Sepa高阶语言中的分离效应与共享效应

PEDRO H. AZEVEDO DE AMORIM,美国康奈尔大学
JUSTIN HSU,美国康奈尔大学

arXiv:2303.01616v1 [cs.PL] 2 Mar 2023

效应程序的交互方式超出了简单的输入输出，使组合推理颇具挑战。已有研究表明，当这类程序彼此“分离”时，也就是说，程序之间互不干扰，推理起来会容易一些。尽管对分离资源的推理已得到充分研究，但关于分离效应的研究仍然很少，尤其是在函数式高阶编程语言方面。

我们提出了两种高阶语言，用于推理效应程序中的共享与分离。第一种语言 λINI 采用线性类型系统和概率语义，其中两种积类型分别表示独立的序对和可能相关的序对。第二种语言 λ^2INI 是一种双层分层语言，其灵感来自 Benton 的线性-非线性（LNL）演算。我们以概率模型来阐释这种语言，同时也提供一般的范畴语义，并展示了概率编程之外的一系列具体模型。我们为所有语言证明了可靠性定理；对于 λ^2INI 的范畴模型，我们的一般可靠性定理采用了范畴粘合构造。

附加关键词和短语：概率编程、指称语义、效应、高阶语言

ACM 参考文献格式：
Pedro H. Azevedo de Amorim 和 Justin Hsu。2018 年。高阶语言中的分离效应与共享效应。*ACM* 程序设计语言会议录 1，CONF，文章 1（2018年1月），31 页。

1 引言

编程语言理论中的一个核心挑战，是提出针对带效应程序的可靠且富有表现力的推理原则。与纯程序不同，在纯程序中，不同程序只能通过明确定义的接口（例如函数调用的输入或输出）相互影响；而带效应程序之间的交互可能十分微妙，难以推理。为了简化形式化分析，了解不同的带效应计算何时是分离的非常有用，即它们不会相互干扰。例如，在存在内存分配或概率等效应时，了解指针何时不指向同一位置，或随机量何时必须相互独立，都很有用。

先前工作：关于资源分离的推理。尽管分离的效应在文献中受到的关注相对较少，但关于资源分离推理的研究由来已久资源 [O’Hearn 等人 2001；Pym 等人 2004]。资源的概念在计算机科学中无处不在，通常体现在效应程序与外部世界交互时。例如，在进行内存分配编程时，堆是一种资源；在进行概率抽样编程时，随机性可以被视为一种资源。

在某些情况下，确保计算能够分别访问资源很有用。使用指针编程时，指向同一地址的不同指针会别名，使得堆更新难以推理；要求程序不产生别名，可以让形式化验证更具模块化和组合性。在概率效应的例子中，资源分离对应于概率独立性，而一般的联合分布则可以共享资源。与其他分离概念一样，独立性能够简化对程序的推理。对于

作者地址：Pedro H. Azevedo de Amorim，康奈尔大学，美国纽约州伊萨卡，pamorim@cs.cornell.edu；Justin Hsu，康奈尔大学，美国纽约州伊萨卡，justin@cs.cornell.edu。

2018年。2475-1421/2018/1-文章
1 $15.00https://doi.org/

例如，如果程序的两个部分产生独立分布，那么它们的联合分布只取决于各自的概率——这两个部分之间不会出现意料之外的概率交互。独立性也可以是一项值得验证的属性；例如，在密码协议中，基本的安全属性可以用独立性来表述 [Barthe 等人 2019]。先前工作已经为一阶命令式语言的上下文开发了能够处理独立性的程序逻辑 [Barthe 等人 2019]。遗憾的是，目前尚不清楚如何在高阶语言中刻画独立性。

我们的工作。我们的目标是开发一种高阶语言，能够在各种上下文中推理共享效应和分离效应。该领域最接近的工作是分束演算 [O'Hearn 2003],分束蕴涵逻辑的 Curry–Howard 对应 [O'Hearn 和 Pym 1999]。尽管O'Hearn [2003] 为该语言给出了预层模型，并构建了一个用于推理内存操作程序的具体模型，但其他具体模型却难以找到。事实上，目前尚无已知的分束演算模型能够容纳概率或状态以外的其他常见单子效应。

在本文中，我们将始终以概率效应作为指导示例。我们首先通过对概率样本进行资源解释来确立独立性：如果两次计算使用互不相交的资源（即概率样本），那么它们产生的随机量彼此独立。我们的视角导出了两种能够推理概率独立性的线性高阶语言。这两种语言都具有一种积类型构造子 $\otimes$，它能够强制独立性，也就是说，类型为 $\mathbb{N} \otimes \mathbb{N}$ 的闭程序应由独立分布表示。

我们的第一种语言 λ_{INI} 是一种线性 λ演算，具有两种积类型：$\otimes$ 类型构造子确保序对的分量不共享任何资源，而 $\times$ 类型构造子则允许分量共享资源。直观而言，$\otimes$ 表示独立值的序对，而$\times$ 表示一般的、可能相互依赖的值的序对。我们为 λ_{INI}赋予指称语义，并证明其可靠性定理：积 $\otimes$ 确保概率独立性。

尽管概念上简洁，λ_{INI} 的表达能力有限。例如，为其扩展和类型会破坏可靠性。为了缓解这些问题，我们定义了一种更丰富的双层语言 λ^2_{INI}，其中 λ_{INI} 的两种积类型分别限定在不同层中。直观而言，一层允许共享随机性的计算，另一层则禁止计算共享随机性。为了使各层能够交互，独立语言包含一种模态，可以可靠地导入以共享语言编写的程序。这种设计受到 Azevedo de Amorim [2023], 近期工作的启发；他们提出了一种双层语言，用于结合概率编程语言的采样语义和线性算子语义。我们证明 λ^2_{INI}支持两种不同的和类型：“共享”层中的“共享”和类型，以及“独立”层中的“分离”和类型。我们为 λ^2_{INI}赋予指称语义，证明其可靠性，并给出 λ_{INI} 的两个片段到 λ^2_{INI}的翻译。

范畴语义与具体模型。 为了展示 λ^2_{INI} 的一般性及其与其他效应类别的联系，我们为 λ^2_{INI} 提出一种范畴语义，并证明我们的类型系统满足一般健全性定理。随后，我们介绍受多种现有带效应的编程语言启发而构建的本语言的具体模型。

- **线性逻辑**。线性逻辑的模型已被用于为概率语言提供语义 [Azevedo de Amorim 和 Kozen 2022年；Danos 和 Ehrhard 2011；Ehrhard 等人。 2017]。我们证明，将这些模型与马尔可夫核范畴配对，可得到λ^2INI 的模型。我们的可靠性定理保证了概率独立性；据我们所知，我们的方法首次确保了这些模型中的独立性。

• **分布式编程**。接下来，我们为分布式编程构建 λ^2_{INI} 的关系模型。在该模型中，程序描述多个代理的实现与通信模式。我们的可靠性定理表明，类型为$\tau_1 \otimes \tau_2$ 的全局程序可以编译为两个独立执行的局部程序。这一性质让人联想到编排式语言中的投影性质 [Montesi 2014]。

