad A \in \mathsf{Agents} ::= v_A \mid [i_1, \ldots, i_n] \qquad d \in \mathsf{Data} ::= v_d \mid c_d$$

tptl-dist 还包括用于表示时间约束的表达式，其中 v_t遍历一个时间变量的 $\mathcal{V}_t$ 集合，$\mathbf{0}$是一个常量，表示一段空的时间，而 $\bullet$ 是一个二元运算，直观地表示将两个时间表达式 ⚙相加:

$$t \in \mathsf{TimeExp} ::= v_t \mid \mathbf{0} \mid t \bullet t$$

为了比较时间表达式，我们进一步引入以下运算符 ⚙:

$$\sim \in \mathsf{Comparison} ::= \sqsubset \mid \equiv \mid \lhd$$

直观地说，$t_1 \sqsubset t_2$ 表示 t_1 严格小于 t_2，$t_1 \equiv t_2$ 表示 t_1与 t_2 相等，而 $t_1 \lhd t_2$ 表示 t_1 是 t_2的直接前驱。

要对分布式系统进行推理，必须推理代理随时间获取的知识。主要有两种方法：隐式或外部知识表示，即由外部赋予代理知识；以及显式表示，即将知识明确“存储”在状态中，使代理能够计算并依据其知识采取行动 [26]。我们采用显式方法，引入若干原子表达式，通过代理之间的消息交换来刻画知识。分布式原子 a 可以是发送原子 send(i, d, A)，表示主体 i 将数据 d 发送给代理集合 A；也可以是接收原子 recv(i, d, j)，表示主体 i 从主体 j接收了数据 d；还可以是内部事件原子 internal(i, d)，表示事件 d 发生在主体 i处；或者是正确性原子 correct(i)，表示主体 i 是正确的 ⚙:

$$a_d \in \mathsf{DistAtom} ::= \mathtt{send}(i, d, A) \mid \mathtt{recv}(i, d, j) \mid \mathtt{internal}(i, d) \mid \mathtt{correct}(i)$$

此外，tptl-dist 还包含用于推理代理集合的原子命题，如下所示，其中 $n \in \mathbb{N}$、size(A, n) 表示代理集合 A 的大小为 n，而 $i \in A$ 表示代理 i 属于代理集合 A ⚙:

$$a_s \in \mathsf{SetAtom} ::= \mathtt{size}(A, n) \mid i \in A$$

原子命题可以是分布式原子命题，也可以是集合原子命题：$a \in \mathsf{Atom} ::= a_d \mid a_s$。

tptl-dist 的语法包括：(1) 命题联结词；(2) 过去和将来的 LTL 算子；(3) 冻结算子以及作用于时间表达式的比较算子；(4) 量词；(5) 用于推理分布式系统的原子命题。其定义如下，其中$\mathbf{S}$和$\mathbf{Y}$ 分别是$\mathbf{U}$ 和 $\bigcirc$ 的过去时对应算子“自从”和“昨天”，而 v 是 $\forall v$ 中类型为 T 的变量：$T.\varphi$ 和 $\exists v : T.\varphi$ ⚙:

$$\begin{aligned}\varphi \in \mathsf{Formula} ::= {} & a \mid \top \mid \bot \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \varphi \to \varphi \mid \varphi \,\mathbf{U}\, \varphi \mid \bigcirc \varphi \\ & \mid \varphi \,\mathbf{S}\, \varphi \mid \mathbf{Y}\, \varphi \mid t \sim t \mid v_t \cdot \varphi \mid \forall v : T.\varphi \mid \exists v : T.\varphi\end{aligned}$$

我们将 $\forall v_1, \ldots, v_n : T.\varphi$ 表示为 $\forall v_1 : T. \ldots \forall v_n : T.\varphi$，对于 $\exists$ 也同样如此。否定按通常方式定义：$\neg\varphi \coloneqq \varphi \to \bot$。我们回顾*eventually*和*always*（白色）以及它们的过去时对应形式（黑色）的定义 ⚙:

$$\Diamond\varphi \coloneqq \top \,\mathbf{U}\, \varphi \qquad \Box\varphi \coloneqq \neg\Diamond\neg\varphi \qquad \blacklozenge\varphi \coloneqq \top \,\mathbf{S}\, \varphi \qquad \blacksquare\varphi \coloneqq \neg\blacklozenge\neg\varphi$$

借助 tptl-dist 的一般比较算子 $t_1 \sim t_2$，可以比在 TPTL 中更直接地定义 $\Diamond$ 的有界时间版本：$\Diamond_t\varphi \coloneqq v_t \,.\, \Diamond(v'_t \cdot v'_t \sqsubseteq (v_t \bullet t) \wedge \varphi)$，表示 φ 将在未来某个时间、最迟不超过 t 时发生，其中 $t_1 \sqsubseteq t_2$ 表示 $t_1 \sqsubset t_2 \vee t_1 \equiv t_2$。类似地，我们定义 $\Box$ 算子的有界时间版本：$\Box_t\varphi \coloneqq v_t \cdot \Box(v'_t \cdot v'_t \sqsubseteq (v_t \bullet t) \to \varphi)$，表示 φ 始终在现在到 t 之间成立。此外，我们将如下定义 严格的 曾经 算子 $\blacklozenge$ 的版本：$\blacktriangleleft\varphi \coloneqq \mathbf{Y}(\blacklozenge\varphi)$，表示 φ 曾在过去某个时间发生过，但不是现在发生 ⚙。

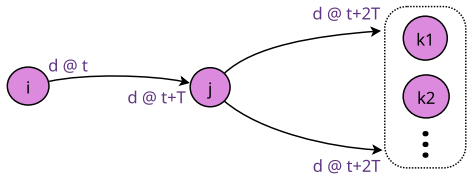

图 1. 转发示例

3.2 转发示例

现在，我们通过一个简单的转发示例来说明该语言的相关方面，展示如何捕捉分布式系统中的消息传递模式，其中代理通过发送/接收消息来交换信息（更多示例请参见附录 A）。在此示例中，消息在同步系统的代理之间中继。第 3.6 节随后详细证明了该系统的一个性质，说明了 TPTL-DIST 在第 3.5 节中介绍的规则的使用。

如图 1所示，在此示例中，主体 i 在时间 t 向另一个主体 j发送消息 d，随后该主体将 d 转发给所有其他主体。因此可以保证，任何主体都会在时间 $t+2T$之前收到该消息，前提是主体之间的最大传输延迟为 T。

假设所有主体均正确，且通信是同步的，从而确保所有发送的消息都会在区间 T内收到。以下公式对此进行了描述：若某主体在时间 t 向一组主体发送消息 d，则这些主体会在时间 $t+T$ ⚙之前收到消息 d：

$$\forall i, j : \textsc{Agent}.\forall d : \textsc{Data}.\forall A : \textsc{Agents}. \\ \Box(\texttt{send}(i, d, A) \rightarrow \Diamond_T(j \in A \rightarrow \texttt{recv}(j, d, i))) \tag{1}$$

此外，我们假设主体会立即将收到的信息转发给所有其他主体。以下公式对此进行了描述：每当 j 收到一条消息时，它都会立即将其发送给 k，并且此性质对任意 k ⚙都成立：

$$\forall i, j, k : \textsc{Agent}.\forall d : \textsc{Data}.\Box(\texttt{recv}(j, d, i) \rightarrow \texttt{send}(j, d, [k])) \tag{2}$$

如第 3.6 节所示，利用下文介绍的 TPTL-DIST 演算，可以从上述假设推导出以下公式，表明如果主体 i 向主体 d 发送消息 j，那么任意主体 k 也会在 d 之内收到该消息 $2T$（其中 $2T$ 就是 $T \bullet T$） ⚙：

$$\forall i, j, k : \textsc{Agent}.\forall d : \textsc{Data}.\Box(\texttt{send}(i, d, [j]) \rightarrow \Diamond_{2T}\texttt{recv}(k, d, j)) \tag{3}$$

3.3 语义

现在，我们将 TPTL 的语义调整并扩展到我们的演算中，尤其是为了纳入领域专用原子命题，以及一种适用于时间表达式的更一般比较运算符。公式相对于一组世界进行解释，

·-对称性 $\forall w_1, w_2$: $\mathcal{W}.w_1 \cdot w_2 = w_2 \cdot w_1$ ·-结合性 $\forall w_1, w_2, w_3$: $\mathcal{W}.w_1 \cdot w_2 \cdot w_3 = w_1 \cdot w_2 \cdot w_3$ $\preceq$ $\forall w_1\ w_2\ w_3$ $\mathcal{W}.w_1 \preceq w_2 \to w_2 \preceq w_3 \to w_1 \preceq w_3$, : $\mathcal{W}.w_1 \cdot (\cdot)(\cdot)$ · - 传递性 , , : ·- 单位元 $\forall w$: $\mathcal{W}.0 \cdot w = w$ $\preceq$ $\forall w$ $\mathcal{W}.w \preceq w$ - 0 $\forall w$ $\mathcal{W}.0 \preceq w$ 自反性 : -初始性 : 单调性 $\forall w_1, w_2, w_3, w_4$: $\mathcal{W}.w_1 \preceq w_2 \to w_3 \preceq w_4 \to w_1 \cdot w_3 \preceq w_2 \cdot w_4$ 子 $\forall w_1, w_2$: $\mathcal{W}.w_1 \lhd w_2 \to w_1 \preceq w_2$ 右稠密性 $\forall w_1, w_2$: $\mathcal{W}.w_1 \preceq w_2 \to w_1 = w_2 \vee \exists w$: $\mathcal{W}.w_1 \preceq w \wedge w \lhd w_2$ 左稠密性 $\forall w_1, w_2$: $\mathcal{W}.w_1 \preceq w_2 \to w_1 = w_2 \vee \exists w$: $\mathcal{W}.w_1 \lhd w \wedge w \preceq w_2$ 条件全序性 $\forall w_1, w_2, w$: $\mathcal{W}.w_1 \preceq w \to w_2 \preceq w \to w_1 \preceq w_2 \vee w_2 \preceq w_1$ 归纳 $\forall P$: $\mathcal{W} \to$ 集合。 $(\forall w$: $\mathcal{W}.(\forall u, v$ $\forall w$ $\mathcal{W}.P\ w$ w $\to$: $\mathcal{W}.u \preceq v \to v \lhd w \to P$ $()) \to P()) : ()$

图 2. 世界公理

其为时间表达式赋予含义。世界集合 $\mathcal{W}$ 包含一个常量 0 和一个二元运算 $\cdot$（对应于 TimeExp 中时间表达式的语义表示），并配有两个关系 $\preceq$和 $\lhd$（对应于 Comparison 中关系的语义表示），满足图 2 ⚙中的公理。

公理 $\cdot$-对称、$\cdot$-结合、$\cdot$-单位元、$\preceq$-自反、$\preceq$-传递、单调性和0-初始使 $\mathcal{W}$ 成为正序（交换）幺半群 [27]（不过请注意，我们不要求反对称性）。有序交换幺半群被用作资源组合与转换性的数学结构 [28]。稍后，我们会将时间表达式解释为世界，因此可以把时间视为一种资源。

我们将 $\lhd$ 视为捕捉 $\preceq$ 关系中的直接前驱；因此，子公理仅表示前者是后者的子关系。公理 右稠密性尤其与归纳公理一同用于推导第3.5节所述的归纳规则，这对证明引理2中的主要应用至关重要。公理 条件全序性在存在 $\preceq$-自反性和单调性的情况下，与全性等价，只需令 $w = w_1 \cdot w_2$。

右稠密性和左稠密性公理尤其用于证明$(\blacklozenge(\mathbf{Y}\ \varphi)) \leftrightarrow (\mathbf{Y}(\blacklozenge\varphi))$的从左到右和从右到左蕴涵。值得注意的是，在第4.2节中只需要前者。

以下 [26, 29–31]，公式均相对于一个模型进行解释，该模型是形如 $\langle r, \pi, w, \mathcal{V}\rangle$ 的元组，由以下部分组成：(1) 一个运行 r；(2) 一个解释函数 π，为原子公式赋予含义；(3) 一个世界 $w \in \mathcal{W}$；以及 (4) 一个变量赋值 $\mathcal{V}$，为自由变量赋予含义。运行 r 是 $\mathcal{W} \to$Global 中的一个函数，其中 Global 是一个全局状态，即 List(ClosedAtom) 的一个元素，而 ClosedAtom 是闭原子（Atom 的闭元素）的类型。解释是 Global $\to$ ClosedAtom $\to$Set 中的一个函数，用于判断给定原子在给定全局状态下是否为真。变量赋值是 $(\mathcal{V}_a \to \mathcal{C}_a) \cup (\mathcal{V}_A \to \mathrm{List}(\mathcal{C}_a)) \cup (\mathcal{V}_d \to \mathcal{C}_d) \cup (\mathcal{V}_t \to \mathcal{W})$ 中的一个函数。我们记作

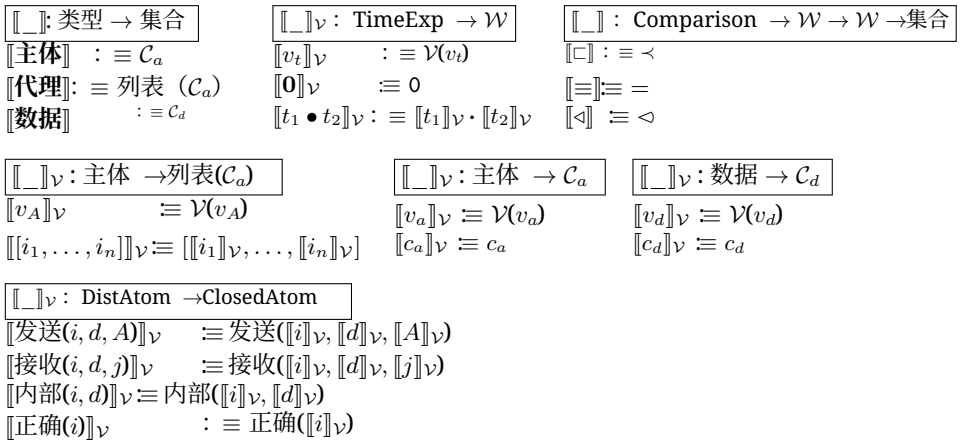

图3. 类型、时间表达式、比较运算符、代理、主体集合、数据和分布式原子命题的语义

$\mathcal{V}(X_1, \ldots, X_n)$ 表示 $\mathcal{V}$ 是一个变量赋值，其定义域包含语法表达式 X_1 至 X_n 中的自由变量。我们还将 $w_1 \prec w_2$ 记作 $\exists w :\ \mathcal{W}. w_1 \lhd w \wedge w \preceq w_2$。

图 3 介绍了时间表达式 $[\![t]\!]_\mathcal{V} \in \mathcal{W}$、类型 $[\![T]\!] \in$集合、原子命题 $[\![a]\!]_\mathcal{V} \in$ ClosedAtom、代理 $[\![i]\!]_\mathcal{V} \in \mathcal{C}_a$、主体集合 $[\![A]\!]_\mathcal{V} \in$列表($\mathcal{C}_a$)、数据 $[\![d]\!]_\mathcal{V} \in \mathcal{C}_d$以及比较运算符 $[\![\sim]\!] \in \mathcal{W} \to \mathcal{W} \to$ 集合的语义，而图 4 定义了公式的语义。具体而言，原子命题的语义 a_d 检查解释 $[\![a_d]\!]_\mathcal{V}$ 是否位于由 r在世界 w返回的全局状态中。

3.4 序列式与规则

针对类线性时序逻辑，人们提出了一些证明系统 [32, 33]。然而，众所周知，直到算子会带来困难 [34]。为规避这些困难， [18] 中提出了一种带标签自然演绎系统，沿袭了带标签演绎系统方面的一系列研究 [35–38]。我们对该系统进行简化和推广，提出一种带标签序列演算，使用同一套机制对直到算子和定时性质进行推理。

序列式的形式为 $\Gamma \vdash \varphi^{\mathsf{a}}$，其中 Γ 是一个假设列表， φ 是一个公式，而 a 是一个时间标注。时间标注如下，其中 $\centerdot$表示无标注，即该公式与时间无关：

$$\mathsf{a} ::= \centerdot \mid t \mid [t_1, t_2] \mid [t_1, t_2) \mid (t_1, t_2] \mid (t_1, t_2)$$