• **名称生成**。 具有名称生成功能的编程语言包含一种可生成新鲜标识符的原语。在某些上下文中，控制名称生成的时机和次数十分重要；例如，在密码学应用中重复使用 一次性随机数 值（“仅用一次的数字”）可能会使协议容易受到重放攻击。我们基于名称生成定义了 λ^2 INI 的一个模型。我们的可靠性定理表明，联结词⊗ 能够确保各组成部分使用的名称彼此不相交。

• **交换效应**。 我们指出，名称生成模型和有限分布模型都是交换效应的单子语义实例，并据此对它们进行推广。在温和的假设下，每个交换单子都能导出一个 λ^2INI 模型。

• **分束逻辑与分离逻辑**。 长期以来，研究者一直运用分束逻辑 来推理资源分离 [O'Hearn 和 Pym 1999；O'Hearn 等人 2001]。我们证明，仿射分束逻辑的所有模型也是 λ^2INI 的模型，但反之不成立。为作说明，我们重新考察 O'Hearn 的 SCI+，这是一种用于内存分配编程的分束类型系统 [O'Hearn2003]。我们基于 SCI+定义了 λ^2INI 的一个模型，并给出从 λ^2INI 到 SCI+的可靠翻译。

模型的多样性表明， λ^2_{INI} 是用于推理效应式高阶程序中的分离与共享的合适框架。

概述。 回顾数学预备知识（§2）后，我们介绍主要贡献：

• 首先，我们定义一种线性、高阶概率 λ-演算，称为 λ_{INI}，其类型能够刻画概率独立性和依赖性。我们为该语言给出指称语义，并证明 ⊗ 能够刻画概率独立性（§3）。

• 接下来，我们定义一种双层、高阶概率 λ-演算，称为 λ^2_{INI}。该语言结合了独立片段与共享片段，并采用两种不同的和类型：独立和与共享和。我们给出概率语义，并证明 ⊗ 能够刻画概率独立性；此外，我们还将 λ_{INI} 的两个片段嵌入 λ^2_{INI} （§4）中。

• 进一步推广后，我们为 λ^2_{INI}提出一种范畴语义。我们的语义是 Benton 提出的线性逻辑线性／非线性（LNL）模型 [Benton 1994]以及Azevedo de Amorim [2023]提出的演算（§5.1）的弱化版本。

• 我们给出一系列 λ^2_{INI}的模型，如上所述。类型系统的可靠性保证了这些模型中自然的独立性概念（§5.2）。

• 最后，我们证明一个一般健全性定理：每个类型为 $\tau_1 \otimes \tau_2$ 的程序都可以分解为两个程序 t_1 和 t_2，它们的类型分别为 τ_1 和 τ_2。我们的证明依赖于一个范畴粘合论证（§6）。

我们在（§7）综述相关工作，并在（§8）作总结。

2 背景

2.1 单子及其代数

我们假定读者了解范畴论的基本概念，包括函子、积、余积、笛卡尔闭范畴以及对称幺半闭范畴（SMCC）。有兴趣的读者可参阅 Leinster [2014]； Mac Lane [2013]，以了解该主题的优秀入门介绍。

单子。 遵循 Moggi [1991], 的开创性工作，带效应计算可以通过单子赋予语义。一个 单子在范畴 $\mathbf{C}$ 上是一个三元组 (T, μ, η) 满足 $T{:}\mathbf{C} \to \mathbf{C}$ 是一个函子，

$\mu_A\colon T^2A \to TA$和 $\eta_A\colon A \to TA$是满足以下条件的自然变换：$\mu_A \circ \mu_{TA} = \mu_A \circ T\mu_A$,$id_A = \mu_A \circ T\eta_A$ 和 $id_A = \mu_A \circ \eta_{TA}$。

另一种有用且等价的单子定义需要一个自然变换 η_A 以及一个提升操作（–）*：$\mathbf{C}(A, TB) \to \mathbf{C}(TA, TB)$，使得来自 C的对象和态射 $A \to TB$构成一个范畴，通常称为 *Kleisli* 范畴 $\mathbf{C}_T$。该范畴与C具有相同的对象，并且具有 $Hom_{\mathbf{C}_T}(A, B) = Hom_{\mathbf{C}}(A, TB)$。Kleisli 范畴常用于为带效应的编程语言赋予语义。

单子代数。给定一个单子 T，一个 T-代数 是一个序对 $(A, f\colon TA \to A)$，使得 $id_A = f \circ \eta_A$且 $f \circ \mu_A = f \circ Tf$。一个 T-代数态射 $h\colon (A, f) \to (B, g)$ 是一个 C态射 $h\colon A \to B$，使得 $g \circ Th = h \circ f$。T-代数和态射构成一个范畴 $\mathbf{C}^T$，即 *Eilenberg-Moore* 范畴。

2.2 概率论

我们将通过概率程序和效应来阐释我们的高阶语言。

定义 2.1。集合 X上的分布是一个函数 $\mu\colon X \to [0, 1]$，满足 $\sum_{x \in X} \mu(x) = 1$。

联合分布是集合 $X \times Y$上的分布。给定一个定义在 $X \times Y$上的联合分布 μ，其在 X上的边缘分布定义为 $\mu_X(x) = \sum_{y \in Y} \mu(x, y)$，而第二个边缘分布 μ_Y的定义类似。

定义 2.2。分布 μ在 $X \times Y$上是概率意义下的独立的，如果它是其边缘分布 μ_X和 μ_Y的乘积，即 $\mu(x, y) = \mu_X(x) \cdot \mu_Y(y)$，$x \in X$且 $y \in Y$。

可以为 Set 定义一个概率单子。给定一个集合 X，令 DX 为函数 $\mu\colon X \to [0, 1]$组成的集合，这些函数仅在有限个值上非零，并满足 $\sum_{x \in supp(\mu)} \mu(x) = 1$ [Fritz2020]。单子的单位元定义为 $\delta(a, b) = 1$当且仅当 $a = b$，否则为 0；而绑定定义为 $\mathsf{bind}(f)(\mu) = \sum_{x \in X} f(x)\mu(x)$。

3 用于独立性的线性语言

为了说明分离效应和共享效应的语言，我们将聚焦于一种效应：概率采样。我们将逐步构建两种高阶语言，使类型能够确保概率独立性——这是概率效应自然的分离概念。

3.1 通过线性性实现独立

在许多概率程序中，独立的量最初是通过采样指令生成的。因此，推理一对随机表达式是否独立的一种简单方法，是分析每个分量使用了哪些随机性来源：如果两个表达式使用不同的随机性来源，那么它们彼此独立；否则，它们可能彼此依赖。