这样的序列式表明 φ 在时间 a 的上下文中成立 Γ。假设的形式为 φ^{a}。我们将 φ记作带注释公式 $\varphi^{\centerdot}$。与 [18], TPTL-DIST 类似，它依靠区间来捕捉 $\mathbf{U}$ 的含义。公式 $\varphi \mathbf{U} \psi$ 为真

$$
\begin{array}{l}
r, \pi, w, \mathcal{V} \models a_d \iff \pi(r(w))([\![a_d]\!]_{\mathcal{V}}) r, \pi, w, \mathcal{V} \models i \in A \iff [\![i]\!]_{\mathcal{V}} \in [\![A]\!]_{\mathcal{V}} r, \\
\pi, w, \mathcal{V} \models \text{大小}(A, n) \iff |[\![A]\!]_{\mathcal{V}}| = n r, \pi, w, \mathcal{V} \models \top \iff \text{真} r, \pi, w, \\
\mathcal{V} \models \bot \iff \text{假} r, \pi, w, \mathcal{V} \models \varphi \wedge \psi \iff r, \pi, w, \mathcal{V} \models \varphi \text{且}\ r, \pi, w, \mathcal{V} \models \psi r, \\
\pi, w, \mathcal{V} \models \varphi \vee \psi \iff r, \pi, w, \mathcal{V} \models \varphi \text{或}\ r, \pi, w, \mathcal{V} \models \psi \\
r\ \pi\ w\ \mathcal{V} \models \varphi \to \psi \iff r\ \pi\ w\ \mathcal{V} \models \varphi \quad r\ \pi\ w\ \mathcal{V} \models \psi, , , , , , \text{蕴含}, , , , , , : \\
r\ \pi\ w\ \mathcal{V} \models \forall v\ \ T.\varphi \iff r\ \pi\ w\ \mathcal{V}[v \mapsto z] \models \varphi \quad z \in [\![T]\!] \\
, , , \text{对所有}\ r\ \pi\ w\ \mathcal{V} \models \exists v\ \ T.\varphi \iff r\ \pi\ w\ \mathcal{V}[v \mapsto z] \models \varphi \quad , , , \therefore , , , \\
\\
\text{存在}\ \ z \in [\![T]\!] r\ \pi\ w\ \mathcal{V} \models t_1 \sim t_2 \iff [\![t_1]\!]_{\mathcal{V}}\ [\![\sim]\!]\ [\![t_2]\!]_{\mathcal{V}}, , , , , , , , , \\
\quad r\ \pi\ w\ \mathcal{V} \models v_t \cdot \varphi \quad \mathcal{V}[v_t \mapsto w] \models \varphi \\
r\ \pi\ w\ \mathcal{V} \models \varphi\ \ \psi \iff \exists w' \succeq w.\ r\ \pi\ w' \quad , , , U(, , , \\
\\
\quad \mathcal{V} \models \psi \wedge) (, \pi, w'', \mathcal{V} \models \varphi) \\
\forall w''.w \preceq w'' \prec w' \to\ r \\
r\ \pi\ w\ \mathcal{V} \models \bigcirc \varphi \iff \exists w' \rhd w.r\ \pi\ w'\ \mathcal{V} \models \varphi\ , , , , , , , , , S(, , ,) \\
r\ \pi\ w\ \mathcal{V} \models \varphi\ \ \psi \quad \wedge \iff \exists w' \preceq w.\ r\ \pi\ w'\ \mathcal{V} \models \psi \\
\forall w''.w' \prec w'' \preceq w \to\ r(, \pi, w'', \mathcal{V} \models \varphi)\, r, \pi, w, \mathcal{V} \models Y\ \varphi \iff \exists w' \lhd w.r, \pi, w', \\
\\
\mathcal{V} \models \varphi
\end{array}
$$

图4。 公式语义

在某个时间 t ，每当存在一个更晚的时间 t' 使得 φ 在区间 $[t, t'$内为真，并且 ψ 在 t'时为真。

如下文所述，某些规则仅在公式带有非空注释时有效。非空注释可以是 *形式，也可以是t形式，或者是一个非空区间，例如在 $t_1 \sqsubseteq t_2$成立时采用 $[t_1, t_2]$ 形式。因此，为了陈述此类规则，tptl-dist还支持形如 $\Gamma \vdash \mathsf{a}$ 的序列式，表示 a 是上下文 Γ中的非空注释。

带注释公式的语义定义如下 ⚙:

$$
\begin{array}{l}
r, \pi, \mathcal{V} \models \varphi^{\cdot} \iff r, \pi, 0, \mathcal{V} \models \varphi \qquad r, \pi, \mathcal{V} \models \varphi^{t} \iff r, \pi, [\![t]\!]_{\mathcal{V}}, \mathcal{V} \models \varphi \\
\quad r, \pi, \mathcal{V} \models \varphi^{[t_1, t_2]} \iff \forall w. [\![t_1]\!]_{\mathcal{V}} \preceq w \preceq [\![t_2]\!]_{\mathcal{V}} \to r, \pi, w, \mathcal{V} \models \varphi \\
\quad \text{and similarly for the other intervals}
\end{array}
$$

以下定义了：(1) 形如 h_1，……， h_n 的假设列表 Γ；(2) 形如 $\Gamma \vdash \varphi^{\mathsf{a}}$ 的序列式 S；以及 (3) 具有前提 S_1，……， S_n 和结论 S 的规则 R ⚙:

$$
\begin{array}{ll}
r, \pi, \mathcal{V} \models \Gamma & \iff \forall i \in [1, n]. r, \pi, \mathcal{V} \models h_i \\
r, \pi \models S & \iff \forall \mathcal{V}(\Gamma, \mathsf{a}, \varphi).\ r, \pi, \mathcal{V} \models \Gamma \to r, \pi, \mathcal{V} \models \varphi^{\mathsf{a}} \\
r, \pi \models R & \iff r, \pi \models S_1 \to \cdots \to r, \pi \models S_n \to r, \pi \models S
\end{array}
$$

规则 R 有效，当且仅当 r， $\pi \models R$ 对于所有运行 r 和解释 π均成立。上述语义已在 Agda 中形式化，我们证明了下文给出的 tptl-dist 规则均有效。

3.5 序列演算

下面列出TPTL-DIST的部分规则。完整列表见附录。

命题逻辑规则命题逻辑规则均为标准规则 ⚙:

$$\frac{\Gamma, \Delta \vdash \mathsf{a}_1}{\Gamma, \bot^{\mathsf{a}_1}, \Delta \vdash \varphi^{\mathsf{a}}}\ \bot\mathrm{L} \qquad \frac{\Gamma \vdash \varphi^{\mathsf{a}} \quad \Gamma \vdash \psi^{\mathsf{a}}}{\Gamma \vdash \varphi \wedge \psi^{\mathsf{a}}}\ \wedge\mathrm{R} \qquad \frac{\Gamma, \varphi^{\mathsf{a}}, \psi^{\mathsf{a}}, \Delta \vdash \delta^{\mathsf{a}_1}}{\Gamma, \varphi \wedge \psi^{\mathsf{a}}, \Delta \vdash \delta^{\mathsf{a}_1}}\ \wedge\mathrm{L}$$

$$\frac{\Gamma \vdash \varphi^{\mathsf{a}}}{\Gamma \vdash \varphi \vee \psi^{\mathsf{a}}}\ \vee\mathrm{R}_1 \qquad \frac{\Gamma \vdash \psi^{\mathsf{a}}}{\Gamma \vdash \varphi \vee \psi^{\mathsf{a}}}\ \vee\mathrm{R}_2 \qquad \frac{\Gamma, \varphi^t, \Delta \vdash \delta^{\mathsf{a}} \quad \Gamma, \psi^t, \Delta \vdash \delta^{\mathsf{a}}}{\Gamma, \varphi \vee \psi^t, \Delta \vdash \delta^{\mathsf{a}}}\ \vee\mathrm{L}$$

$$\frac{\Gamma, \varphi^t \vdash \psi^t}{\Gamma \vdash \varphi \to \psi^t}\ \to\mathrm{R} \qquad \frac{\Gamma, \Delta \vdash \varphi^t \quad \Gamma, \psi^t, \Delta \vdash \delta^{\mathsf{a}}}{\Gamma, \varphi \to \psi^t, \Delta \vdash \delta^{\mathsf{a}}}\ \to\mathrm{L} \qquad \frac{}{\Gamma, \varphi^{\mathsf{a}} \vdash \varphi^{\mathsf{a}}}\ ax$$

请注意，$\vee$**L** 规则中的析取不能用区间标注，因为区间的语义是 $\forall$，而 $\forall$ 不能对 $\vee$应用分配律。同样，$\to$**R** 也不能用区间标注。

非空注释 $\bot$**L** 还会引出另一个问题：如果没有前提 $\Gamma, \Delta \vdash \mathsf{a}_1$，该规则便不成立。例如，当 a_1 是空区间时，实例 $\bot^{\mathsf{a}_1} \vdash \bot$· 在语义上为假，因为 $\bot^{\mathsf{a}_1}$ 恒真。tptl-dist 提供以下规则来证明注释非空 ⚙:

$$\frac{\Gamma \vdash t_1 \sqsubseteq t_2}{\Gamma \vdash [t_1, t_2]} \qquad \frac{\Gamma \vdash t_1 \sqsubset t_2}{\Gamma \vdash [t_1, t_2)} \qquad \frac{\Gamma \vdash t_1 \sqsubset t_2}{\Gamma \vdash (t_1, t_2]} \qquad \frac{\Gamma \vdash t_1 \sqsubset t \quad \Gamma \vdash t \sqsubset t_2}{\Gamma \vdash (t_1, t_2)}$$

$$\frac{}{\Gamma \vdash t} \qquad \frac{}{\Gamma \vdash \centerdot}$$

时序逻辑规则 现在我们介绍时序规则，它们展示了演算的关键特性，其中 v_t 是一个“新鲜”变量 ⚙:

$$\frac{\Gamma \vdash t \lhd t_1 \quad \Gamma \vdash \varphi^{t_1}}{\Gamma \vdash \bigcirc \varphi^t}\ \bigcirc\mathrm{R} \qquad \frac{\Gamma \vdash t \sqsubseteq t_1 \quad \Gamma \vdash \psi^{t_1} \quad \Gamma \vdash \varphi^{[t,t_1)}}{\Gamma \vdash \varphi\, \mathbf{U}\, \psi^t}\ \mathbf{U}\mathrm{R}$$

$$\frac{\Gamma, t \lhd v_t, \varphi^{v_t}, \Delta \vdash \delta^{\mathsf{a}}}{\Gamma, \bigcirc \varphi^t, \Delta \vdash \delta^{\mathsf{a}}}\ \bigcirc\mathrm{L} \qquad \frac{\Gamma, t \sqsubseteq v_t, \psi^{v_t}, \varphi^{[t,v_t)}, \Delta \vdash \delta^{\mathsf{a}}}{\Gamma, \varphi\, \mathbf{U}\, \psi^t, \Delta \vdash \delta^{\mathsf{a}}}\ \mathbf{U}\mathrm{L}$$

UR 规则指出，要使 $\varphi\,\mathbf{U}\,\psi$ 在时间 t 为真，必须存在一个大于或等于 t 的时间 t_1，使得 ψ 成立，并且 φ 在 t 和 t_1 之间成立。**U**L 规则指出，如果 $\varphi\mathbf{U}\,\psi$ 在时间 t 为真，那么必须存在一个（变量）时间 v_t，其值大于或等于 t，且在该时间 ψ 成立；同时，φ 必须在 t 和 v_t 之间为真。

定时规则 比较运算符和冻结运算符的规则如下（我们只展示部分比较规则，因为它们反映了图中的公理 2） ⚙:

$$\frac{\Gamma, \varphi[v_t \backslash t]^t, \Delta \vdash \psi^{\mathsf{a}}}{\Gamma, v_t \cdot \varphi^t, \Delta \vdash \psi^{\mathsf{a}}}\ \cdot\mathrm{L} \qquad \frac{\Gamma \vdash \varphi[v_t \backslash t]^t}{\Gamma \vdash v_t \cdot \varphi^t}\ \cdot\mathrm{R}$$

$$\frac{}{\Gamma \vdash t_1 \bullet t_2 \equiv t_2 \bullet t_1^{\mathsf{a}}}\ \bullet sym \qquad \frac{}{\Gamma, t_1 \sqsubseteq t_2, \Delta \vdash t_1 \sqsubseteq t_2^{\mathsf{a}}}\ ax_{\sqsubseteq}$$

请注意，除非 a 也是空区间，否则假设 $t_1 \sqsubseteq t_2$ 上的注释可以是除空区间之外的任意注释。

此外，TPTL-DIST 的 ()L、]L、(]L、 [)L、()R、]R、(]R 和 [)R 规则允许实例化区间，如以下示例，其中 v_t 是一个新变量 ⚙:

$$\frac{\Gamma, \varphi^t, \Delta \vdash \psi^{\mathsf{a}} \quad \Gamma, \Delta \vdash t_1 \sqsubseteq t \quad \Gamma, \Delta \vdash t \sqsubseteq t_2}{\Gamma, \varphi^{[t_1,t_2]}, \Delta \vdash \psi^{\mathsf{a}}}\ []\mathrm{L} \qquad \frac{\Gamma, t_1 \sqsubseteq v_t, v_t \sqsubseteq t_2 \vdash \phi^{v_t}}{\Gamma \vdash \phi^{[t_1,t_2]}}\ []\mathrm{R}$$

$$\frac{\Gamma, \varphi^t, \Delta \vdash \psi^{\mathsf{a}} \quad \Gamma, \Delta \vdash t_1 \sqsubseteq t \quad \Gamma, \Delta \vdash t \sqsubset t_2}{\Gamma, \varphi^{[t_1,t_2)}, \Delta \vdash \psi^{\mathsf{a}}}\ [)\mathrm{L} \qquad \frac{\Gamma, t_1 \sqsubseteq v_t, v_t \sqsubset t_2 \vdash \phi^{v_t}}{\Gamma \vdash \phi^{[t_1,t_2)}}\ [)\mathrm{R}$$

归纳 此外，假设世界满足图 2 中的归纳公理，则以下规则成立，且假设 v_t 不在 Γ ⚙中自由出现:

$$\frac{\Gamma, v_t \cdot \varphi^{[\mathbf{0},v_t)} \vdash \varphi^{v_t}}{\Gamma \vdash \varphi^{v_t}}\ ind$$

请注意，归纳假设（IH）中冻结算子的使用 $v_t \cdot \varphi^{[\mathbf{0},v_t)}$。这是必要的，因为不仅 v_t 是声明 φ成立的时间， v_t还可以在 φ中自由出现。如果 IH 是 $\varphi^{[\mathbf{0},v_t)}$（不使用冻结算子），那么我们将假设对于 $[\mathbf{0}, v_t$中的任意时间 t ，公式 φ 在 t处成立，但其中的变量 v_t 在 φ 中仍然是 v_t。然而，我们还希望变量 v_t 也成为 t ，冻结算子通过绑定 v_t在 φ中的出现实现了这一点。

另一种直接陈述这一归纳假设的尝试是 $\mathbf{Y}(\blacksquare v_t \cdot \varphi)^{v_t}$。然而，这无法准确表达归纳假设，因为 $\mathbf{Y}$ 要求存在一个位于 v_t之前的世界，而该世界可能不存在。若要以这种方式陈述归纳假设，tptl-dist 需要包含另一种 $\mathbf{Y}$算子，例如 $\mathbf{Y}_?$，其语义为 $\forall w' \lhd w.(r, \pi, w', \mathcal{V} \models \varphi)$。相反，我们依赖上文所示的 tptl-dist 区间。最后，假设必须是$\mathbf{Y}_?(\blacksquare v_t \cdot \varphi)^{v_t}$，而不是 $v_t \cdot \mathbf{Y}(\blacksquare \varphi)^{v_t}$，后者具有不同的语义。

经典规则 tptl-dist 提供了一个 规则，可在不引入多个结论的情况下明确表达经典推理 ⚙:

$$\frac{}{\Gamma \vdash \varphi \vee \neg\varphi^{\mathsf{a}}}\ lem$$

导出规则 使用上述规则，特别是 UR 和 UL，可以为 $\Box$ 和 $\Diamond$导出右规则和左规则，以及它们的过去、有界和严格版本。这些规则对于证明实时系统的性质尤其关键，正如第 3.6 节中的一个简单示例所示。在下面的所有规则中， v_t 表示一个“新鲜”变量。例如，以下规则适用于 $\Box$，即 $\Diamond$的有界版本，以及◀ ⚙:

$$\frac{\Gamma, t \sqsubseteq v_t \vdash \varphi^{v_t}}{\Gamma \vdash \Box\varphi^t}\ \Box\mathrm{R} \qquad \frac{\Gamma, \varphi^{t_2}, \Delta \vdash \psi^t \quad \Gamma, \Delta \vdash t_1 \sqsubseteq t_2}{\Gamma, \Box\varphi^{t_1}, \Delta \vdash \psi^t}\ \Box\mathrm{L}$$

$$\frac{\Gamma \vdash \varphi^{t_1} \quad \Gamma \vdash t_1 \sqsubset t}{\Gamma \vdash \blacktriangleleft\varphi^t}\ \blacktriangleleft\mathrm{R} \qquad \frac{\Gamma \vdash \varphi^{t_1} \quad \Gamma \vdash t \sqsubseteq t_1 \quad \Gamma \vdash t_1 \sqsubseteq t_0 \bullet t}{\Gamma \vdash \Diamond_{t_0}\varphi^t}\ \Diamond\_\mathrm{R}$$

$$\frac{\Gamma, \varphi^{v_t}, v_t \sqsubset t_1, \Delta \vdash \psi^t}{\Gamma, \blacktriangleleft\varphi^{t_1}, \Delta \vdash \psi^t}\ \blacktriangleleft\mathrm{L} \qquad \frac{\Gamma, \varphi^{v_t}, t_1 \sqsubseteq v_t, v_t \sqsubseteq t_0 \bullet t_1, \Delta \vdash \psi^t}{\Gamma, \Diamond_{t_0}\varphi^{t_1}, \Delta \vdash \psi^t}\ \Diamond\_\mathrm{L}$$

可靠性

定理 1。 *TPTL-DIST* 相对于第 *3.3* 节和第 *3.4* 节中介绍的语义是 *sound*，即上文介绍的所有规则都有效。

虽然大多数规则都可以在元理论中（使用 Agda）以构造性方式证明其有效性，但有些规则需要经典逻辑，例如 引理 和 $\Box$L。

3.6 示例

第 3.2 节介绍了一个简单的转发示例：在同步网络中，代理转发其收到的消息，从而保证发送给单个代理的任何消息都会在某个时间界限内被所有代理接收。如下所示，TPTL-DIST 可以证明以下性质（完整细节见附录 C ）：

引理 1. 以下结论在所有模型中均成立 ⚙：$(eq.\ (1))^t,\ (eq.\ (2))^t \vdash eq.\ (3)^t$。

证明。假设 $(eq.\ (1))^t$ 和 $(eq.\ (2))^t$、代理 a、 b以及 c，并且数据为 d，证明 $\Box(\mathsf{send}(a,d,[b]) \to \Diamond_{2T}\mathsf{recv}(c,d,b))$ 在 t 时成立。根据 $\Box$R，只需证明：在假设 $t \sqsubseteq v_t$和 $\mathsf{send}(a,d,[b])^{v_t}$ 的情况下， $\Diamond_{2T}\mathsf{recv}(c,d,b)$ 在某个时间 v_t 成立。特别地，由 $\mathsf{send}(a,d,[b])^{v_t}$和 $(eq.\ (1))^t$，利用 $\Box$L，我们可以进一步假设 $\Diamond_T(b \in [b] \to \mathsf{recv}(b,d,a))^{v_t}$。现在根据 $\Diamond$L，我们_可以将这一假设细化为：对于某个 v'_t，$\mathsf{recv}(b,d,a)^{v'_t}$，且满足 $v_t \sqsubseteq v'_t$和 $v'_t \sqsubseteq T \bullet v_t$。由 $\mathsf{recv}(b,d,a)^{v'_t}$和 $(eq.\ (2))^t$，利用 $\Box$L，我们可以进一步假设 $\mathsf{send}(b,d,[c])^{v'_t}$。再次利用 $(eq.\ (1))^t$和 $\Box$L 处理该假设，我们可以假设 $\Diamond_T(c \in [c] \to \mathsf{recv}(c,d,b))^{v'_t}$；并利用 $\Diamond$L，将这一假设细化为：对于某个 v''_t，$\mathsf{recv}(c,d,b)^{v''_t}$，且满足 $v'_t \sqsubseteq v''_t$和 $v''_t \sqsubseteq T \bullet v'_t$。最后，根据 $\Diamond$R，由 $v_t \sqsubseteq v''_t$和 $v''_t \sqsubseteq 2T \bullet v_t$ 这一事实得出结论。_ □

4 实时分布式系统

4.1 Pistis 实时可靠广播协议

我们的主要应用场景是 Pistis 实时拜占庭容错可靠广播协议 [16]。广播协议使分布式系统中的（正确）节点发送的消息最终能够被其他（正确）节点接收（或交付），前提是网络条件“足够好”。我们尤其关注这样的场景：节点可能发生故障，故障类型既包括无害的崩溃，也包括更严重的对抗性攻击。这种故障模型称为拜占庭故障模型 [39,第2.2节。6], 在该模型中，故障节点的行为不受限制。我们还假设节点可以通过身份验证，通常借助数字签名来实现。所有消息均由其发送者签名，这意味着正确节点无法被冒充。现有许多拜占庭容错广播协议，提供不同的性质。特别是，拜占庭一致广播 [39, 第3节。10] 保证：如果某个正确节点广播一条消息，那么所有其他正确节点都会交付该消息。然而，这并未涵盖广播节点发生故障的情况；在这种情况下，

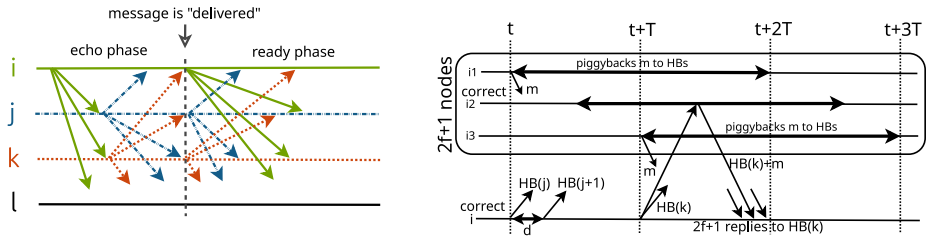

图5. Pistis：高层概览（左）与拉取机制（右）

可能只有部分其他正确节点会交付该消息。拜占庭可靠广播 [39, 第3节。11]还保证：正确节点交付一条消息，当且仅当所有正确节点都交付该消息。更准确地说，拜占庭可靠广播协议提供以下保证：有效性，即如果某个正确节点广播一条消息，那么每个正确节点最终都会交付该消息；无重复，即每个正确节点至多交付一条消息；完整性，即如果一个正确节点交付了一条消息，且其发送者正确 i，那么该消息曾由 i广播；一致性，即如果两个正确节点 i 和 j 交付了一条消息，那么这两条消息相同；以及全性，即如果一个正确节点交付了一条消息，那么每个正确节点最终都会交付一条消息。

已经开发出许多拜占庭可靠广播算法，例如Bracha算法 [40], ，它要求 $3f+1$ 个节点才能容忍多达 f 个拜占庭故障，并利用两个消息传输阶段（通常称为回声 和就绪）来保证上述性质。最近，研究者设计出了能够进一步保证及时性属性 [16, 41]的算法。例如，Pistis [16] 建立在Bracha算法之上，并进一步满足及时性，即存在一个已知上界 Δ ，使得如果某个正确节点在实时 t广播一条消息 m，那么任何正确节点都不会在 $t+\Delta$之后才交付该消息 m。

Pistis借助一种连通性证明 （PoC）机制实现及时性：若节点未能及时收到消息，就必须转为被动状态，因为它们很可能会延迟交付消息。在消息传输有界且节点已知该上界（此处称为 d）的同步 网络中，很容易实现及时性。相比之下， [16] 假设的是一种更现实的概率同步 网络，在这种网络中，消息以较低概率在上界 d 之后到达。在完全异步的网络中，即使消息传输没有上界，Pistis仍能满足上述性质。然而，频繁出现超过 d 的延迟，会导致PoC机制使过多节点转为被动状态，进而造成系统关闭。另一方面，概率同步性可以保证系统不太可能关闭。

图 5展示了包含 4 个节点的 Pistis 协议，即最多有1 个故障节点。节点 i通过向所有其他节点发送 Echo 消息来广播一条消息，其他节点则通过向所有其他节点发送各自的 Echo 消息进行回复。收到

个 $2f+1$ Echo 消息（包括自身发送的消息）[4] （或一条就绪消息）的节点会继续向所有其他节点发送一条就绪消息。任何未收到$2f+1$ 个 Echo 消息（或一条就绪消息） T 个时间单位后，若该节点已发送其 Echo 消息，则会变为非活动状态。同样，任何未收到 $2f+1$条就绪消息 $2T$ 个时间单位后，若该节点已发送其就绪消息，则会变为非活动状态。下面将详细说明这些 T 和 $2T$ 界限。

现在，我们将更详细地介绍 Pistis，说明其通过推送和拉取消息实现的主要功能，并讨论它如何实现及时性；而第 4.2 节将展示如何使用 TPTL-DIST 验证分布式系统的实时性质，方法是证明一项对及时性至关重要的性质。

推送 在时间 t 发送消息且未收到 $2f+1$ 回复的节点，到 $t+T$ 时会变为被动状态。此处， T 是 d的倍数； d是最大传输延迟，由系统管理员根据网络质量选定。假设具有同步性，则可以保证在 t 时发送的所有消息都会在 $t+d$之前被正确节点收到。然而，假设具有概率同步性，我们只能推出较弱的“法定人数”同步性假设，该假设保证：如果正确节点在时间 t发送消息，那么到时间 $t+T$时，已知有 $2f+1$ 个节点 $2f+1$ 已收到该消息 $t+T$。该性质可表述如下，其中“法定人数”是由 $2f+1$ 个节点组成的集合， $f+1$ 因此其中 ⚙个节点是正确节点：

$$\begin{aligned}&\forall p : \text{Node}.\ \text{correct}(p) \to \forall m : \text{Msg}.\forall t : \text{Time}.\ p \text{ sends } m \text{ at } t \to \\ &\exists Q : \text{Quorum}.\forall q \in Q. \\ &\quad p \text{ receives the acknowledgment within } [t, t+T] \text{ that } q \text{ received } m\end{aligned} \tag{4}$$

拉取 为确保消息延迟到达时仍能保证及时性，节点会实现上述 PoC机制，并以**心跳消息** 每隔 d个时间单位发送一次，以探测系统中的其他节点并获取（或拉取）广播消息。未收到足够心跳回复的节点会转为被动状态，因为它可能无法及时交付消息。心跳消息带有唯一标识符。对于节点 i发送的、标识符为 k的心跳消息 HB(k)（见图 5），若其在时间 t发送， $2f+1$ 则预期(k) 会在 $t+T$ 之前回复 HB i ，以保持活跃状态，从而保持正确。PoC机制保证：若法定人数的节点在 $[t, t+T]$内发送一条消息，那么所有正确节点都会在$[t+T, t+2T]$内收到该消息，从而避免节点在未交付消息的情况下仍保持活跃：

$$\begin{aligned}&\forall m : \text{Msg}.\forall t : \text{Time}.\forall Q : \text{Quorum}.(\forall q \in Q.\ q \text{ sends } m \text{ within } [t, t+T]) \to \\ &\forall p : \text{Node}.\ \text{correct}(p) \to p \text{ receives } m \text{ within } [t+T, t+2T]\end{aligned} \tag{5}$$

确保等式 (5) 成立的一个简单方法，是让节点在回复心跳消息时，附上它们在最近 $2T$ 时间区间内发送的消息。必须有一个 $2T$ 区间，因为一个节点 (i_1 见图 5) 在 Q 发送 m 的 t ($[t, t+T]$区间一端）会在该区间内回复心跳时，将 m 附带在回复中；而另一个节点（ $[t, t+2T]$见图 i_3 5）在 m 发送 $t+T$ 的（ $[t, t+T]$

[4] 由于共有 $3f+1$ 个节点，其中 f 个可能出现故障，因此节点只能期望从 $2f+1$ 个节点收到消息。

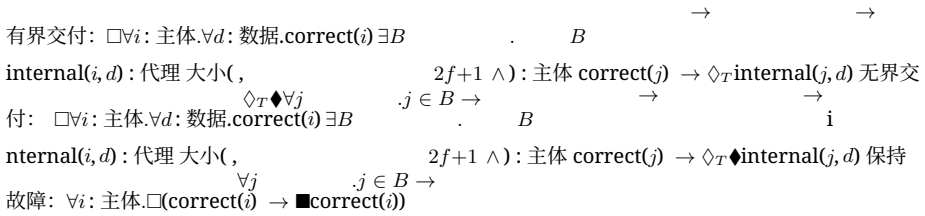

图6. tptl-dist 中的有界交付与无界交付

区间）附带发送 m 在该区间内 $[t{+}T, t{+}3T]$，从而在区间 $[t{+}T, t{+}2T]$ 上相交。如图 5 所示，这两种机制（发送心跳消息并对其作出回复）保证在 $[t{+}T, t{+}2T]$ 区间内，正确（非被动）节点（i 图 5 中）能够获得由仲裁组节点发送的消息。这是因为一个正确节点（i_2 图 5 中）必然位于以下两者的交集中：发送 HB 回复的 $2f{+}1$ 个节点（k）以及 $2f{+}1$ 个附带发送 m 的节点。

性质 上述组件保证了及时性，因为根据性质 4，如果 p正确，并在时间 t广播消息 m，那么到时间 $t_d \le t{+}T$，它会收到确认消息，得知 m 已被 $2f{+}1$ 个节点接收。随后，它便可以开始交付并发送相应的就绪消息。同样根据性质 4，到时间$t_d{+}T$， p 会收到确认消息，得知该就绪消息已被$2f{+}1$ 个节点接收。接着，我们通过对时间 t_d 进行归纳，证明存在一个由 $2f{+}1$ 个节点组成的集合 B，使得其中所有正确节点都在某个区间 $[t_{d'}, t_{d'}{+}T]$ 内、且早于 $t_d{+}T$的一段时间开始交付 m （并发送相应的就绪消息）。因此，它们的 PoC 组件会在区间 $[t_{d'}{+}T, t_{d'}{+}2T]$内，将这些就绪消息附带在心跳消息（即拉取请求）中。根据性质 5，所有正确节点都会在该区间内收到 m 。

因此，这项证明的关键在于，正如图 6 中的有界交付所示，如果某个正确节点在时间 t交付，那么就存在一个由 B个 $2f{+}1$ 节点组成的集合，使得在某个早于 t'的时间 $t{+}T$， B中的所有正确节点都已在 $t'{+}T$之前交付。这是因为，正如图 6中的无界交付tptl-dist公式所示，如果某个正确节点在时间 t交付，那么就存在一个由 $2f{+}1$ 个节点组成的集合，这些节点都在 $t{+}T$之前的某个时间交付；这是命题 4 的推论。第 4.2 节表明，在故障节点无法变为正确节点的假设下，无界交付蕴含有界交付。

4.2 有界交付的证明

现在我们来说明如何证明有界交付可由无界交付推出（见图 6），这是证明 Pistis 的及时性属性的关键。

引理2。 以下序列式在所有模型中均成立 :

$$unbounded\ delivery^{\mathbf{0}}, stay\ faulty^{\mathbf{0}} \vdash bounded\ delivery^{\mathbf{0}}$$

证明。 使用 $\Box$R，我们需要证明 无界交付 $\vdash C^{v_t}$，其中 v_t 是一个新变量，而 C 是 $\Box$ 中的公式 有界交付。使用第 3.5 节中的归纳规则，我们得到附加假设 $v_t \cdot C^{[\mathbf{0},v_t)}$。

接着，我们假设某个正确代理 i 和数据片段 d 在时间 v_t 的 i “发生”了，并需要证明存在一个代理法定人数 B，使得在 $T \bullet v_t$ 之前的某个时间 t，所有 B 中的正确代理都在时间 $T \bullet t$ 之前使 d “发生”（如有界交付所述）。将无界交付 实例化为 v_t，并使用 $\Box$L，我们知道存在一个代理法定人数 B，使得 d 在 $T \bullet v_t$ 之前于 B 中所有正确代理处“发生”。利用经典的 引理 规则，我们按 $\Diamond_T \blacklozenge \forall j$ 是否成立进行分类：在时间 v_t，**主体** $.j \in B \to \mathsf{correct}(j) \to \Diamond_T \mathsf{internal}(j,d)$；若成立，则可直接得出结论。