例如，考虑一种简单类型的一阶调用按值语言，其中有一个原语 $\vdash \mathsf{coin} : \mathbb{B}$，用于抛掷一枚公平硬币。程序

$$\mathsf{let}\ x = \mathsf{coin}\ \mathsf{in}\ \mathsf{let}\ y = \mathsf{coin}\ \mathsf{in}\ (x, y)$$

抛掷两枚公平硬币，并将结果组成序对。由于 x 和 y 是不同的随机性来源，该程序会产生一个概率独立分布。另一方面，程序

$$\mathsf{let}\ x = \mathsf{coin}\ \mathsf{in}\ (x, x)$$

不会产生独立分布：两个分量始终相等，因此完全相关。这些原则非常适合子结构类型系统，因为子结构类型系统控制变量何时可以共享。为了探究这一想法，我们开发了一种语言 λ_{INI}，它具有能够推理概率独立性的仿射类型系统。

Variables	x, y, z		
Types	τ	::=	$\mathbb{B} \mid \tau \times \tau \mid \tau \otimes \tau \mid \tau \multimap \tau$
Expressions	t, u	::=	$x \mid b \in \mathbb{B} \mid \mathsf{coin} \mid (t, u) \mid \pi_i\, t$
		$\mid$	$t \otimes u \mid \mathsf{let}\, x \otimes y = t\, \mathsf{in}\, u \mid \lambda x.\, t \mid t\ u$
Contexts	Γ	::=	$x_1 : \tau_1, \ldots, x_n : \tau_n$

图1。 类型和项： λINI

3.2 引入语言 λ_{INI}

语法。图1 展示了类型和项的语法。除了基本类型（$\mathbb{B}$）之外，还有两种积类型：我们将 $\times$ 视为共享或可能依赖的积，而 $\otimes$ 则是独立积。该语言是高阶的，并具有线性箭头类型。相应的项语法相当标准，包括变量、数值常量和原始分布（硬币）。两种积可以分别由两种序对构造，并分别通过投影和 let 绑定消去。最后，我们还有常见的 λ抽象和应用。示例将使用标准语法糖 $\mathsf{let}\ x = t \mathsf{in}\ u \triangleq (\lambda x.\, u)\ t$。

类型系统。 图 2 展示了 λ_{INI}的类型规则；这些规则来自线性逻辑，是标准规则。变量规则 Var 是仿射的：上下文中的变量可以不被使用，并且不能任意复制变量。对于共享积 $\times$，引入规则 $\times$ Intro 在各前提之间共享上下文：两个分量可以使用相同的变量。可以从这些序对中投影出任一分量（$\times$ Elim$_i$）。相比之下，对于独立积 $\otimes$，引入规则 $\otimes$Intro 要求两个前提使用不相交的上下文。因此，两个分量不能共享变量。张量序对通过一个同时消耗两个分量的 let-序对构造来消去（$\otimes$ Elim）。在子结构类型系统中，$\times$ 被称为加法的积，而 $\otimes$ 被称为乘法的积。抽象和应用规则是标准的。

加法箭头？ 请注意，应用规则是乘法的：函数不能与其参数共享变量。一个自然的问题是，箭头是否应当是加法的：函数与其参数之间能否共享变量？像分簇逻辑这样的子结构类型系统 [O'Hearn 和 Pym 1999] 同时包含乘法箭头和加法箭头。

虽然我们还没有定义该语言的语义，但我们先举一个例子，说明加法箭头会使 $\otimes$ 难以捕捉概率独立性。如果允许函数及其参数共享变量，我们就可以通过类型检查：

$$\cdot \vdash \mathsf{let}\ x = \mathsf{coin}\ \mathsf{in}\ (\lambda y.\, x \otimes y)\ x : \mathbb{B} \otimes \mathbb{B}$$

根据我们接下来将介绍的语义，该程序等价于 $\mathsf{let}\ x = \mathsf{coin}\ \mathsf{in}\ x \otimes x$，它会产生一对相关值。因此，我们的语言采用乘法箭头。

3.3 指称语义

我们可以利用范畴**Set** 和有限概率单子D为该语言赋予语义。图 3 自上而下定义了类型、上下文和类型推导的语义，后者会产生类型正确的项。对于类型，我们将两种积类型都解释为集合的积。箭头类型被解释为 Kleisli 箭头的集合，即映射 $[\![\tau_1]\!] \to D\,[\![\tau_2]\!]$。上下文被解释为集合的积。

我们将类型正确的项解释为 Kleisli 箭头。下面简要介绍项语义，它本质上与 Moggi [1991]提出的 Kleisli 语义相同。变量的解释是

$$\frac{}{\cdot \vdash b : \mathbb{B}}\ \text{常量} \qquad \frac{}{\cdot \vdash \text{硬币规则} : \mathbb{B}}\ \textsc{Coin} \qquad \frac{}{\Gamma,\ x :\ \tau \vdash x :\ \tau}\ \textsc{Var}$$

$$\frac{\Gamma \vdash t_1 : \tau \qquad \Gamma \vdash t_2 : \tau_2}{\Gamma \vdash (t_1, t_2) :\ \tau_1 \times \tau_2}\ \times \text{引入} \qquad \frac{\Gamma \vdash t :\ \tau_1 \times \tau_2}{\Gamma \vdash \pi_i\, t : \tau_i}\ \times \text{消去}_i$$

$$\frac{\Gamma_1 \vdash t_1 : \tau \qquad \Gamma_2 \vdash t_2 : \tau_2}{\Gamma_1,\ \Gamma_2 \vdash t_1 \otimes t_2 :\ \tau_1 \otimes \tau_2}\ \otimes \text{引入} \qquad \frac{\Gamma_1 \vdash t :\ \tau_1 \otimes \tau_2 \qquad \Gamma_2,\ x :\ \tau_1,\ y :\ \tau_2 \vdash u :\ \tau}{\Gamma_1,\ \Gamma_2 \vdash \text{Let绑定 } x \otimes y = t \text{其中 } u :\ \tau}\ \otimes \text{消去}$$

$$\frac{\Gamma,\ x :\ \tau_1 \vdash t :\ \tau_2}{\Gamma \vdash \lambda x.\, t :\ \tau_1 \multimap \tau_2}\ \text{抽象} \qquad \frac{\Gamma_1 \vdash t : \tau_1 \multimap \tau_2 \qquad \Gamma_2 \vdash u :\ \tau_1}{\Gamma_1,\ \Gamma_2 \vdash t\, u :\ \tau_2}\ \text{应用}$$

图 2. 类型规则： λINI

$$\begin{aligned}
[\![\mathbb{B}]\!] &= \mathbb{B} \\
[\![\tau \times \tau]\!] &= [\![\tau]\!] \times [\![\tau]\!] \\
[\![\tau \otimes \tau]\!] &= [\![\tau]\!] \times [\![\tau]\!] \\
[\![\tau_1 \multimap \tau_2]\!] &= [\![\tau_1]\!] \to D\,[\![\tau_2]\!]
\end{aligned}$$

$$[\![x_1 : \tau_1, \ldots, x_n : \tau_n]\!] = [\![\tau_1]\!] \times \cdots \times [\![\tau_n]\!]$$

$$[\![\Gamma \vdash t : \tau]\!] : [\![\Gamma]\!] \to D\,[\![\tau]\!]$$

$$\begin{aligned}
[\![x]\!]\,(\gamma, v_x) &= \mathsf{return}\ v_x \\
[\![b]\!]\,(*) &= \mathsf{return}\ b \\
[\![\mathsf{coin}]\!]\,(*) &= \frac{1}{2}(\delta_{\mathsf{tt}} + \delta_{\mathsf{ff}}) \\
[\![(t_1, t_2)]\!]\,(\gamma) &= x \leftarrow [\![t_1]\!]\,(\gamma); y \leftarrow [\![t_2]\!]\,(\gamma); \mathsf{return}\ (x, y) \\
[\![\pi_i\ t]\!]\,(\gamma) &= (x, y) \leftarrow [\![t]\!]\,(\gamma); \mathsf{return}\ x \\
[\![t_1 \otimes t_2]\!]\,(\gamma_1, \gamma_2) &= x \leftarrow [\![t_1]\!]\,(\gamma_1); y \leftarrow [\![t_2]\!]\,(\gamma_2); \mathsf{return}\ (x, y) \\
[\![\mathsf{let}\ x \otimes y = t\ \mathsf{in}\ u]\!]\,(\gamma_1, \gamma_2) &= (x, y) \leftarrow [\![t]\!]\,(\gamma_1); [\![u]\!]\,(\gamma_2, x, y) \\
[\![\lambda x.\, t]\!]\,(\gamma) &= \mathsf{return}\ (\lambda x.\, [\![t]\!]\,(\gamma)) \\
[\![t\ u]\!]\,(\gamma_1, \gamma_2) &= f \leftarrow [\![t]\!]\,(\gamma_1); x \leftarrow [\![u]\!]\,(\gamma_2); f(x)
\end{aligned}$$

图 3. 指称语义： λINI

使用单子的单位元，它将值 v 映射为点质量分布 δ_v。硬币规则被解释为 tt 和 ff 上两个点质量分布的公平凸组合。

其余构造涉及采样，在语义上由 Kleisli 态射的复合建模。我们使用单子箭头表示法来表示 Kleisli 复合，即 $x \leftarrow f; g \triangleq g^* \circ f$。两个对构造子的语义相同：分别从每个分量采样，然后

将结果配对。× 的投影用于计算联合分布的边缘分布，而针对 ⊗ 的 Let 绑定则从对 t 中采样，然后在主体中使用该样本 u。λ 抽象被解释为点质量分布，而应用则被解释为对函数进行采样、对参数进行采样，然后将前者应用于后者。

例 3.1（相关对）。乍看之下，似乎无法创建非独立的序对，因为两种序对的语义都会分别对各个分量进行采样。然而，考虑程序 let $x =$ 硬币规则 in $(x,\ x)$。展开定义后，其语义为

$$x \leftarrow \frac{1}{2}(\delta_0 + \delta_1); y \leftarrow \delta_x; z \leftarrow \delta_x; \delta_{(y,z)} = x \leftarrow \frac{1}{2}(\delta_0 + \delta_1); \delta_{(x,x)} = \frac{1}{2}(\delta_{(0,0)} + \delta_{(1,1)}).$$

所得的采样完全相关，而非相互独立。

例 3.2（独立对是相关对）。独立分布也可能是依赖分布。在 λ_{INI}中，这一点体现在以下程序中：

$$\cdot \vdash \lambda z.\, \text{let } x \otimes y = z \text{ in } (x, y) : \tau_1 \otimes \tau_2 \multimap \tau_1 \times \tau_2.$$

展开该程序的语义后，可以看到该程序并未修改输入。

3.4 可靠性

λ_{INI} 的类型系统保证 ⊗ 强制实现概率独立性。具体而言，如果 $\cdot \vdash t :\ \tau_1 \otimes \tau_2$ 类型正确，则 $[\![t]\!]\,(*)$ 是 $[\![\tau_1]\!] \times [\![\tau_2]\!]$ 上的独立概率分布。我们通过构造逻辑关系 $\mathrm{R}_\tau \subseteq D([\![\tau]\!])$ 来证明这个可靠性定理，其定义如下：

$$\begin{aligned}
\mathcal{R}_{\mathbb{B}} &= D(\mathbb{B}) \\
\mathcal{R}_{\tau_1 \otimes \tau_2} &= \{\mu_1 \otimes \mu_2 \in D([\![\tau_1]\!] \times [\![\tau_2]\!]) \mid \mu_i \in \mathcal{R}_{\tau_i}\} \\
\mathcal{R}_{\tau_1 \times \tau_2} &= \{\mu \in D([\![\tau_1]\!] \times [\![\tau_2]\!]) \mid \pi_i(\mu) \in \mathcal{R}_{\tau_i} \text{ for } i \in \{1,2\}\} \\
\mathcal{R}_{\tau_1 \multimap \tau_2} &= \{\mu \in D([\![\tau_1]\!] \to D([\![\tau_2]\!])) \mid \forall \mu' \in \mathcal{R}_{\tau_1}, x \leftarrow \mu'; f \leftarrow \mu; f(x) \in R_{\tau_2}\}.
\end{aligned}$$

定理 3.3。若 $x_1 :\ \tau_1, \ldots\ldots, x_n :\ \tau_n \vdash t :\ \tau$ 且 $\mu_i \in \mathcal{R}_{\tau_i}$ 则

$$(x_1 \leftarrow \mu_1; \cdots; x_n \leftarrow \mu_n; [\![t]\!]\,(x_1, \ldots, x_n)) \in \mathcal{R}_\tau.$$

证明。令上述分布为 ν。我们用 x_i 作为 $x_1, \ldots\ldots, x_n$ 的简写，用 $x_i \leftarrow \mu_i$ 作为 $\overline{x_1 \leftarrow \mu_1}; \cdots; x_n \leftarrow \mu_n$ 的简写。我们通过对 $\Gamma \vdash t :\ \tau$ 的推导进行归纳法，证明 $\nu \in \mathrm{R}_\tau$。

常量/硬币规则/Var。显然。例如，Var：$\nu = x_i \leftarrow \overline{\mu_i};$ 根据假设，return $x_i = \mu_i$ 属于 R_{τ_i}。

×**引入**。我们有 $\nu = x_i \leftarrow \overline{\mu_i};\ x \leftarrow [\![t_1]\!](x_i);$ 返回 $(x,\ y)$。不难证明，ν的第一个边缘分布是 $x_i \leftarrow \mu_i;\ x \leftarrow [\![t_1]\!](x_i);$ 返回 x，根据归纳假设，它是R_{τ_1}中的一个元素；类似地，ν的第二个边缘分布是 R_{τ_2}中的一个元素。