现在假设上述公式不成立。通过经典推理，我们推导出：在 v_t，$\Box_T \blacksquare \exists j$: **主体**$.j \in B \wedge \mathsf{correct}(j) \wedge \Box_T \neg \mathsf{internal}(j,d)$ 成立。将 $\Box$ 和 $\blacksquare$ 都实例化为当前时间 v_t，可得一个位于 B 中的正确代理 j，使得 $\Box_T \neg \mathsf{internal}(j,d)$ 在 v_t 成立（注意，假设 j 正确这一点是在 v_t）。根据无界交付，我们还可得 $\Diamond_T \blacklozenge \mathsf{internal}(j,d)$ 在 v_t 成立。因此，$\blacktriangleleft \mathsf{internal}(j,d)$ 也在 v_t 成立。使用 $\blacktriangleleft$L，可得 $\mathsf{internal}(j,d)$ 在 $v_t' \sqsubset v_t$ 成立。

使用 $[)$L，我们用 v_t'、j 和 d 实例化归纳假设，并获得一个代理法定人数 B'，使得 $\Diamond_T \blacklozenge \forall j$：**主体** $.j \in B' \to \mathsf{correct}(j) \to \Diamond_T \mathsf{internal}(j,\ d)$ 在 v_t' 成立（随后我们必须证明 j 在 $v_t' \sqsubset v_t$ 正确，这得益于 保持故障）。最后，我们用 B' 实例化结论，并证明如果 $\Diamond_t \blacklozenge \psi$ 在 $t_1 \sqsubseteq t_2$ 成立，那么它在 t_2 也成立。$\square$

5 自动化 tptl-dist 中的证明

第 3.5 节 的可靠性定理使 Agda形式化 成为 证明检查器，用于 TPTL-DIST：由有效规则构成的推导会产生有效序列式。然而，手动证明规则的有效性仍然很繁琐。例如，有界最终规则 $\Diamond$ R_

$$\frac{\Gamma \vdash \varphi^{t_1} \qquad \Gamma \vdash t \sqsubseteq t_1 \qquad \Gamma \vdash t_1 \sqsubseteq t_0 \bullet t}{\Gamma \vdash \Diamond_{t_0} \varphi^{t}} \ \Diamond\_\mathrm{R}$$

有效性证明约需 20 行 Agda ⚙ 代码。这些代码展开 $\Diamond_{t_0}\varphi$、冻结量词和满足关系，还需要两个弱化引理。在形式化过程中，变量采用德布鲁因指标，而公式的类型会记录它可能包含哪些变量：公式的 Agda 类型为 Form V，其中 V列出可能在公式中自由出现的变量的类型（时间、主体、数据，.....），最近绑定的变量索引为 0。时间表达式也以相同方式编入索引。因此，将公式置于 k 个新绑定符之下，会使其类型从 Form V变为 Form $(V, u_1, \ldots, u_k)$，并且其自由变量必须偏移 k，才能继续指向相同的绑定符。我们用 $\uparrow_k e$ 表示对公式或时间表达式进行的这种弱化 e。第3节对 $\Diamond_t$ 的定义省略了弱化，但在 Agda 中，$\Diamond_{t_0}\varphi$的两个冻结绑定符使得其定义必须引用$\uparrow_2 \varphi$ 和 $\uparrow_2 t_0$。这些引理说明，弱化不会改变含义：

在扩展了两个变量的环境中的值 $\uparrow_2 t_0$ 等于 t_0 在原始环境中的值；同样， $\uparrow_2 \varphi$ 在扩展后的模型中成立，当且仅当 φ 在原始模型中成立。手写证明会显式应用这些引理；而借助下文介绍的自动化方法，这些繁琐的处理在此类证明中便不复存在：搜索找到的推导只组合那些有效性已获一次性证明的规则，而实例化所需的弱化则由第5.3节的实例化函数以通用方式一次性处理。因此，$\Diamond$ R 的证明无需提及任何弱化引理。事实上，该规则是我们已证明有效的若干规则（$\bar{}$R、·R、UR、 $\wedge$R 和 ax）的简短组合。本节介绍我们在 Agda 中实现的一种证明搜索，它能自动找到此类组合。我们还将说明如何对包含元层未知项的目标应用该搜索：我们先将未知项抽象为变量，运行搜索，然后在找到的推导中撤销这种抽象。

要了解它们是什么，请比较两个陈述。序列式 $\vdash (p \wedge q) \to (q \wedge p)^{\mathbf{0}}$, 对于两个特定的原子命题 p 和 q，是一段具体的语法：其中的每个公式都是由构造子构成的，因此程序可以检查它，并决定应用哪条规则。相比之下，上面的规则 $\Diamond$ R 是用$\bar{}$Agda针对所有上下文 Γ、公式 φ 和时间表达式 t、 t_0、 t_1陈述的。它的 Agda 证明是一个以这些内容为参数的函数，而在函数体内， Γ、 φ 和 t、 t_0、t_1 是类型分别为上下文、公式和时间表达式的 Agda 变量。关于它们的形态，我们一无所知。搜索无法判断 φ是合取式还是原子命题，也无法确定两个这样的未知量是否相等。因此，对它们进行计算会陷入停滞。这些 Agda 层面的变量不得与 tptl-dist 的时间变量 v_t 混淆，后者属于对象语法。

此处描述的所有内容都兼容 Agda 的 --safe 编译指示。它不使用任何公设或 TERMINATING 编译指示，并且它生成的每个证明都会由 Agda 的类型检查器检查。因此，搜索本身无需被信任。

5.1 推导与生成证明的搜索

我们将推导表示为一个归纳族 ⚙ $\Delta \vdash_{\mathsf{d}} \varphi^a$ ，其构造子是第 3.5 节中演算规则的实例。每个构造子都接受其规则前提的推导，并返回其结论的推导。举例来说，以下是不带隐式参数的公理规则构造子： $\wedge$R， $\to$R 和 $\to$L：

$$
\begin{aligned}
&\mathsf{ax} \;\; : \varphi^a \in \Delta \;\to\; \Delta \vdash_{\mathsf{d}} \varphi^a \\
&\wedge\mathsf{I} \;\; : \Delta \vdash_{\mathsf{d}} \varphi^a \;\to\; \Delta \vdash_{\mathsf{d}} \psi^a \;\to\; \Delta \vdash_{\mathsf{d}} (\varphi \wedge \psi)^a \\
&\to\mathsf{I} \;\; : \Delta, \varphi^t \vdash_{\mathsf{d}} \psi^t \;\to\; \Delta \vdash_{\mathsf{d}} (\varphi \to \psi)^t \\
&\to\mathsf{L} \;\; : (\varphi \to \psi)^a \in \Delta \;\to\; \Delta \vdash_{\mathsf{d}} \varphi^a \;\to\; \Delta, \psi^a \vdash_{\mathsf{d}} \chi^b \;\to\; \Delta \vdash_{\mathsf{d}} \chi^b
\end{aligned}
$$

这里 $\varphi^a \in \Delta$ 是一个证明，表明 φ^a 是 Δ的一个假设，其记录了它在 Δ中的位置。例如，$\to$lax 是 $\vdash_{\mathsf{d}} \varphi \to \varphi^t$的一个推导，其中ax 推导出$\varphi^t \vdash_{\mathsf{d}} \varphi^t$，而 $\to$I 移到 φ 转向符的右侧。可靠性⚙,

$$\mathsf{sound} : \Delta \vdash_{\mathsf{d}} \varphi^a \;\to\; \Delta \models \varphi^a,$$

可通过对推导进行归纳得出。每种情况都会重用相应规则的有效性证明，因此几乎不需要新的语义推理。

搜索是一个函数

$$\mathsf{search}\vdash : \mathbb{N} \to (\Delta : \mathsf{Ctxt}) \to (a : \mathsf{Ann}\,\Delta) \to (\varphi : \mathsf{Form}\,\Delta) \to \mathsf{Maybe}\,(\Delta \vdash_{\mathsf{d}} \varphi^a)$$

通过对其燃料参数 n进行递归定义。在每一步，也就是针对目标以及它生成的每个子目标，它都会先尝试用公理规则关闭目标。如果失败且 $n > 0$，则对目标的主要联结词应用右规则；如果仍然失败，就依次尝试对每个假设应用左规则。无论应用哪条规则，接着都会在该规则的每个前提上以燃料 $n - 1$ 调用自身。例如， $\wedge$R 会在 $\Gamma \vdash \varphi \wedge \psi^a$ 上产生两个子目标 $\Gamma \vdash \varphi^a$ 和 $\Gamma \vdash \psi^a$；而对假设 $(\varphi \wedge \psi)^a$ 应用 $\wedge$L，则会产生一个子目标，其上下文扩展了 φ^a和 ψ^a。燃料耗尽时，只会尝试公理规则，因此 n限定了所找到推导的深度。有两条规则需要见证： $\bigcirc$R 需要一个时间 t_1满足 $t \lhd t_1$，而 $\mathbf{U}$R 需要一个时间 t_1 满足 $t \sqsubseteq t_1$。搜索从上下文中的比较假设获取这些见证。由于搜索$\vdash$ 是全函数，且由结构递归定义，Agda 可以在类型检查期间运行它。为将其结果转换为证明，我们使用标准库函数

$$\mathsf{from\text{-}just} : (x : \mathsf{Maybe}\,A) \to \mathsf{From\text{-}just}\ x$$

其返回类型由参数计算得出：From-just (just d) $= A$ 和 From-just nothing $= \top$。因此，要证明目标 $\Delta \vdash_{\mathsf{d}} \varphi^a$，只需写下 from-just(search $\vdash\ n\ \Delta\ a\ \varphi$)。在对这个项进行类型检查时，Agda 会运行搜索。如果返回 just d，该项的类型为 $\Delta \vdash_{\mathsf{d}} \varphi^a$ ，并会求值为 d。如果返回 nothing，该项的类型为 $\top$，与预期类型 $\Delta \vdash_{\mathsf{d}} \varphi^a$ 不匹配，因此 Agda 会以类型错误拒绝此定义。

5.2 元层未知项与抽象

搜索会比较公式和时间表达式是否相等，例如，使用 ax 规则，以假设关闭一个目标。对于具体语法，这些比较可以计算，但遇到 Agda 变量时就会卡住。考虑目标 $\Gamma \vdash \varphi \to \varphi^t$，其中$\Gamma$、 φ 和 t 是 Agda 变量。搜索首先应用 $\to$R，由此产生子目标 $\Gamma, \varphi^t \vdash \varphi^t$。接着，它尝试应用公理规则，即检查上下文中是否出现φ^t 。为此，它使用公式相等性的判定过程（Form-dec ）以及时间表达式相等性的判定过程，将目标与每个假设进行比较。这些判定过程通过对其参数的语法进行模式匹配来运行。由于 φ和 t 是 Agda 变量，其构造子未知，因此即使比较 φ与自身，判定过程也无法化简。因此，搜索的正规化会卡住。同样，上下文 Γ也是 Agda 变量，也会造成这个问题，因为搜索甚至无法枚举其中的假设。这些正是证明规则有效时会遇到的目标。

为了支持证明搜索，我们扩展第 3 节的语法，加入两种新的对象层变量，它们由自然数集索引： 公式变量 X_k，一种新的公式构造子，以及 时间表达式变量 x_k，一种新的

时间表达式构造子。它们不同于由冻结量词 v_t 绑定的时间变量：它们从不被绑定，仅作为元层未知量的占位符。它们的语义被刻意设定得很简单：二者都被解释为常量，其中公式变量解释为假，时间表达式变量解释为初始时间点。如下文所述，这已经足够，因为在使用可靠性之前，这些变量总会被实例化。随后，我们通过将每个元层未知量替换为一个新变量来抽象化一个目标；对于语法上相等的未知量，则使用同一个变量。在抽象目标上，所有比较都会计算，搜索也随之适用。例如，我们的示例 $\Gamma \vdash \varphi \to \varphi^t$ 变为 $\vdash \mathsf{X}_0 \to \mathsf{X}_0{}^{\mathbf{x}_0}$（上下文 Γ 另行处理，见第 5.3节）。在 $\to$R 之后，子目标为 $\mathsf{X}_0{}^{\mathbf{x}_0} \vdash \mathsf{X}_0{}^{\mathbf{x}_0}$。此时，将目标与假设进行比较，就相当于比较两个变量的索引0 和 0；Form-dec 通过计算作出判定，因此公理规则适用，搜索返回该推导。

由于这种平凡的语义，替换并不保持有效性。例如，序列式$\mathsf{X}_0{}^{\mathbf{x}_0} \vdash \bot^{\mathbf{x}_0}$ 是有效的，因为X_0 被解释为假，但它的一个实例 $\varphi^t \vdash \bot^t$ 却不是有效的。然而，替换确实保持可导出性：演算规则是模式化的，其中没有任何规则取决于变量的含义，因此在推导中实例化变量会得到一个推导。注意，$\mathsf{X}_0{}^{\mathbf{x}_0} \vdash \bot^{\mathbf{x}_0}$ 不可导出。因此，我们不是对有效序列式，而是对推导解除抽象，使用推导上的替换引理，插入$\vdash$ ⚙ （第5.3节）。它将抽象序列式的一个推导中的公式变量和时间表达式变量实例化，并将结果插入任意上下文Γ中。根据可靠性，该实例随后是有效的。出于同样的原因，变量的含义在可靠性证明中不起作用，这也是常量足够的原因。

这将通过三个步骤给出证明：（i）将目标抽象化；（ii）在抽象目标上运行搜索$\vdash$；（iii）使用 plug-$\vdash$将其找到的推导实例化。对于我们的示例，（i）将 $\Gamma \vdash \varphi \to \varphi^t$ 转换为 $\vdash \mathsf{X}_0 \to \mathsf{X}_0{}^{\mathbf{x}_0}$；（ii）找到一个推导$d$： $\vdash_{\mathsf{d}} \mathsf{X}_0 \to \mathsf{X}_0{}^{\mathbf{x}_0}$（$\to$R，随后应用公理规则）；（iii）将 d 转换为 $\Gamma \vdash_{\mathsf{d}} \varphi \to \varphi^t$的推导：将 X_0 替换为 φ，将 $\mathbf{x}_0$ 替换为 t，并将 Γ 添加到序列式的（空）上下文中。步骤（ ii ）始终是自动完成的。在方法 1（第 5.3 节）中，步骤（i）和（iii）由手动编写。在方法 2（第 5.4 节）中，反射宏也会执行这两个步骤。

5.3 方法 1：替换

关键引理 plug-$\vdash$指出，封闭序列式的推导可以进行实例化，并<行插入 i d='2'>嵌入任意上下文中。令 Δ 为一个封闭的假设列表，其唯一未知项是对象变量；令 Γ 为任意上下文。令 σ 将公式变量映射到关于 Γ中的变量的公式，并令 ρ 对时间表达式变量执行相同映射。我们定义实例化$(\!|\cdot|\!)_{\sigma,\ \rho}$，将每个 X_i 替换为 $\sigma(i)$，并将每个 $\mathbf{x}_i$ 替换为 $\rho(i)$，然后通过

$$\mathsf{plug}\text{-}\vdash\ \Gamma\ \sigma\ \rho :\ \Delta \vdash_{\mathsf{d}} \psi^b \ \to\ \Gamma, (\!|\Delta|\!)_{\sigma,\rho} \vdash_{\mathsf{d}} (\!|\psi^b|\!)_{\sigma,\rho} \tag{6}$$

对推导的归纳来证明，其中$(\!|\psi^b|\!)_{\sigma,\ \rho}$同时实例化公式 ψ 及其注释 b。这里 Γ，$(\!|\Delta|\!)_{\sigma,\ \rho}$ 实例化 Δ 的假设，并将其追加到 Γ中，同时保留 Γ自身的假设不变。

给出的推导 plug-⊢ 并非手动写出，而是表达式from-just (搜索⊢ ...) 第 5.1 节中的，它会计算出该推导。例如，方法 **1** 证明 $\Gamma \vdash_{\mathsf{d}} \varphi \to \varphi^t$ 使用

$$\mathsf{plug}\text{-}\vdash\ \Gamma\ (0 \mapsto \varphi)\ (0 \mapsto t)\ (\mathsf{from\text{-}just}\ (\mathsf{search}\vdash\ 10\ [\,]\ \mathsf{x}_0\ (\mathsf{X}_0 \to \mathsf{X}_0))).$$