×**消去**。我们有 $\nu = x_i \leftarrow \overline{\mu_i;\ (x,\ y)} \leftarrow [\![t]\!](x_i);$ 返回 x。根据归纳假设，$[\![t]\!](x_i) \in \mathrm{R}_{\tau_1 \times \tau_2}$，并且根据假设，其边缘分布是 R_{τ_1} 和 R_{τ_2}中的元素。

⊗**引入**。Let绑定 μ_i是对应于 Γ_1的分布序列，并且令 η_i为对应于 Γ_2的分布序列。由于 D 是一个交换单子 [Borceux 1994], 我们可以应用结合律和交换律来证明：

$$\begin{aligned}
\nu &= x_i \leftarrow \overline{\mu_i}; y_i \leftarrow \overline{\eta_i}; x \leftarrow [\![t_1]\!]\,(\overline{x_i}); y \leftarrow [\![t_2]\!]\,(\overline{y_i}); \text{return } (x, y) \\
&= \overline{x_i \leftarrow \mu_i}; x \leftarrow [\![t_1]\!]\,(\overline{x_i}); \overline{y_i \leftarrow \eta_i}; y \leftarrow [\![t_2]\!]\,(\overline{y_i}); \text{return } (x, y) \\
&= (\overline{x_i \leftarrow \mu_i}; x \leftarrow [\![t_1]\!]\,(\overline{x_i}); \text{return } x) \otimes (\overline{y_i \leftarrow \eta_i}; y \leftarrow [\![t_2]\!]\,(\overline{y_i}); \text{return } y) = \nu_1 \otimes \nu_2.
\end{aligned}$$

此外，根据归纳假设，$\nu_i \in \mathrm{R}_{\tau_i}$，因此 $\nu = \nu_1 \otimes \nu_2 \in \mathrm{R}_{\tau_1 \otimes \tau_2}$，如所需。

⊗ **消去**。 令 $\overline{\mu_i}$ 为与 Γ_1 对应的分布序列，并令 η_i 为与 Γ_2 对应的分布序列。我们有：

$$\begin{aligned} \nu &= \overline{x_i \leftarrow \mu_i}; \overline{y_i \leftarrow \eta_i}; (x,y) \leftarrow [\![t]\!]\,(\overline{x_i}); \\ &= \overline{x_i \leftarrow \mu_i}; (x,y) \leftarrow [\![t]\!]\,(\overline{x_i}); \overline{y_i \leftarrow \eta_i}; [\![u]\!]\,(\overline{y_i}, x, y) \\ &= (x,y) \leftarrow \nu_1 \otimes \nu_2; \overline{y_i \leftarrow \eta_i}; [\![u]\!]\,(\overline{y_i}, x, y) \\ &= \overline{y_i \leftarrow \eta_i}; x \leftarrow \nu_1; y \leftarrow \nu_2; [\![u]\!]\,(\overline{y_i}, x, y) \end{aligned}$$

其中第三个等式由第一条前提中的归纳假设得出。根据第二条前提中的归纳假设，最终的分布属于 R_τ，正如所需。

抽象。 展开定义后，我们需要证明

$$x \leftarrow \mu; f \leftarrow (x_i \leftarrow \mu_i; \delta_{\lambda x.[\![t]\!](x_i)}); f(x) \in \mathcal{R}_{\tau_2},$$

对于某个 $\mu \in \mathrm{R}_{\tau_1}$。根据结合律和交换律，该分布等于 $x_i \leftarrow \mu_i;\ x \leftarrow \mu; f \leftarrow \delta_{\lambda x.[\![t]\!](x_i)};\ f(x)$。根据归纳假设以及 δ 是单子的单位元这一事实，我们可以得出此情形的结论。

应用。 此情形可直接由归纳假设得出。 □

关于 λ_{INI} 的可靠性可立即得出。

推论 3.4. 若 $\cdot \vdash t : \tau_1 \otimes \tau_2$，则 $[\![t]\!](*)$ 是 $[\![\tau_1]\!] \times [\![\tau_2]\!]$ 上的独立概率分布。

4 用于独立性的双层语言

仿射类型系统 λ_{INI} 能够区分相互独立和可能存在依赖性的随机量，但该语言的表达能力还不够理想。我们先讨论这些局限，然后引入一种分层的双层语言 λ^2_{INI} 来解决这些问题。最后，我们将展示如何把 λ_{INI} 的两个片段嵌入 λ^2_{INI}。

4.1 λ_{INI}的局限：和类型与 Let绑定

添加和类型。 尽管 λ_{INI}中有诸如 $\mathbb{B}$ 这样的基本类型，却没有条件表达式。为λ_{INI} 扩展和类型与分情况分析会立即引发问题。考虑以下程序：

$$\mathsf{if\ coin\ then\ tt} \otimes \mathsf{tt\ else\ ff} \otimes \mathsf{ff}$$

从操作上看，这个概率程序会抛一枚公平硬币，并将结果复制两份组成一个对：tt ⊗tt 或 ff ⊗ ff。由于 tt 和 ff 是常量，因此它们不共享任何变量，所以两个分支都可以赋予类型 $\mathbb{B} \otimes \mathbb{B}$，而标准的分情况分析规则会将整个程序赋予类型 $\mathbb{B} \otimes \mathbb{B}$。然而，这种扩展会破坏可靠性（定理3.3）：这个对并非概率独立，因为它的两个分量始终相等。

这个例子说明，我们不应允许分情况分析产生类型为$\tau_1 \otimes \tau_2$的程序。不过请注意，允许分情况分析产生类型为 $\tau_1 \times \tau_2$的程序是安全的，因为这种乘积并未断言独立性。因此，在 λ_{INI} 中引入和类型，同时保持可靠性，似乎需要对消去规则施加特设限制。

重用变量。 另一项限制是，函数应用是乘法的。使用 let 绑定时可以看出这一限制，因为 let 绑定是函数应用的语法糖。在 $\mathsf{let}\ x = t\ \mathsf{in}\ u$ 中，项 t 和 u 不能 共享任何变量。例如， λ_{INI} 不允许以下程序：

$$\begin{aligned} &\mathsf{let}\ x_1 = \mathsf{coin}\ \mathsf{in}\ \mathsf{let}\ x_2 = \mathsf{coin}\ \mathsf{in} \\ &\quad \mathsf{let}\ y = f(x_1, x_2)\ \mathsf{in}\ \mathsf{let}\ z = g(x_1, x_2)\ \mathsf{in}\ (y, z) \end{aligned}$$

不过，也有一些实用的采样算法（例如 Box-Muller 变换 [Box 和 Muller1958]），它们遵循这种形式。为了在 λ_{INI}中编写这个程序的类型正确版本，我们可以内联 y 和 z的定义：对构造器(–, –) 具有可加性，因此两个分量都可以使用 x_1 和 x_2。不过，要求进行这种修改并不妥当。

类似地，给定一个类型为 $\tau_1 \times \tau_2$ 的项，我们无法同时直接投影出两个分量。例如，下面的程序

$$\text{let } x = \pi_1\, z \text{ in let } y = \pi_2\, z \text{ in } f(x, y)$$

类型检查不通过，因为外层 let 绑定与其主体共享变量 z。如果 λ_{INI} 中的函数应用具有可加性，就能解决这些问题；然而，正如我们在第 3 节所见，允许函数与参数共享变量也会破坏 λ_{INI}的可靠性。

4.2 语言 λ^2_{INI}：语法、类型规则与语义

为了解决这些局限，我们引入一种分层语言。我们的思路源于对积、和类型和分布的一项简单观察，这一观察或许具有更广泛的意义。在λ_{INI}中，积类型对应于将分布与积组合的两种不同方式：共享积 $\tau_1 \times \tau_2$对应于积的分布， $M(\tau_1 \times \tau_2)$；而分离积 $\tau_1 \otimes \tau_2$ 对应于分布的积， $M\tau_1 \times M\tau_2$。

类似地，分布与和类型有两种组合方式：和类型的分布， $M(\tau_1+\tau_2)$，以及分布的和类型，$M\tau_1 + M\tau_2$。我们将第一种组合视为共享和，因为分布可以同时将概率质量分配给和类型的两个分量。相反，第二种组合是分离和，因为分布要么将全部概率质量分配给 τ_1，要么全部分配给 τ_2。

最后，共享与分离、和类型与积之间存在有趣的相互作用。例如，我们之前看到的那个有问题的和类型示例对硬币规则执行分情况分析——它是一个共享和，因为它有一定概率返回 true，也有一定概率返回 false——但却产生了一个分离积 $\mathbb{B} \otimes \mathbb{B}$。如果我们改为对分离和类型执行分情况分析，那么程序要么总是执行第一个分支，要么总是执行第二个分支，此时产生分离积就没有问题了。

这些观察促使我们设计一种双层语言，其中一层包含共享联结词，另一层包含分离联结词。我们将这种语言称为 λ^2_{INI}，其中 INI 代表独立/非独立。

语法。 图 λ^2_{INI}所总结的程序和类型语法4分为两个层次：非独立（NI）层和独立（I）层。我们将用颜色标记它们：NI语言为橙色，而I语言为紫色。

NI 层具有基础类型、积类型（×）和和类型（+）。该语言基本上是标准的：其中包含变量、常量、基本分布（硬币规则）以及一个从 τ_1到 τ_2的原语操作集合 $O(\tau_1, \tau_2)$，此外还包含积的常规配对和投影构造，以及和类型的单射和分情况分析构造。NI 层没有箭头，但支持 let 绑定。

该I层与 λ_{INI}非常相似：它有自己的积类型（⊗）和和类型（⊕），以及线性箭头类型（⊸）。类型 $\mathcal{M}(\tau)$ 将NI层中的类型带入I层。该语言同样相当标准，包含用于引入和消去积与和类型的构造，以及函数和应用。最后一个构造（采样 t 作为 x 在 M中）来自 [Azevedo de Amorim2023]：它允许两个层次进行交互。在此， t 和 x 是两个长度相同（可能为空）的列表。

直观而言，NI-语言允许共享，而 I-语言则不允许共享。每种语言都有各自的和类型，分别是共享和分离和，二者都能与各自的积类型良好地交互。可以将 $\mathcal{M}$ 模态视为两种语言之间的一道抽象屏障，它能

让分离程序能够操作共享程序，同时不允许检查其中的共享，除非是为了生成另一个装箱项。

变量 x, y, zNI 类型 $\tau ::= \quad \mathbb{B} \mid \tau \times \tau \mid \tau + \tau$I 类型 $\tau ::= \quad \tau \otimes \tau \mid \tau \oplus \tau \mid \tau \multimap \tau \mid$M (τ) NI 表达式 $M, N ::= \quad x \mid b \in \mathbb{B} \mid$ 硬币规则 $\mid f \in$O$(\tau_1, \tau_2) \mid (M, N) \mid \pi_i\, M \mid$ 在 $_i$t 情况 t 取值为 $(\mid$ 在 $_1 x \Rightarrow u_1 \mid$ 在$_2 x \Rightarrow u_2) \mid$Let绑定 $x = M$ 在 NI 表达式 $t, u ::= \quad x \mid t \otimes u \mid$Let绑定 $x \otimes y = t$ 在 $u \mid$运算符 $\mid$在$_i$ t情况 t取值为 $(\mid$在$_1 x \Rightarrow u_1 \mid$在$_2 x \Rightarrow u_2) \mid \lambda x.t \mid t\ u \mid$采样 t作为 x在 M NI 上下文 $\Gamma ::= \quad x_1 : \tau_1, \ldots, x_n : \tau_n$I 上下文 $\Gamma ::= \quad x_1 : \tau_1, \ldots, x_n : \tau_n$

图 4. 类型和项： λ^2INI

类型规则。 的类型规则 λ^2_{INI} 见图 5。我们针对两个层次分别使用两种类型判断；通过在推导符号上添加下标来表示所在层次。首先介绍第一组类型规则，即用于共享（NI）层的规则。这些类型规则完全遵循带有积和和类型的一阶语言的标准规则。请注意，所有规则都允许在不同前提之间共享上下文。特别是，let 绑定规则是加法的而非乘法的，就像 λ_{INI}中的规则一样：let 绑定允许在变量与其主体之间共享变量。

第二组类型规则为独立（I）层赋予类型。这些规则是乘法加法线性逻辑（MALL）的标准规则，与 λ_{INI}的类型规则几乎完全相同。与之前一样，这些规则仿射地处理变量，不允许在不同前提之间共享变量。和类型 $\tau_1 \oplus \tau_2$ 的规则是新增的。同样，消去（Case）规则不允许在守卫与主体之间共享变量。

最后一条规则，采样，给出了两种语言之间的交互规则。第一个前提来自共享（NI）语言，其中程序 M可以有自由变量 x_1、……、 x_n。其余前提来自独立（I）语言，其中线性程序 t_i 具有装箱类型$\mathcal{M}\tau_i$。该规则的结论将程序 t_i 与 M结合起来，生成一个具有装箱类型的I-程序。直观而言，该规则允许将共享语言中的程序导入线性语言。就操作而言，采样 t 作为 x 在 M 中构造一个分布 t，使用独立语言从中采样，并将采样值绑定到 x 共享程序 M 中，最后将结果装箱到线性语言中。

概率语义。为使表述具体，本节将采用一种受概率独立性启发的具体语义，其中程序是带有离散采样的概率程序。下一节将介绍 λ^2_{INI} 的一般范畴语义，并考察其他模型。

……的概率语义 λ^2_{INI} 定义于图 6中。对于 NI 层，我们采用与 λ_{INI}相同的语义，即将类型正确的程序解释为有限分布单子 D的 Kleisli 箭头。Kleisli 范畴 $\mathbf{Set}_D$ 以集合为对象，因此我们可以直接将每种类型的语义定义为一个集合。众所周知，$\mathbf{Set}_D$ 具有积和余积，可用于解释NI中的类型正确的程序。