最后一个参数的求值结果是推导 →I ax，它属于 $\vdash_{\mathsf{d}} \mathsf{X}_0 \to \mathsf{X}_0{}^{\mathsf{x}_0}$ (→R（应用于一个公理）。Agda 分两步对这个项进行类型检查。首先，from-just x 的类型取决于 x 是否为 just，因此 Agda 会对搜索进行求值，直到它返回 just (→I . . .)。其中的推导是计算的结果，其类型由 search⊢ 的类型保证，因此不会对它进行类型检查。其次，根据 (6)，整个项的类型仅根据序列式 $\vdash_{\mathsf{d}} \mathsf{X}_0 \to \mathsf{X}_0{}^{\mathsf{x}_0}$、$\sigma$ 和 ρ 计算得出，方法是将 X_0 替换为 φ，并将 x_0 替换为 t，从而得到 $\Gamma \vdash_{\mathsf{d}} \varphi \to \varphi^t$。对 plug-⊢ 的应用进行类型检查时不会运行它，因此只有在证明本身被求值时，plug-⊢ 才会遍历该推导。

证明 (6) 中值得关注的情况是 ·R 和 **U**R，这两条规则都涉及一个新的时间变量：·R 通过用一个时间表达式替换冻结量词所约束的变量来消去该变量；而将 []R 应用于 **U**R 的第三个前提，则会用一个新鲜时间变量扩展上下文（该变量代表两个注释之间的中间时间点）。本段其余部分中，我们省略下标，将 $(\!|\cdot|\!)$ 写作 $(\!|\cdot|\!)_{\sigma,\rho}$，将 $(\!|\cdot|\!)^+$ 写作在多一个时间变量下使用的实例化；后者将每个 $\mathbf{X}_i$ 替换为 $\uparrow_1 \sigma(i)$，并将每个 $\mathbf{x}_i$ 替换为 $\uparrow_1 \rho(i)$，从而使它们的自由时间变量跳过这个新变量。这些情况依赖于两个交换引理。首先，实例化与弱化可交换：$(\!|\uparrow_1 \varphi|\!)^+ = \uparrow_1 (\!|\varphi|\!)$。其次，实例化与用时间表达式替换约束时间变量可交换：

$$(\!|\varphi[v_t \backslash t]|\!) = (\!|\varphi|\!)^+[v_t \backslash (\!|t|\!)].$$

例如，·R 得出 $\Delta \vdash_{\mathsf{d}} (v_t \cdot \varphi)^t$，其前提为 $\Delta \vdash_{\mathsf{d}} \varphi[v_t \backslash t]^t$。通过归纳，该前提给出 $\Gamma, (\!|\Delta|\!) \vdash_{\mathsf{d}} (\!|\varphi[v_t \backslash t]|\!)^{(\!|t|\!)}$ 的推导；根据第二个引理，这就是 $\Gamma, (\!|\Delta|\!) \vdash_{\mathsf{d}} (\!|\varphi|\!)^+[v_t \backslash (\!|t|\!)]^{(\!|t|\!)}$。应用 ·R 后，得到 $\Gamma, (\!|\Delta|\!) \vdash_{\mathsf{d}} (v_t \cdot (\!|\varphi|\!)^+)^{(\!|t|\!)}$，这正是结论的一个实例 $(\!|(v_t \cdot \varphi)^t|\!)$。类似地，由于**U**R 的第三个前提会向上下文中添加一个新鲜时间变量，因此其中提到了被弱化的公式$\uparrow_1 \varphi$，根据第一个引理，该公式的实例与 $\uparrow_1 (\!|\varphi|\!)$ 相匹配。这两个引理都是通过对整个语法进行归纳证明的，且不需要函数外延性。

有一个设计决策至关重要。实例化会遍历公式和上下文，在到达 $\mathbf{X}_i$ 的某个出现位置之前，可能会经过不属于 Γ 的时间变量：公式中由冻结量词绑定的变量，或通过 Δ 或 **U**R 的第三个前提添加到上下文中的变量。如果它已经经过了 k 个这样的变量，就会将 $\mathbf{X}_i$ 替换为 $\uparrow_k \sigma(i)$，使得 $\sigma(i)$ 的自由时间变量跳过这些 k 个变量（$\mathbf{x}_i$ 和 ρ 的情况也类似；从上面的 $(\!|\cdot|\!)$ 到 $(\!|\cdot|\!)^+$ 会使 k 增加 **1**）。关键在于，弱化会在遍历过程中累积

并且只应用一次，作为单个弱化 $\uparrow_k$，而不是作为 k 连续的弱化 $\uparrow_1(\cdots(\uparrow_1\sigma(i))\cdots)$。两者相等，但仅在命题上相等，而非通过计算得出。派生算子（如 $\Diamond_{t_0}\varphi$ ）正是使用这些累积弱化来定义的（$\uparrow_2\varphi$ 以及 $\uparrow_2 t_0$ 在其两个冻结绑定符之下的情形）。使用累积弱化，抽象目标 $\Diamond_{x_0}X_0$ 的实例化因此是 定义上 $\Diamond_{t_0}\varphi$。因此，式（6）可以直接应用于原始目标，无需任何传输。若没有这一设计，派生算子中的每个绑定符都需要显式的重写步骤，而这恰恰是搜索旨在避免的推理方式。

最后，一个前提共享其结论的上下文的规则，可以由一个序列式推出，其中这些前提已成为假设 ⚙：

$$\mathsf{hyps{\to}rule} : \models \left(\Gamma, \varphi_1^{a_1}, \ldots, \varphi_n^{a_n} \vdash \psi^b\right) \to \models \frac{\Gamma \vdash \varphi_1^{a_1} \quad \cdots \quad \Gamma \vdash \varphi_n^{a_n}}{\Gamma \vdash \psi^b} \tag{7}$$

这可通过语义层面的割和弱化得到。

现在我们可以证明 $\Diamond$ R 是有效的，也就是说，它对任意的¯上下文 Γ、公式 φ 以及时间表达式 t_0、 t 和 t_1都成立。将各部分组合起来，证明为 ⚙

$$\begin{gathered}\mathsf{hyps{\to}rule}\left(\mathsf{sound}\,(\mathsf{plug}\text{-}\vdash\ \Gamma\ \sigma\ \rho\ d)\right)\\ \text{with}\quad \sigma = [0 \mapsto \varphi],\ \rho = [0 \mapsto t_0, 1 \mapsto t, 2 \mapsto t_1],\end{gathered}$$

其中， $d = \mathsf{from\text{-}just}\,(\mathsf{search}\vdash\ 10\ \Delta\ \mathrm{x}_1\ \Diamond_{x_0}X_0)$ 是搜索在燃料为 10 时自动找到的推导，而 $\Delta = \mathrm{x}_1 \sqsubseteq \mathrm{x}_2, \mathrm{x}_2 \sqsubseteq \mathrm{x}_1 \bullet \mathrm{x}_0, {X_0}^{\mathrm{x}_2}$。

在方法 1 中，用户手动写出 $\Diamond$ R 的证明：抽象序列式 ¯${X_0}^{x_1}$ (step (i))、代换 σ 和 ρ，以及 hyps→规则和 plug-⊢ (step (iii)) 的应用。只有 d 是自动找到的。Agda 会检查实例化是否与规则匹配。

5.4 方法 2：反射

方法 2 使用一个宏自动完成方法 1 中的两个手动步骤，该宏prove是使用 Agda 的反射 API 编写的。宏写在证明的位置，并接受关于序列式的三类陈述：S：即 S 可推导（$\Delta \vdash_{\mathsf{d}} S$）、 S有效（$\models S$），以及规则 $S_1, \ldots, S_n / S$有效（$\models S_1, \ldots, S_n / S$），其中各前提与结论共享上下文。三者都归结为同一个问题：为一个封闭序列式求出推导。第一种情形显而易见。第二种情形可由 S 的推导根据可靠性得出。第三种情形中，等式（7）将规则的有效性归约为 S 的有效性，其中将前提作为假设加入。宏的执行步骤如下。

1. 它会获取待证明的命题（即 Agda 目标中预期的类型）作为引用的 Agda 项，也就是prove 策略所写的位置（例如下文的$M \models \Diamond \mathrm{R}(\Gamma, t_0, t, t_1, \varphi)$）。它从中提取序列式 S（以及¯规则的前提），并计算出 $\Delta \vdash_{\mathsf{d}} \psi^b$ 的 S 抽象版本（step (i)），其形式符合等式 (6) 的要求。 S 的上下文被拆分为一部分 Γ，

通常是一个 Agda 变量（或为空），以及写在它后面的具体假设，例如 $t \triangleleft t_1$和 A^{t_1}在 $\Gamma, t \triangleleft t_1, A^{t_1} \vdash \bigcirc A^t$中，其中 t和 t_1是任意时间表达式。封闭上下文 Δ 由这些假设构成；对于规则，还包括其前提 $\Gamma \vdash \varphi_i^{a_i}$的右端项 $\varphi_i^{a_i}$，如等式 (7) 所示。在 Δ 以及 S的右端项 ψ^b 中，元层未知项（第 5.2 节）会被抽象处理：作为 Agda 变量的公式（例如 A）会变为公式变量 X_k；作为 Agda 变量的时间表达式（例如 t 和 t_1）会变为时间表达式变量 x_k，且语法相同的子项使用同一个变量。在上面的“下一时刻”示例中，$\Delta \vdash_{\mathsf{d}} \psi^b$ 是 $\mathrm{x}_0 \triangleleft \mathrm{x}_1, \mathrm{X}_0^{\mathrm{x}_1} \vdash_{\mathsf{d}} \bigcirc \mathrm{X}_0^{\mathrm{x}_0}$。

2. 它会将 search$\vdash$ 归一化为 $\Delta \vdash_{\mathsf{d}} \psi^b$（步骤 (ii)）。
3. 它构造第5.3 节中的项（步骤 (iii)）：plug-$\vdash$ 应用于 Γ（在第 1 项中移除其具体假设后剩余的 S 上下文部分）、恰如方法 1 中那样由 from-just(search $\vdash n\ \Delta\ \ldots$) 得到的推导，以及 σ 和 ρ；后两者将变量映射回原始子项。其类型为 $\Gamma, \langle\!|\Delta|\!\rangle_{\sigma,\rho} \vdash_{\mathsf{d}} \langle\!|\psi^b|\!\rangle_{\sigma,\rho}$，再次是 S（对于规则，将前提作为假设）。对于有效性陈述，它随后应用 sound；对于规则，还会应用 hyps→rule。

Agda 会针对原始陈述对所得项进行类型检查。整个 $\Diamond$ R 规则的证明变为⚙：

$$\mathsf{rule}\Diamond{\downarrow}\mathsf{R\text{-}sat} : \forall\, M\, \Gamma\, t_0\, t\, t_1\, \varphi.\ M \models \Diamond\_\mathrm{R}(\Gamma, t_0, t, t_1, \varphi)$$
$$\mathsf{rule}\Diamond{\downarrow}\mathsf{R\text{-}sat}\ \cdots = \mathsf{prove}\ 10,$$

以及由任意 Agda 公式 A、B 和 C构成的序列式，以及时间表达式 t，均以相同的方式 ⚙得到证明：

$$\rightarrow\text{-}\mathsf{trans} : \forall\, M\, A\, B\, C\, t.\ M \models \left(\vdash (A \rightarrow B) \rightarrow (B \rightarrow C) \rightarrow A \rightarrow C^t\right)$$
$$\rightarrow\text{-}\mathsf{trans}\ \cdots = \mathsf{prove}\ 10.$$

宏本身无需被信任：如果它错误地抽象了该陈述，项就无法通过类型检查。

为什么不直接在推导项中进行替换？借助反射，宏同时拥有搜索找到的推导 d 以及手头作为引用项的原始子项。因此，对于步骤 (iii)，一个捷径自然浮现出来：直接在引用的 d中用原始子项替换变量，然后让 Agda 检查结果。由于规则是模式化的，d 的每一步仍然是同一规则的一个实例，这样便可完全避免 plug-$\vdash$ 及其元理论。对于命题推理而言，这一捷径是可行的，例如对于 $\Gamma \vdash \varphi \rightarrow \varphi^t$，但无法适用于冻结量词。在 d中，·R 步骤针对 $\Diamond_{\mathrm{x}_0}\mathrm{X}_0$ 提及 $(\uparrow_1 \mathrm{X}_0)[v_t \backslash t]$，其计算结果为 X_0。一旦将 X_0 替换为 φ，·R 就要求 $(\uparrow_1 \varphi)[v_t \backslash t]$，而它只在命题上等于 φ，因此该项会被拒绝。plug-$\vdash$ 通过第 5.3 节的交换引理一劳永逸地处理了这一问题。

5.5 Comparison andEvaluation

两种方法都遵循相同的三个步骤，区别仅在于步骤 (i) 和 (iii) 由谁执行：用户还是宏。两者都能生成证明。搜索、

宏和抽象都无需信任，而且一切都兼容 Agda 的 --safe 编译指示。

我们在表1列出的目标上评估了这两种方法。这里仅展示了其中一部分目标；完整的自动证明示例见 https://github.com/jaeem006/TPTL-DIST/blob/main/Examples.lagda 。每个目标都证明两次：一次使用方法 1，即手动编写抽象并应用 plug-⊢；另一次使用宏 prove（方法 2），燃料始终设为 10。表格上半部分包含关于任意公式和时间表达式的相继式，其中一些带有具体假设。下半部分则包含演算在任意上下文和任意时间表达式下的（导出）规则。这些规则的有效性此前已在语义层面通过手动方式证明，现在则由 prove 得出。对于 $\Diamond$ R，这将约 20 行¯的语义推理或 15 行的手写推导缩减为一行。

为了量化搜索过程，我们使用经过插桩的 search⊢ 副本来统计递归调用次数，即它在返回前探索的搜索树节点数；同时统计它返回的推导中的规则应用次数。我们还报告每个目标得以解决所需的最少燃料。所报告的时间是 Agda 对各定义进行类型检查所花费的时间，使用 Agda 2.7.0.1 和标准库 2.1.1，通过 --profile=definitions 测得（运行十次取中位数）。所有实验均在 Linux 容器（Ubuntu 24.04）中运行，使用 Intel Xeon 处理器的 4 个核心（2.10GHz）和 15GB 内存，并采用 Ubuntu 打包的 Agda 二进制文件。其他机器上的绝对耗时会有所不同，但两种方法之间以及各目标之间的耗时比例才是关键。

对于大多数目标，搜索空间都很小：搜索探索的节点少于 200 个，返回的推导最多包含 41 次规则应用。例外是 distr，其中析取的左规则会产生两个分支，搜索因此探索了 620 个节点。这是唯一一个方法 1 耗时超过 0.12 秒的目标（耗时 248 毫秒，是 chain_6 耗时的两倍多）。

方法 2 比方法 1 慢 5 到 25 倍，每个目标的开销为 0.07 到 1.4 秒。两种方法都在同一个抽象序列式上执行相同的搜索，并且都将同一类项交给 Agda，其中推导表现为 from-just (search⊢ 10 Δ ...)。差异来自宏在此之前于类型检查时间所做的工作：它通过反射检查语句，推断每个子项的类型，以判断它是要抽象的公式还是时间表达式，构建 Δ, σ 和 ρ，并再次执行搜索以检查是否成功，否则报告错误。第一部分的开销随语句大小增长，如链式规则所示；最后一部分则随搜索成本增长，如 distr 所示。推导的大小无关紧要：为 $\rightarrow$找到的推导几乎是为 $\Diamond$ R 找到的推导的三倍大，但宏在两者上耗时大致相同。_

plug-⊢ 以推导作为参数，而宏如何写入该参数很重要。如方法 1 所示，它将其写作表达式

$$\mathsf{from\text{-}just}\ (\mathsf{search}\vdash 10\ \Delta\ \ldots),$$

也就是说，作为产生该推导的计算，而不是完整写出的推导，即由 $\vdash_{\mathsf{d}}$ 的构造子构成的显式项（例如

表 1。 自动解决的目标。 A, B, C, D, A_i 是任意公式， Γ 是任意上下文， t, t_0, t_1 是任意时间表达式。规则由其前提和结论给出。$(A_{i<k} \to A_{i+1})$ $\to$ φ 表示$(A_0 \to A_1)$ $\to \cdots \to (A_{k-1} \to A_k)$。燃料：解决目标所需的最小燃料。节点：使用燃料 10 探索的搜索树节点。大小：使用燃料 10 找到的推导中的规则应用次数。宏 和手动：分别使用方法 2 和方法 1 进行类型检查所需的时间（毫秒）。

名称	目标	燃料	节点	大小	宏	手动
id	$\vdash A \to A^t$	1	2	2	87	15
$\wedge$	$\vdash (A \wedge B) \to B \wedge A^t$	3	6	6	161	20
$\to$	$\vdash (A \to B) \to (B \to C) \to A \to C^t$	5	29	28	272	28
链式规则$_3$	$\vdash (A_i \to A_{i+1})_{i<3} \to A_0 \to A_3^t$	6	51	41	434	40
链式规则$_4$	$\vdash (A_i \to A_{i+1})_{i<4} \to A_0 \to A_4^t$	8	60	30	655	47
链式规则$_5$	$\vdash (A_i \to A_{i+1})_{i<5} \to A_0 \to A_5^t$	9	87	17	1408	57
链式规则$_6$	$\vdash (A_i \to A_{i+1})_{i<6} \to A_0 \to A_6^t$	10	175	20	1178	82
$\vee$	$\vdash (A \vee B) \to B \vee A^t$	3	12	6	156	20
分配律	$\vdash (A \vee B) \wedge (C \vee D) \to (A \wedge C \vee A \wedge D) \vee (B \wedge C \vee B \wedge D)^t$	7	620	25	1266	248
ctx	$(A \wedge B)^t \vdash B \vee \bot^t$	2	3	3	260	20
混合规则	$\vdash (\bigcirc A \wedge \mathrm{X}_0) \to (\mathrm{X}_0 \wedge \bigcirc A)^t$	3	6	6	212	20
next	$t \lhd t_1$, $A^{t_1} \vdash \bigcirc A^t$	1	2	3	253	22
事件。	$t \sqsubseteq t_1$, $B^{t_1} \vdash \Diamond B^t$	2	3	4	375	24
绑定变量	v_t, $(A \wedge B)^{v_t} \vdash B \wedge A^{v_t}$	2	5	5	354	30
$\bigcirc$R	$\Gamma \vdash t \lhd t_1$, $\Gamma \vdash A^{t_1}$ / $\Gamma \vdash \bigcirc A^t$	1	2	3	142	26
$\Diamond$R	$\Gamma \vdash t \sqsubseteq t_1$, $\Gamma \vdash A^{t_1}$ / $\Gamma \vdash \Diamond A^t$	2	3	5	164	28
$\Diamond$_R	$\Gamma \vdash t \sqsubseteq t_1$, $\Gamma \vdash t_1 \sqsubseteq t \bullet t_0$, $\Gamma \vdash A^{t_1}$ / $\Gamma \vdash \Diamond_{t_0} A^t$	5	7	10	270	48

ax、 $\to$R 和 $\to$L）。二者表示同一个推导。对于 $\varphi \to \varphi$，显式形式是 $\to$I ax，以及记录每一步上下文、注释和公式的所有隐式参数；这个形式很小，但对于更大的目标则并非如此（见下文）。对于该表达式，如第 5.3 节所述，Agda 会运行搜索，直到返回一个 just，但不会对其中包含的推导进行类型检查，也不会运行 plug-$\vdash$。相比之下，显式推导会作为一个项交给 Agda 进行类型检查，就像它被直接输入文件中一样，因此必须像手动书写的证明那样逐条规则应用地进行类型检查，包括记录每一步上下文和公式的隐式参数。对于大型推导，这一过程代价高昂。例如，考虑表 1 中的目标 $\to$，即 $\vdash (A \to B) \to (B \to C) \to A \to C^t$。使用 10 单位燃料时，找到的推导中包含一串冗余的 $\to$L 步骤。应用三次 $\to$R 后，目标为 C，搜索随后对假设 $B \to C$ 应用 $\to$L，这需要证明 B。为了证明 B，深度优先搜索又先尝试对 $B \to C$ 应用 $\to$L，这再次需要 B，如此反复，直到燃料耗尽，此时它改用 $A \to B$。这些嵌套尝试随后都成功了，因此最终全都进入了推导。该推导包含 28 次规则应用（而可行的最小燃料值对应的推导仅有 8 次），其引用的范式有

约 9,000 个节点。重新检查它约需 2 秒，而整个宏在改为给出 from-just 表达式时只需 0.44 秒。

这些链式规则 $\rightarrow$ （即链式规则$_2$）到链式规则$_6$ 展示了成本如何随目标大小增长。所需燃料随链的长度增加而增长：从 5 增至 10；探索的节点数从 29 增至 175；方法 1 的时间从 28 毫秒增至 82 毫秒；方法 2 的时间则从 0.27 秒增至 1.41 秒。相比之下，推导大小并非持续增长：链式规则$_3$ 的推导大小为 41，而链式规则$_6$的推导大小却只有 20。这是由于上文所述冗余的 $\rightarrow$L 步骤造成的，针对 $\rightarrow$而言更是如此。大小是在燃料设为 10 时测得的；目标所需燃料越少，留给深度优先搜索的燃料就越多，搜索会在找到正确假设前花费更多燃料绕行。在可行的最小燃料下，链式规则$_3$ 到链式规则$_6$，的大小分别为 11、14、17 和 20，即每增加一条链环，就多出三次规则应用。如果搜索失败，证明过程就会耗时很长；失败的原因可能是目标不可证，也可能是燃料不足。此时，搜索会在燃料限制内遍历整棵搜索树。由于左规则分支，搜索树的大小会随着燃料增加而呈指数级增长。例如，链式规则$_7$ 在燃料为 10 时无法求解：燃料为 8、9 和 10 时，搜索分别在探索 1,049、3,829 和 15,197 个节点后失败，耗时分别为 0.1、0.4 和 1.6 秒。这部分成本来自搜索，因此两种方法受到的影响相同。所有示例（见 https://github.com/jaeem006/TPTL-DIST/blob/main/Examples.lagda）的类型检查约需 52 秒，其中约 10 秒用于加载开发代码。

除了上述示例外，自动化工具还证明了引理 1 ⚙，即第 3.2 节中转发示例的主要性质。由于其证明使用了 $\Box$L，而这需要经典推理，因此我们扩展了 搜索$\vdash$ ，增加了一个可选参数，用于表示排中律，从而在搜索过程中启用经典 $\Box$ L 规则。

6 结论

我们介绍了 tptl-dist，这是一种用于验证容错分布式系统时间相关性质的演算。它扩展了带量词和通用比较算子的 TPTL $+$Past，并采用一种基于区间的序列演算，以精确刻画时间约束。tptl-dist 能够形式化并验证容错算法的非平凡性质，包括 Pistis 的一项关键性质。接下来，我们计划探索几个方向。

基于 tptl-dist，我们计划设计一种针对分布式系统独特需求的类型系统，使其不仅能够对系统性质进行建模，还能够对运行中的代码进行推理。

另一个方向是基于 PRISM 的架构，将现有的模型检查器和自动化验证技术应用于 tptl-dist。尽管 PRISM 可以对 tptl-dist 的子集进行模型检查，但这些技术需要经过调整，以处理 tptl-dist 的冻结算子和分布式算子。

最后，除了上述示例，我们还在 TPTL-DIST中形式化了另外两个问题：环中的领导者选举和拜占庭一致广播。二者均见附录，并留待未来使用第 5节介绍的自动化工具，在 Agda 中自动化其证明。

人工智能披露

Agda形式化由作者指导并审阅，但开发代码过程中也借助了人工智能。人工智能还用于纠正排版错误，并协助起草讨论该形式化的章节。所有此类文本均由作者审阅和编辑。

参考文献

[1] R. Alur 和 T. A. Henzinger。《一种真正的时序逻辑》。载于：《*ACM*学报》41.1（1994年1月），第181–203页。[2] C. Baier 和 J. Katoen。模型检测原理。麻省理工学院出版社，2008年。[3] M. Kwiatkowska、G. Norman 和 D. Parker。《PRISM 4.0：概率实时系统的验证》。载于：第*23*届国际计算机辅助验证会议（*CAV'11*）论文集。第6806卷。LNCS。施普林格，2011年，第585–591页。[4] L. Lamport。系统规约：*TLA+* 语言及其面向硬件和软件工程师的工具。波士顿：Addison-Wesley，2003年。[5] L. Lamport。《实时其实很简单》。技术报告MSR-TR-2005-30。2005年3月，第72页。[6] D. Cousineau、D. Doligez、L. Lamport、S. Merz、D. Ricketts 和 H. Vanzetto

TLA+ 证明。2012年。arXiv：1208.5933 [cs.SE].[7] G. Norman、D. Parker 和 X. Zou。“部分可观测概率实时系统的验证与控制”。载于：《定时系统的形式化建模与分析——第13届国际会议，FORMATS 2015》。第9268卷。《计算机科学讲义》。施普林格，2015年，第240–255页。[8] M. Oxford、D. Parker 和 M. Ryan。“证书透明度八卦协议的定量验证”。载于：第8届IEEE 通信与网络安全会议，CNS 2020，法国阿维尼翁，2020年6月29日至7月1日。IEEE，2020年，第1–9页。[9] T. Lu、S. Merz 和 C. Weidenbach。“使用 TLA+ 验证 Pastry 协议的探索”。2011年，第244–258页。[10] R. Joshi、L. Lamport、J. Matthews、S. Tasiran、M. R. Tuttle 和 Y. Yu。“使用 TLA+ 检查缓存一致性协议”。载于：《系统设计中的形式化方法》22.2（2003年），第125–131页。[11] W. J. Bolosky、J. R. Douceur 和 J. Howell。“Farsite项目：回顾”。载于：《操作系统评论》41.2（2007年），第17–26页。[12] C. Newcombe。“亚马逊为何选择 TLA+”。2014年，第25–39页。[13] C. Newcombe、T. Rath、F. Zhang、B. Munteanu、M. Brooker 和 M. Deardeuff。“亚马逊云科技如何运用形式化方法”。载于：《ACM通讯》58.4（2015年），第66–73页。[14] S. Chand、Y. A. Liu 和 S. D. Stoller。“分布式共识的 Multi-Paxos 形式化验证”。载于：FM 2016：形式化方法——第21届国际研讨会。第9995卷。2016年，第119–136页。

[15] S. Gao、B. Zhan、D. Liu、X. Sun、Y. Zhi、D. N. Jansen 和 L. Zhang。“Taurus分布式数据库中的共识形式化验证”。载于：形式化方法。2021年，第741–751页。[16] D. Kozhaya、J. Decouchant、V. Rahli 和 P. E. Veríssimo。“PISTIS：事件触发的实时拜占庭容错协议套件”。载于：*IEEE*并行与分布式系统汇刊32.9（2021年），第2277–2290页。[17] J. E. Mendoza、S. Speight 和 V. Rahli。“tptl-dist：用于验证实时分布式系统的演算”。载于：分布式对象、组件与系统形式化技术——第*46*届 *IFIP WG 6.1* 国际会议 *FORTE 2026*，作为第*21*届分布式计算技术国际联合会议 *DisCoTec 2026* 的一部分举行，意大利乌尔比诺，*2026*年*6*月*8*日至*12*日，会议录。第16589卷。《计算机科学讲义》。施普林格，2026年，第197–223页。[18] M. Volpe。“时态逻辑的标记自然演绎”。博士论文。意大利维罗纳大学，2010年。[19] H. Kopetz。实时系统：分布式嵌入式应用的设计原则。第1版。美国：Kluwer Academic Publishers，1997年。[20] H. Kopetz。“可信系统之系统为何需要全局时间”。载于：*CoRR*abs/1404.6772（2014年）。arXiv：1404.6772。[21] H. Hansson 和 B. Jonsson。“关于时间与可靠性推理的逻辑”。载于：《计算的形式化方面》6.5（1994年），第512–535页。[22] A. Bianco 和 L. de Alfaro。“概率系统与非确定性系统的模型检测”。载于：第*15*届软件技术基础与理论计算机科学会议论文集（*FSTTCS'95*）。第1026卷。LNCS。施普林格，1995年，第499–513页。[23] A. Aziz、K. Sanwal、V. Singhal 和 R. Brayton。“连续时间马尔可夫链的验证”。载于：第*8*届计算机辅助验证国际会议论文集（*CAV'96*）。第1102卷。LNCS。施普林格，1996年7月，第269–276页。[24] C. Baier、J.-P. Katoen 和 H. Hermanns。“连续时间马尔可夫链的近似符号模型检测”。载于：第*10*届并发理论国际会议论文集（*CONCUR'99*）。第1664卷。LNCS

埃因霍温：施普林格，1999年8月，第146–161页。[25] L. Bozzelli、A. Murano 和 L. Sorrentino。“具有线性过去的交替时间时序逻辑”。载于：《理论计算机科学》813（2020年4月），第199–217页。[26] R. Fagin、J. Y. Halpern、Y. Moses 和 M. E. Vardi。关于知识的推理。马萨诸塞州剑桥：麻省理工学院出版社，1995年。[27] F. Wehrung。“单射正序幺半群 I”。载于：《纯粹与应用代数杂志》83.1（1992年），第43–82页。[28] T. Fritz。“资源可转换性与有序交换幺半群”。载于：《计算机科学中的数学结构》27.6（2017年），第850–938页。

[29] R. Kuznets、L. Prosperi、U. Schmid 和 K. Fruzsa。“与拜占庭故障代理进行认知推理”。载于：组合系统前沿。第11715卷。沙姆：施普林格国际出版公司，2019年，第259–276页。[30] A. Baltag、B. Renne 和 S. Smets。“正当信念、显式知识与确凿证据逻辑”。载于：纯粹与应用逻辑年刊 165.1（2014年1月），第49–81页。[31] A. Baltag、L. S. Moss 和 S. Solecki。“公共公告、共同知识与私人猜疑逻辑（扩展摘要）”。载于：()。[32] A. Bolotov、A. Basukoski、O. Grigoriev 和 V. Shangin。“线性时态逻辑的自然演绎演算”。载于：人工智能中的逻辑。柏林，海德堡：施普林格·柏林·海德堡，2006年，第56–68页。[33] A. Bolotov、O. Grigoriev 和 V. Shangin。“命题线性时态逻辑的自动化自然演绎”。载于：第14届国际时间表示与推理研讨会（TIME’07）。2007年，第47–58页。[34] A. Masini、L. Viganò 和 M. Volpe。“Until 的历史”。载于：Electron. Notes Theor. Comput. Sci. 262（2010年5月），第189–204页。[35] D. M. Gabbay。带标签演绎系统。克拉伦登出版社，1996年。[36] A. K. Simpson。“直觉主义模态逻辑的证明论与语义”。博士论文。英国爱丁堡大学，1994年。[37] D. Basin、C. Caleiro、J. Ramos 和 L. Viganò。“分布式时态逻辑的带标签语义树”。载于：逻辑与计算杂志 19.6（2009年7月），第1245–1279页。电子版：https://academic.oup.com/logcom/article-pdf/19/6/1245/2805892/exp022.pdf。[38] L. Viganò 和 M. Volpe。时态逻辑族的带标签自然演绎系统。2008年。arXiv：0803。3187 [cs.LO][39] C. Cachin、R. Guerraoui 和 L. Rodrigues。可靠安全的分布式编程导论。柏林，海德堡：施普林格·柏林·海德堡，2011年。[40] G. Bracha。“异步拜占庭一致协议”。载于：Inf. Com- put. 75.2（1987年），第130–143页。[41] D. Kozhaya、J. Decouchant 和 P. Esteves-Verissimo。“RT-ByzCast：拜占庭

弹性实时可靠广播”。载于：*IEEE*计算机学报 68.3（2019），第440–454页。[42] G. L. Lann. “分布式系统——迈向形式化方法”。载于：信息处理：第七届*IFIP*大会（*1977*）论文集，多伦多，加拿大，*1977*年*8*月*8*日至*12*日。北荷兰，1977年，第155–160页。[43] E. J. H. Chang和R. Roberts. “进程的环形配置中的分布式极值查找改进算法”。载于：*ACM*通讯 22.5（1979），第281–283页。