对于 I-语言，我们使用有限分布单子 D 和普通映射 $\mathbf{Set}^D$所对应的代数范畴。具体而言，其对象是序对 (A, f)，其中 f是一个 M-代数，而态射

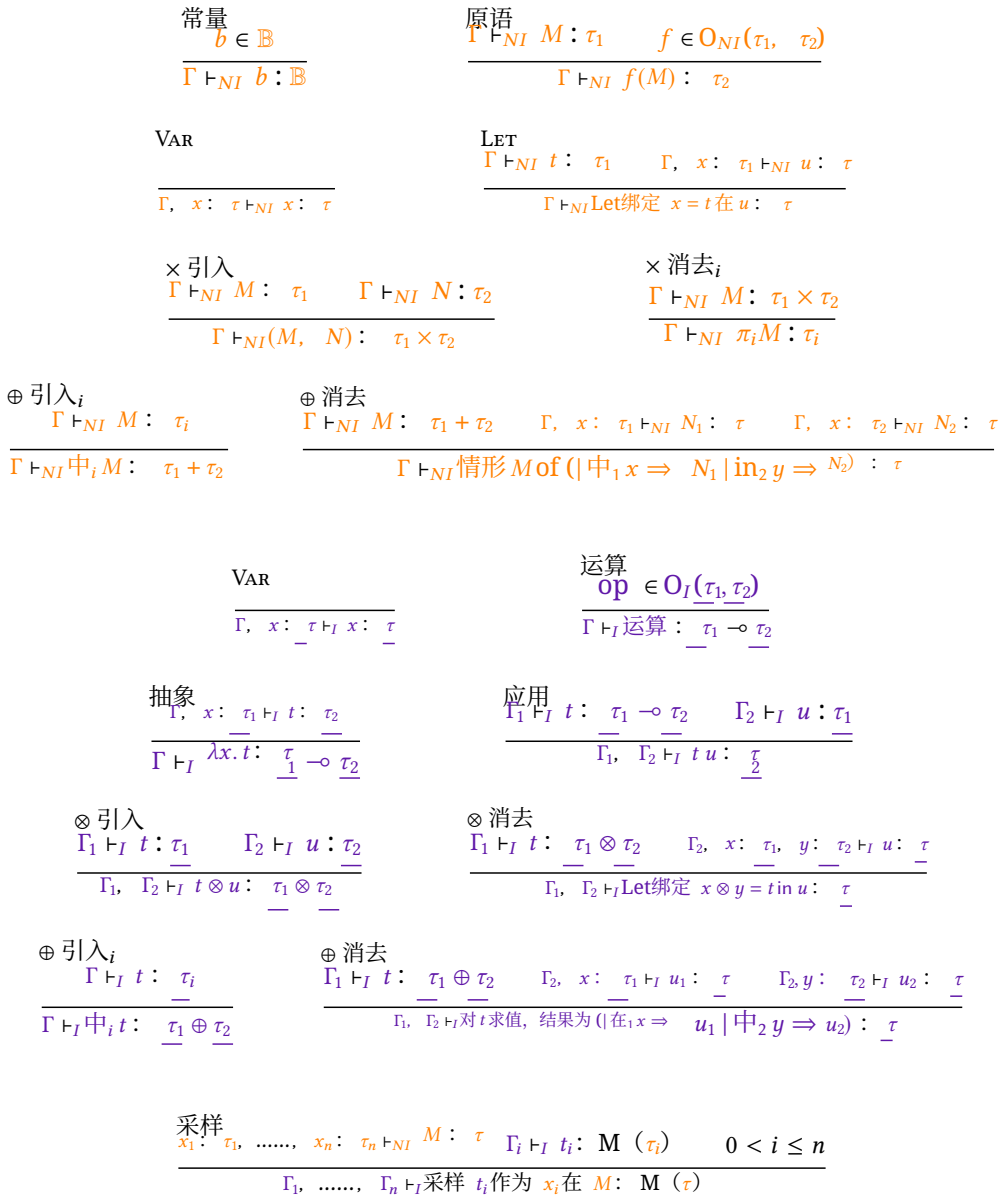


图 5. 类型规则：λ^2INI

$(A, f) \to (B, g)$是一个函数 $A \to B$。给定两个对象 (A, f)和 (B, g)，我们可以定义集合 $A \times B$上的积代数。此外，还可以为集合论不交并 $A + B$和指数对象 $A \Rightarrow B$赋予代数结构，从而得到一个支持分情况分析的高阶编程模型 [Simpson 1992]。只有在解释类型构造子 $\mathcal{M}$时，我们才需要显式定义代数结构；它被解释为自由 D-代数，其代数结构为该单子的乘法。

$$
\begin{array}{rr}
(\!|\mathbb{B}|\!) = \mathbb{B} & [\![\mathcal{M}\tau]\!] = (D\,[\![\tau]\!], \mu_{[\![\tau]\!]}) \\
(\!|\tau \times \tau|\!) = (\!|\tau|\!) \times (\!|\tau|\!) & [\![\underline{\tau} \otimes \underline{\tau}]\!] = [\![\underline{\tau}]\!] \times [\![\underline{\tau}]\!] \\
(\!|\tau + \tau|\!) = (\!|\tau|\!) + (\!|\tau|\!) & [\![\underline{\tau} \oplus \underline{\tau}]\!] = [\![\underline{\tau}]\!] + [\![\underline{\tau}]\!] \\
 & [\![\underline{\tau} \multimap \underline{\tau}]\!] = [\![\underline{\tau}]\!] \to [\![\underline{\tau}]\!] \\
(\!|x_1 : \tau_1, \ldots, x_n : \tau_n|\!) = (\!|\tau_1|\!) \times \cdots \times (\!|\tau_n|\!) & [\![x_1 : \underline{\tau}_1, \ldots, x_n : \underline{\tau}_n]\!] = [\![\underline{\tau}_1]\!] \times \cdots \times [\![\underline{\tau}_n]\!] \\
(\!|\Gamma \vdash M : \tau|\!) \in \mathbf{Set}_D((\!|\Gamma|\!), (\!|\tau|\!)) & [\![\Gamma \vdash t : \underline{\tau}]\!] \in \widetilde{\mathbf{Set}^D}([\![\Gamma]\!], [\![\underline{\tau}]\!])
\end{array}
$$