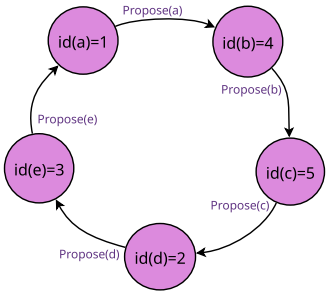

图 7. 环中的领导者选举

A 更多示例

除了第3.2节介绍的转发（FW）示例外，我们还通过另外两个简单示例进一步说明 TPTL-DIST 的相关方面：第A.1节的环中的领导者选举（LER），以及第A.2节的拜占庭一致广播（BCB）。这三个示例都展示了如何在分布式系统中刻画消息传递模式，其中代理通过发送和接收消息来交换信息。此外，LER和BCB示例涉及逻辑上的定性时间概念，而FW示例则借助形如 $\Diamond_t \varphi$ 的公式，涉及物理上的定量时间概念。最后，LER示例涉及一种典型的归纳证明性质，而BCB则涉及一种典型模式：只有在收到系统中其他代理的足够多消息后，代理才能取得进展。

A.1 环中的领导者选举

环中的领导者选举问题涉及从一组按环形组织的代理中选出一位领导者。每个代理都有一个唯一标识符（id），目标是选出标识符最高的代理。代理并不了解整个环，消息只能按照固定方向（例如顺时针）传递给相邻节点。领导者选举对于分布式系统中的协调至关重要，例如数据库复制或时间同步。

算法必须满足的一个关键属性是活性，它确保所有代理最终都能收到选举结果，即领导者的身份：

$$\forall a : \textbf{AGENT}.\exists b, c : \textbf{AGENT}.@_{\mathbf{0}} \Diamond \mathtt{recv}(a, \mathtt{Elected}(c), b) \tag{8}$$

@ 运算符用于表示某个公式在特定时间点为真：$@_t \varphi \coloneqq x \cdot (\varphi \wedge x \equiv t)$。Propose($a$) 和 Elected($a$) 都是数据，分别表示 a 已被提议担任领导者，以及已当选为领导者。

现在，我们将经典环选举算法形式化为 [42, 43]在TPTL-DIST中的实现。我们假设不存在故障，系统是同步系统，并且有一个定义在代理上的二元谓词$>$，其中 $a > b$ 表示 a 的标识符高于 b，以及一个函数$\texttt{nhbr} : \textbf{主体} \to \textbf{主体}$，用于返回环中每个主体的邻居。

初始化：每个主体 a 都将自己提议为领导者，并在提议中向其邻居发送自己的标识符（见图7）：

$$\forall a : \textbf{AGENT}.@_0\texttt{send}(a, \texttt{Propose}(a), [\texttt{nhbr}(a)]) \tag{9}$$

提议传播：当主体 a 接收到一条 Propose(c) 消息时，会将 c 的标识符与自己的标识符进行比较。如果 c 的标识符大于 a 的，a 就简单地将该消息转发给其邻居；如果 c 的标识符小于 a 的，a 则将自己的标识符转发给其邻居：

$$\begin{aligned}\forall a, b, c : \textbf{AGENT}.\Box(&\texttt{recv}(a, \texttt{Propose}(c), b) \to \\ &(c > a \to \texttt{send}(a, \texttt{Propose}(c), [\texttt{nhbr}(a)])) \wedge \\ &(a > c \to \texttt{send}(a, \texttt{Propose}(a), [\texttt{nhbr}(a)]))\end{aligned} \tag{10}$$

领导者确定：如果 a 接收到包含自身标识符的提议，这意味着 a 具有最高标识符。随后，它会在环中发送一条 Elected(a) 消息，以宣布自己当选为领导者。

$$\forall a,\ b : \textbf{主体}.\Box(\text{接收}(a,\ \text{Propose}(a)\ b) \to \text{发送}(a,\ \text{Elected}(a), [\text{邻居}(a)])) \tag{11}$$

领导者传播：当主体 a 接收到一条 Elected(b) 消息时，它知道 b 是领导者，并将该消息转发给其邻居（除非 a是 b）：

$$\begin{aligned}&\forall a,\ b,\ c : \textbf{主体}。\\ &\Box(\texttt{recv}(a, \texttt{Elected}(b), c) \to b > a \to \texttt{send}(a, \texttt{Elected}(b), [\texttt{nhbr}(a)]))\end{aligned} \tag{12}$$

当领导者收到一条包含其自身标识符的 Elected 消息时，算法即告结束。

通过对时间进行归纳，利用以下不变式证明活性（选举终止）：在时间 x，如果 a 尚未收到通知其为领导者的消息（领导者算子），那么它必定正在接收其在环中的 x 个前驱中标识符最高者的消息（rcvMaxId 算子）；其中，max(a, x)表示环中 a的 x 个前驱里标识符最高的主体（当 x 为 0 时，它返回 a ）：

$$\forall a : \textbf{AGENT}.\ \bigcirc\ \Box(x \cdot \neg\texttt{leader}(a) \to \texttt{rcvMaxId}(a, \texttt{max}(a, x))) \tag{13}$$

其中

$$\begin{aligned}\texttt{leader}(a) \quad &:\equiv \exists c : \textbf{AGENT}.\blacklozenge\texttt{recv}(a, \texttt{Propose}(a), c) \\ \texttt{rcvMaxId}(a, b) &:\equiv \exists c : \textbf{AGENT}.\texttt{recv}(a, \texttt{Propose}(b), c)\end{aligned}$$

该不变式允许得出结论：选举阶段将在 N 步后终止，其中 N 是环中代理的数量，且所有代理都已收到

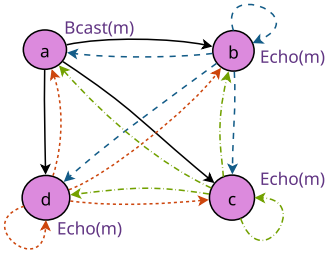

图8。 拜占庭一致广播

最高标识符的代理/领导者，包括领导者自身。式（11）随后可用于让领导者开始最终阶段，而式（12）保证`Elected`消息将沿环传回领导者，标志着算法执行的结束。

我们进一步考虑代理以同步步进的方式推进，这意味着每条消息恰好需要一个时间单位才能到达其目的地：

$$\forall a, b : \textsc{Agent}.\forall d : \textsc{Data}.\Box(\texttt{send}(a, d, [b]) \rightarrow \bigcirc\, \texttt{recv}(b, d, a)) \qquad (14)$$

现在让我们概述上述不变式的证明。在时间 $x = 0$时，$\texttt{max}(a, x) = a$的基础情况下，不变式成立，因为根据式（9），假设每个代理在初始化时都已收到针对自身的提议。

对于归纳步骤，假设该性质在时间 x 时成立。那么，在时间$x+1$时，代理 a 的前驱（记为 b）如果还不是领导者，就已经收到 $\max(b, x)$，即 x 前驱中标识符最大的代理，该代理属于 b的前驱。当 b收到此信息时，它会将自身标识符与 $\max(b, x)$ 进行比较。如果 $b > \max(b, x)$或 $b < \max(b, x)$，则分别应用式（10），向 a发送 $\text{Propose}(b)$ 或 $\text{Propose}(\max(b, x))$。在这两种情况下， a 都会根据式（14）在时间 $x + 1$ 时收到 $\max(a, x + 1)$。

A.2 拜占庭一致广播

拜占庭一致广播 [39, 第3节。10] （BCB）是一种基本的广播原语，专为拜占庭故障模型设计。在该模型中，故障代理会任意偏离其规格说明，甚至故意试图阻止算法实现其目标。它要求 $3f + 1$ 代理能够容忍最多f 个故障代理。

每个 BCB实例都有一个唯一的指定发送者，负责发起消息的广播。BCB 保证有效性 属性，即如果正确代理a 广播消息 m，那么每个正确代理最终都会 Deliver m。在 tptl-dist 中，该属性表述如下，其中 $\text{Bcast}(m)$，$\text{Echo}(m)$， $\text{Deliver}(m)$ $\in$**数据**分别表示消息 m （也是一个数据片段）的广播、回显和传递的一部分

数据）的相应操作：

$$\begin{aligned}\forall a, b : \textbf{AGENT}.\forall m : \textbf{DATA}.\texttt{correct}(a) \to \texttt{correct}(b) \to \\ \Box(\texttt{internal}(a, \texttt{Bcast}(m)) \to \Diamond\texttt{internal}(b, \texttt{Deliver}(m)))\end{aligned} \tag{15}$$

广播：主体通过形如 internal(a,Bcast(m)) 的内部广播事件发起广播。发起广播时，主体会向所有其他代理发送一条广播消息，其中 N 是系统中代理的集合：

$$\begin{aligned}\forall a : \textbf{AGENT}.\forall m : \textbf{DATA}.\,\texttt{correct}(a) \to \\ \Box(\texttt{internal}(a, \texttt{Bcast}(m)) \to \texttt{send}(a, \texttt{Bcast}(m), N))\end{aligned} \tag{16}$$

Echo：主体收到广播消息后，会向所有其他代理发送一条回声消息：

$$\begin{aligned}\forall a, b : \textbf{AGENT}.\forall m : \textbf{DATA}.\,\texttt{correct}(a) \to \\ \Box(\texttt{recv}(a, \texttt{Bcast}(m), b) \to \texttt{send}(a, \texttt{Echo}(m), N))\end{aligned} \tag{17}$$

Deliver：代理收到 $2f+1$ 条回声后即交付：

$$\begin{aligned}\forall a : \textbf{AGENT}.\forall m : \textbf{DATA}.\Box(\,\texttt{correct}(a) \to \\ \left(\begin{array}{l}\exists B : \textbf{AGENTS}. \\ \quad \texttt{size}(B, 2f+1) \wedge \\ \quad \forall b : \textbf{AGENT}.\, b \in B \to \\ \qquad \Diamond\texttt{recv}(a, \texttt{Echo}(m), b)\end{array}\right) \to \\ \Diamond\texttt{internal}(a, \texttt{Deliver}(m)))\end{aligned} \tag{18}$$

可靠通信：我们假设通信可靠，即正确代理发送的任何消息最终都会被所有其他正确代理接收：

$$\begin{aligned}\forall a : \textbf{AGENT}.\forall d : \textbf{DATA}.\forall B : \textbf{AGENTS}. \\ \Box(\texttt{correct}(a) \to \texttt{send}(a, d, B) \to \\ \forall b : \textbf{AGENT}.\, b \in B \to \\ \texttt{correct}(b) \to \Diamond\texttt{recv}(b, d, a))\end{aligned} \tag{19}$$

故障阈值：最后，我们假设在 $3f+1$ 个代理中至少有 $2f+1$ 个正确代理，其中下文的 P 是定义在代理上的谓词：

$$\begin{aligned}\Box(\forall b : \textbf{AGENT}.\texttt{correct}(b) \to P(b)) \to \\ \exists B : \textbf{AGENTS}.\,\texttt{size}(B, 2f+1) \wedge \\ \forall b : \textbf{AGENT}.b \in B \to \texttt{correct}(b) \wedge P(b))\end{aligned} \tag{20}$$

图 8 展示了一个系统中的 BCB 执行，该系统包含 $3f+1 = 4$ 个代理，且至多有 $f = 1$ 个拜占庭代理。主体 a 广播消息 m，随后所有代理都对其进行回显（b对应蓝色虚线，c对应绿色点划线，d对应橙色点线）。在收到 $2f+1 = 3$ 个来自 b、c、d的回显后，主体 b例如，会交付消息。

现在我们简要说明有效性属性的证明。设正确代理 a和 b 满足： a 发起了对 m的广播，即 internal(a,Bcast(m)) 成立。我们需要证明， b 最终会交付 m，即 $\Diamond$ internal(b, Deliver(m))。根据等式 (16)， a 也会向 N中的所有代理发送 Bcast(m)。根据等式 (19)， N 中的所有正确代理都会收到 Bcast(m)；根据等式 (17)，它们会向 N 中的所有其他代理发送 Echo(m) 消息，其中包括 b。因此，根据等式 (19)， b 会收到来自所有正确代理的 Echo(m)。最后，根据等式 (18) 和等式 (20)，主体 b 会交付 m。

B 规则全集

命题逻辑 tptl-dist 的命题逻辑规则如下：

$$\frac{\Gamma, \Delta \vdash \mathsf{a}_1}{\Gamma, \bot^{\mathsf{a}_1}, \Delta \vdash \varphi^{\mathsf{a}}}\ \bot\mathrm{L} \qquad \frac{\Gamma \vdash \varphi^{\mathsf{a}} \quad \Gamma \vdash \psi^{\mathsf{a}}}{\Gamma \vdash \varphi \wedge \psi^{\mathsf{a}}}\ \wedge\mathrm{R} \qquad \frac{\Gamma, \varphi^{\mathsf{a}}, \psi^{\mathsf{a}}, \Delta \vdash \delta^{\mathsf{a}_1}}{\Gamma, \varphi \wedge \psi^{\mathsf{a}}, \Delta \vdash \delta^{\mathsf{a}_1}}\ \wedge\mathrm{L}$$

$$\frac{\Gamma \vdash \varphi^{\mathsf{a}}}{\Gamma \vdash \varphi \vee \psi^{\mathsf{a}}}\ \vee\mathrm{R}_1 \qquad \frac{\Gamma \vdash \psi^{\mathsf{a}}}{\Gamma \vdash \varphi \vee \psi^{\mathsf{a}}}\ \vee\mathrm{R}_2 \qquad \frac{\Gamma, \varphi^t, \Delta \vdash \delta^{\mathsf{a}} \quad \Gamma, \psi^t, \Delta \vdash \delta^{\mathsf{a}}}{\Gamma, \varphi \vee \psi^t, \Delta \vdash \delta^{\mathsf{a}}}\ \vee\mathrm{L}$$

$$\frac{}{\Gamma \vdash \top^{\mathsf{a}}}\ \top\mathrm{R} \qquad \frac{\Gamma, \varphi^t \vdash \psi^t}{\Gamma \vdash \varphi \to \psi^t}\ \to\mathrm{R} \qquad \frac{\Gamma, \Delta \vdash \varphi^t \quad \Gamma, \psi^t, \Delta \vdash \delta^{\mathsf{a}}}{\Gamma, \varphi \to \psi^t, \Delta \vdash \delta^{\mathsf{a}}}\ \to\mathrm{L}$$

tptl-dist 还包含以下规则。使用假设的“左”规则（或消去规则）时，如果希望保留该假设，我们通常会默默使用割 规则来复制该假设，因为 tptl-dist 的“左”规则会丢弃被消去的公式。

$$\frac{\Gamma, \Delta \vdash \varphi^{\mathsf{a}}}{\Gamma, \psi^{\mathsf{a}_1}, \Delta \vdash \varphi^{\mathsf{a}}}\ \mathit{thin} \qquad \frac{\Gamma \vdash \psi^{\mathsf{a}_1} \quad \Gamma, \psi^{\mathsf{a}_1} \vdash \varphi^{\mathsf{a}}}{\Gamma \vdash \varphi^{\mathsf{a}}}\ \mathit{cut}$$

$$\frac{}{\Gamma, \varphi^{\mathsf{a}} \vdash \varphi^{\mathsf{a}}}\ \mathit{ax} \qquad \frac{}{\Gamma \vdash i \in [\ldots, i, \ldots]^{\mathsf{a}}}\ \in\mathrm{R}$$

非空注释

$$\frac{}{\Gamma \vdash \centerdot} \qquad \frac{\Gamma \vdash t_1 \sqsubseteq t_2}{\Gamma \vdash [t_1, t_2]} \qquad \frac{\Gamma \vdash t_1 \sqsubset t_2}{\Gamma \vdash [t_1, t_2)}$$

$$\frac{}{\Gamma \vdash t} \qquad \frac{\Gamma \vdash t_1 \sqsubset t_2}{\Gamma \vdash (t_1, t_2]} \qquad \frac{\Gamma \vdash t_1 \sqsubset t \quad \Gamma \vdash t \sqsubset t_2}{\Gamma \vdash (t_1, t_2)}$$

时序逻辑规则 在以下规则中， v_t 是一个“新鲜”变量：

$$\frac{\Gamma \vdash t \lhd t_1 \quad \Gamma \vdash \varphi^{t_1}}{\Gamma \vdash \bigcirc \varphi^t}\ \bigcirc\mathrm{R} \qquad \frac{\Gamma \vdash t \sqsubseteq t_1 \quad \Gamma \vdash \psi^{t_1} \quad \Gamma \vdash \varphi^{[t,t_1)}}{\Gamma \vdash \varphi \,\mathbf{U}\, \psi^t}\ \mathbf{U}\mathrm{R}$$

$$\frac{\Gamma, t \lhd v_t, \varphi^{v_t}, \Delta \vdash \delta^{\mathsf{a}}}{\Gamma, \bigcirc \varphi^t, \Delta \vdash \delta^{\mathsf{a}}}\ \bigcirc\mathrm{L} \qquad \frac{\Gamma, t \sqsubseteq v_t, \psi^{v_t}, \varphi^{[t,v_t)}, \Delta \vdash \delta^{\mathsf{a}}}{\Gamma, \varphi \,\mathbf{U}\, \psi^t, \Delta \vdash \delta^{\mathsf{a}}}\ \mathbf{U}\mathrm{L}$$

$$\frac{\Gamma \vdash t_1 \lhd t \quad \Gamma \vdash \varphi^{t_1}}{\Gamma \vdash \mathbf{Y}\, \varphi^t}\ \mathbf{Y}\mathrm{R} \qquad \frac{\Gamma \vdash t_1 \sqsubseteq t \quad \Gamma \vdash \psi^{t_1} \quad \Gamma \vdash \varphi^{(t_1,t]}}{\Gamma \vdash \varphi \,\mathbf{S}\, \psi^t}\ \mathbf{S}\mathrm{R}$$

$$\frac{\Gamma, v_t \lhd t, \varphi^{v_t}, \Delta \vdash \delta^{\mathsf{a}}}{\Gamma, \mathbf{Y}\, \varphi^t, \Delta \vdash \delta^{\mathsf{a}}}\ \mathbf{Y}\mathrm{L} \qquad \frac{\Gamma, v_t \sqsubseteq t, \psi^{v_t}, \varphi^{(v_t,t]}, \Delta \vdash \delta^{\mathsf{a}}}{\Gamma, \varphi \,\mathbf{S}\, \psi^t, \Delta \vdash \delta^{\mathsf{a}}}\ \mathbf{S}\mathrm{L}$$

定时规则 比较运算符和冻结运算符的规则如下所示。这里只展示部分比较规则，因为它们反映了图 2 中的公理。此外，TPTL-DIST 的 ()L、]L、(]L、 []L、()R、]R、(]R 和 []R 规则允许实例化区间，下面展示其中一些（其余形式相同）。

$$\frac{}{\Gamma, t_1 \sqsubseteq t_2, \Delta \vdash t_1 \sqsubseteq t_2^{\mathsf{a}}}\ ax_{\sqsubseteq} \qquad \frac{\Gamma \vdash \varphi[v_t \backslash t]^t}{\Gamma \vdash v_t \cdot \varphi^t}\ \cdot\mathrm{R}$$

$$\frac{}{\Gamma \vdash t_1 \bullet t_2 \equiv t_2 \bullet t_1^{\mathsf{a}}}\ \bullet sym \qquad \frac{\Gamma, \varphi[v_t \backslash t]^t, \Delta \vdash \psi^{\mathsf{a}}}{\Gamma, v_t \cdot \varphi^t, \Delta \vdash \psi^{\mathsf{a}}}\ \cdot\mathrm{L}$$

$$\frac{\Gamma, \varphi^t, \Delta \vdash \psi^{\mathsf{a}} \qquad \Gamma, \Delta \vdash t_1 \sqsubseteq t \qquad \Gamma, \Delta \vdash t \sqsubseteq t_2}{\Gamma, \varphi^{[t_1,t_2]}, \Delta \vdash \psi^{\mathsf{a}}}\ []\mathrm{L}$$

$$\frac{\Gamma, t_1 \sqsubseteq v_t, v_t \sqsubseteq t_2 \vdash \phi^{v_t}}{\Gamma \vdash \phi^{[t_1,t_2]}}\ []\mathrm{R}$$

谓词逻辑规则 谓词规则如下，其中 v' 是一个“新鲜”变量， X 是类型为 T 的项：

$$\frac{\Gamma \vdash \varphi[v \backslash v']^{\mathsf{a}}}{\Gamma \vdash \forall v : T.\varphi^{\mathsf{a}}}\ \forall\mathrm{R} \qquad \frac{\Gamma, \varphi[v \backslash X]^{\mathsf{a}_1}, \Delta \vdash \psi^{\mathsf{a}}}{\Gamma, \forall v : T.\varphi^{\mathsf{a}_1}, \Delta \vdash \psi^{\mathsf{a}}}\ \forall\mathrm{L}$$

$$\frac{\Gamma \vdash \varphi[v \backslash X]^{\mathsf{a}}}{\Gamma \vdash \exists v : T.\varphi^{\mathsf{a}}}\ \exists\mathrm{R} \qquad \frac{\Gamma, \varphi[v \backslash v']^t, \Delta \vdash \psi^{\mathsf{a}}}{\Gamma, \exists v : T.\varphi^t, \Delta \vdash \psi^{\mathsf{a}}}\ \exists\mathrm{L}$$

归纳 归纳规则如下，其中 v_t 不在 Γ 中自由出现：

$$\frac{\Gamma, v_t \cdot \varphi^{[\mathbf{0}, v_t)} \vdash \varphi^{v_t}}{\Gamma \vdash \varphi^{v_t}}\ ind$$

经典规则

$$\frac{}{\Gamma \vdash \varphi \vee \neg\varphi^{\mathsf{a}}}\ lem$$

导出规则 在以下所有规则中， v_t 表示一个“新鲜”的变量。

$$\frac{\Gamma \vdash \varphi^{t_1} \qquad \Gamma \vdash t \sqsubseteq t_1}{\Gamma \vdash \Diamond\varphi^t}\ \Diamond\mathrm{R} \qquad \frac{\Gamma, \varphi^{v_t}, t_1 \sqsubseteq v_t, \Delta \vdash \psi^t}{\Gamma, \Diamond\varphi^{t_1}, \Delta \vdash \psi^t}\ \Diamond\mathrm{L}$$

$$\frac{\Gamma, t \sqsubseteq v_t \vdash \varphi^{v_t}}{\Gamma \vdash \Box\varphi^t}\ \Box\mathrm{R} \qquad \frac{\Gamma, \varphi^{t_2}, \Delta \vdash \psi^t \qquad \Gamma, \Delta \vdash t_1 \sqsubseteq t_2}{\Gamma, \Box\varphi^{t_1}, \Delta \vdash \psi^t}\ \Box\mathrm{L}$$

$$\frac{\Gamma \vdash \varphi^{t_1} \qquad \Gamma \vdash t \sqsubseteq t_1 \qquad \Gamma \vdash t_1 \sqsubseteq t_0 \bullet t}{\Gamma \vdash \Diamond_{t_0}\varphi^t}\ \Diamond\_\mathrm{R}$$

$$\frac{\Gamma, \varphi^{v_t}, t_1 \sqsubseteq v_t, v_t \sqsubseteq t_0 \bullet t_1, \Delta \vdash \psi^t}{\Gamma, \Diamond_{t_0}\varphi^{t_1}, \Delta \vdash \psi^t}\ \Diamond\_\mathrm{L}$$

$$\frac{\Gamma, t \sqsubseteq v_t, v_t \sqsubseteq t_0 \bullet t \vdash \varphi^t}{\Gamma \vdash \Box_{t_0}\varphi^t}\ \Box\_\mathrm{R}$$

$$\frac{\Gamma, \varphi^{t_2}, \Delta \vdash \psi^t \qquad \Gamma, \Delta \vdash t_1 \sqsubseteq t_2 \qquad \Gamma, \Delta \vdash t_2 \sqsubseteq t_0 \bullet t_1}{\Gamma, \Box_{t_0}\varphi^{t_1}, \Delta \vdash \psi^t}\ \Box\_\mathrm{L}$$

$$\frac{\Gamma \vdash \varphi^{t_1} \qquad \Gamma \vdash t_1 \sqsubseteq t}{\Gamma \vdash \blacklozenge\varphi^t}\ \blacklozenge\mathrm{R} \qquad \frac{\Gamma, \varphi^{v_t}, v_t \sqsubseteq t_1, \Delta \vdash \psi^t}{\Gamma, \blacklozenge\varphi^{t_1}, \Delta \vdash \psi^t}\ \blacklozenge\mathrm{L}$$

$$\frac{\Gamma, v_t \sqsubseteq t \vdash \varphi^{v_t}}{\Gamma \vdash \blacksquare\varphi^t}\ \blacksquare\mathrm{R} \qquad \frac{\Gamma, \varphi^{t_2}, \Delta \vdash \psi^t \qquad \Gamma, \Delta \vdash t_2 \sqsubseteq t_1}{\Gamma, \blacksquare\varphi^{t_1}, \Delta \vdash \psi^t}\ \blacksquare\mathrm{L}$$

$$\frac{\Gamma \vdash \varphi^{t_1} \qquad \Gamma \vdash t_1 \sqsubset t}{\Gamma \vdash \blacktriangleleft\varphi^t}\ \blacktriangleleft\mathrm{R} \qquad \frac{\Gamma, \varphi^{v_t}, v_t \sqsubset t_1, \Delta \vdash \psi^t}{\Gamma, \blacktriangleleft\varphi^{t_1}, \Delta \vdash \psi^t}\ \blacktriangleleft\mathrm{L}$$

C 关于公式的证明(3)

本节将完整给出公式（3）的证明，其概要已在第 3.6 节中介绍。

Let

$$\begin{aligned}
&\Gamma_0 \coloneqq (eq.\ (1))^t, (eq.\ (2))^t\\
&\Gamma_1 \coloneqq \Gamma_0, t \sqsubseteq v_t, \mathtt{send}(a,d,[b])^{v_t}, \mathtt{recv}(b,d,a)^{v_t'}, v_t \sqsubseteq v_t', v_t' \sqsubseteq T \bullet v_t\\
&\Gamma_2 \coloneqq \Gamma_1, \mathtt{send}(b,d,[c])^{v_t'}\\
&\Gamma_3 \coloneqq \Gamma_2, \mathtt{recv}(c,d,b)^{v_t''}, v_t' \sqsubseteq v_t'', v_t'' \sqsubseteq T \bullet v_t'
\end{aligned}$$

为证明 $\Gamma_0 \vdash eq.(3)^t$，我们先进行一些引入步骤，以特别得到以下假设： 发送$(a,d,[b])^{v_t}$:

$$\cfrac{\cfrac{\cfrac{\Gamma_0, t \sqsubseteq v_t, \mathtt{send}(a,d,[b])^{v_t} \vdash \Diamond_{2T}\mathtt{recv}(c,d,b)^{v_t}}{\Gamma_0, t \sqsubseteq v_t \vdash \mathtt{send}(a,d,[b]) \to \Diamond_{2T}\mathtt{recv}(c,d,b)^{v_t}}\ \to\mathrm{R}}{\Gamma_0 \vdash \Box(\mathtt{send}(a,d,[b]) \to \Diamond_{2T}\mathtt{recv}(c,d,b))^t}\ \Box\mathrm{R}}{\Gamma_0 \vdash eq.\ (3)^t}\ \forall\mathrm{R}(\times 4)$$

现在，我们将公式（1）具体化，以特别得到以下假设 接收$(b,d,a)^{v_t'}$:

$$\cfrac{\cfrac{\cfrac{\cfrac{\Gamma_1 \vdash \Diamond_{2T}\mathtt{recv}(c,d,b)^{v_t}}{\Gamma_0, t \sqsubseteq v_t, \mathtt{send}(a,d,[b])^{v_t}, \phi_3^{v_t'}, v_t \sqsubseteq v_t', v_t' \sqsubseteq T \bullet v_t \vdash \Diamond_{2T}\mathtt{recv}(c,d,b)^{v_t}}\ \to\mathrm{L}\ \&\ \in\mathrm{R}}{\Gamma_0, t \sqsubseteq v_t, \mathtt{send}(a,d,[b])^{v_t}, \phi_2^{v_t} \vdash \Diamond_{2T}\mathtt{recv}(c,d,b)^{v_t}}\ \Diamond\_\mathrm{L}}{\Gamma_0, t \sqsubseteq v_t, \mathtt{send}(a,d,[b])^{v_t}, \phi_1^{v_t} \vdash \Diamond_{2T}\mathtt{recv}(c,d,b)^{v_t}}\ \to\mathrm{L}\ \&\ ax}{\Gamma_0, t \sqsubseteq v_t, \mathtt{send}(a,d,[b])^{v_t} \vdash \Diamond_{2T}\mathtt{recv}(c,d,b)^{v_t}}\ \forall\mathrm{L}(\times 4)\ \&\ \Box\mathrm{L}\ \&\ ax_{\sqsubseteq}$$

其中

$$\begin{aligned}
&\phi_1 \coloneqq \mathtt{send}(a,d,[b]) \to \phi_2\\
&\phi_2 \coloneqq \Diamond_T(\phi_3)\\
&\phi_3 \coloneqq b \in [b] \to \mathtt{recv}(b,d,a)
\end{aligned}$$

我们将等式（2）具体化，特别得到假设 发送$(b,\ d,\ [c])^{v_t'}$：

$$\dfrac{\dfrac{\Gamma_2 \vdash \Diamond_{2T}\mathtt{recv}(c,d,b)^{v_t}}{\Gamma_1, \phi_4^{v_t'} \vdash \Diamond_{2T}\mathtt{recv}(c,d,b)^{v_t}}\ {\to}\mathrm{L}\ \&\ ax \quad \dfrac{}{\Gamma_1 \vdash t \sqsubseteq v_t'^{v_t}}\ \sqsubseteq trans\ \&\ ax_{\sqsubseteq}}{\Gamma_1 \vdash \Diamond_{2T}\mathtt{recv}(c,d,b)^{v_t}}\ \forall\mathrm{L}(\times 4)\ \&\ \Box\mathrm{L}$$

其中

$$\phi_4 \coloneqq \mathtt{recv}(b,d,a) \to \mathtt{send}(b,d,[c])$$

我们再次实例化等式（1），特别得到假设 接收$(c,\ d,\ b)^{v_t''}$：

$$\dfrac{\dfrac{\dfrac{\dfrac{\Gamma_3 \vdash \Diamond_{2T}\mathtt{recv}(c,d,b)^{v_t}}{\Gamma_2, \phi_7^{v_t''}, v_t' \sqsubseteq v_t'', v_t'' \sqsubseteq T \bullet v_t' \vdash \Diamond_{2T}\mathtt{recv}(c,d,b)^{v_t}}\ {\to}\mathrm{L}\ \&\ {\in}\mathrm{R}}{\Gamma_2, \phi_6^{v_t'} \vdash \Diamond_{2T}\mathtt{recv}(c,d,b)^{v_t}}\ \Diamond\_\mathrm{L}}{\Gamma_2, \phi_5^{v_t'} \vdash \Diamond_{2T}\mathtt{recv}(c,d,b)^{v_t}}\ {\to}\mathrm{L}\ \&\ ax \quad \dfrac{}{\Gamma_1 \vdash t \sqsubseteq v_t'^{v_t}}\ \sqsubseteq trans\ \&\ ax_{\sqsubseteq}}{\Gamma_2 \vdash \Diamond_{2T}\mathtt{recv}(c,d,b)^{v_t}}\ \forall\mathrm{L}(\times 4)\ \&\ \Box\mathrm{L}$$

其中

$$\begin{aligned}\phi_5 &\coloneqq \mathtt{send}(b,d,[c]) \to \phi_6\\ \phi_6 &\coloneqq \Diamond_T(\phi_7)\\ \phi_7 &\coloneqq c \in [c] \to \mathtt{recv}(c,d,b)\end{aligned}$$

最后，我们特别使用导出规则 $\Diamond$ R 得出结论，其中 $\Gamma_3 \vdash_v_t'' \sqsubseteq 2T \bullet v_t^{v_t}$很容易证明，但证明过程繁琐，需用到包括 $\sqsubseteq$传递 在内的规则，
•结合律，以及 公理$\Box$:

$$\dfrac{\dfrac{}{\Gamma_3 \vdash v_t \sqsubseteq v_t''^{v_t}}\ \sqsubseteq trans\ \&\ ax_{\sqsubseteq} \quad \dfrac{}{\Gamma_3 \vdash v_t'' \sqsubseteq 2T \bullet v_t^{v_t}}}{\Gamma_3 \vdash \Diamond_{2T}\mathtt{recv}(c,d,b)^{v_t}}\ \Diamond\_\mathrm{R}$$