$$
\begin{array}{rl}
[\![x]\!]\,(\gamma, v_x) &= v_x \\
[\![t \otimes u]\!]\,(\gamma_1, \gamma_2) &= [\![t]\!]\,(\gamma_1) \times [\![u]\!]\,(\gamma_2) \\
[\![\mathsf{let}\ x \otimes y = t\ \mathsf{in}\ u]\!]\,(\gamma_1, \gamma_2) &= [\![u]\!]\,(\gamma_2, [\![t]\!]\,(\gamma_1)) \\
[\![\lambda x.\, t]\!]\,(\gamma)(x) &= [\![t]\!]\,(\gamma)(x) \\
[\![t\ u]\!]\,(\gamma_1, \gamma_2) &= [\![t]\!]\,(\gamma_1, [\![u]\!]\,(\gamma_2) \\
[\![\mathsf{in}_i t]\!]\,(\gamma) &= in_i([\![t]\!]\,(\gamma)) \\
[\![\mathsf{case}\ t\ \mathsf{of}\ (|\mathsf{in}_1 x \Rightarrow u_1 \mid \mathsf{in}_2 x \Rightarrow u_2)]\!]\,(\gamma_1, \gamma_2) &= \begin{cases} [\![u_1]\!]\,(\gamma_2, v), & [\![t]\!]\,(\gamma_1) = in_1(v) \\ [\![u_2]\!]\,(\gamma_2, v), & [\![t]\!]\,(\gamma_1) = in_2(v) \end{cases} \\
[\![\mathsf{sample}\ t_i\ \mathsf{as}\ x_i\ \mathsf{in}\ N]\!] &= \mu \circ D(\!|N|\!) \circ ([\![t_1]\!] \times \cdots \times [\![t_n]\!])
\end{array}
$$

图 6. 具体语义： λ^2INI

既然我们已经定义了 λ^2_{INI}的概率语义，就可以证明其可靠性定理：与 λ_{INI}中一样，类型构造子 $\otimes$ 保证了概率独立性。

定理 4.1. 若 $\cdot \vdash_I t : \mathcal{M}\tau_1 \otimes \mathcal{M}\tau_2$ 则 $[\![t]\!]$是一个独立分布。

证明。$\cdot \vdash_I t$: $\mathrm{M}\tau_1 \otimes \mathrm{M}\tau_2$的语义是一个集合论函数 $[\![t]\!]$: $1 \to D\,[\![\tau_1]\!] \times D\,[\![\tau_2]\!]$，它同构于一个独立分布。□

4.3 重新审视和类型与 Let绑定

让我们重新审视本节开头那个有问题的条件分支程序。 λ^2_{INI} 的类型系统使得我们无法通过对值进行模式匹配来生成独立的二元组：

$$\mathsf{dist} : \mathcal{M}(1+1) \not\vdash_I \mathsf{if}\ \mathsf{dist}\ \mathsf{then}\ (\mathsf{tt} \otimes \mathsf{tt})\ \mathsf{else}\ (\mathsf{ff} \otimes \mathsf{ff}) : \mathcal{M}\mathbb{B} \otimes \mathcal{M}\mathbb{B}$$

其中，if 语句只是对布尔值上的和类型进行消去。不过，如果使用共享积，我们就能写出这个程序的类型正确版本：

$$\mathsf{dist} : \mathrm{M}\,(1+1) \vdash_I \text{对 dist 进行采样，得到 } x\text{，并在其中执行 } (\mathsf{if}\ x\ \mathsf{then}\ (\mathsf{tt},\ \mathsf{tt})\ \mathsf{else}\ (\mathsf{ff},\ \mathsf{ff})) : \mathrm{M}\,(\mathbb{B} \times \mathbb{B})$$

λ^2_{INI} 的设计也消除了我们先前讨论过的 let 绑定的限制，因为共享层具有一种加法式let 绑定。具体来说，我们也可以表达先前遇到的那个有问题的 let 绑定程序：

$$\cdot \vdash_I \mathsf{sample}\ \mathsf{coin}, \mathsf{coin}\ \mathsf{as}\ x_1, x_2\ \mathsf{in}\ \mathsf{let}\ y = f(x_1, x_2)\ \mathsf{in}\ \mathsf{let}\ z = g(x_1, x_2)\ \mathsf{in}\ M : \mathcal{M}(\tau)$$

我们还可以在共享层中从序对中投影出两个分量：

$$\cdot \vdash_{NI} \text{let } x = \pi_1\, M_1 \text{ in let } y = \pi_2\, M_2 \text{ in } M : \tau$$

4.4 从 λ_{INI} 到 λ^2_{INI} 的嵌入

既然我们已经看到了 λ_{INI} 和 λ^2_{INI}，一个自然的问题是这两种语言之间有什么关系。我们首先展示如何将不含箭头类型的 λ_{INI} 片段嵌入 λ^2_{INI}。其思路是，λ_{INI} 的语义由一个 Kleisli 范畴给出，因此可以将其转换到NI层的 λ^2_{INI}中。类型按如下方式转换：

$$\mathcal{T}(\mathbb{B}) \triangleq \mathbb{B} \qquad \mathcal{T}(\tau_1 \times \tau_2) = \mathcal{T}(\tau_1 \otimes \tau_2) \triangleq \mathcal{T}(\tau_1) \times \mathcal{T}(\tau_2)$$

在项层面上，这种转换就是恒等函数。

定理 4.2。 若 $\Gamma \vdash M : \tau$ 属于 λ_{INI}，则 $\mathcal{T}(\Gamma) \vdash_{NI} \mathcal{T}(M) : \mathcal{T}(\tau)$ 属于 λ^2_{INI}。

此外，这种转换保留程序之间的等式，并且是完全抽象的。

定理 4。 **3.** 令 $\Gamma \vdash t_1 : \tau$，且 $\Gamma \vdash t_2 : \tau$ 属于 λ_{INI}，则 $[\![t_1]\!] = [\![t_2]\!]$ 当且仅当 $[\![\mathcal{T}(t_1)]\!] = [\![\mathcal{T}(t_2)]\!]$。

证明。证明基于这样一个事实：该翻译是一个忠实函子。 □

也可以将 $\otimes, \multimap$ 片段中的乘法部分 λ_{INI} 翻译到 I 层 λ^2_{INI}中，类型按如下方式翻译：

$$\mathbf{T}'(\mathbb{B}) \triangleq \mathcal{M}\mathbb{B} \qquad \mathcal{T}'(\tau_1 \otimes \tau_2) \triangleq \mathcal{T}'(\tau_1) \otimes \mathcal{T}'(\tau_2) \qquad \mathcal{T}'(\tau_1 \multimap \tau_2) \triangleq \mathcal{T}'(\tau_1) \multimap \mathcal{T}'(\tau_2)$$

同样，项的翻译是恒等函数。

定理 4.4。 若 $\Gamma \vdash t : \tau$ 属于 λ_{INI}，则 $\mathcal{T}'(\Gamma) \vdash_{I} \mathcal{T}'(t) : \mathcal{T}'(\tau)$ 属于 λ^2_{INI}。

证明。对类型推导 $\Gamma \vdash t : \tau$ 进行归纳即可得证。 □

此翻译具有函子性且忠实，因此相对于 λ_{INI} 和 λ^2_{INI}的指称语义而言，它是可靠且完全抽象的。

备注 4.5。 不可能将整个 λ_{INI} 翻译成 λ^2_{INI}。由于 λ^2_{INI} 的语言中只有一种具有箭头类型，并且无法从 I 转换到 NI，因此这种翻译需要将 λ_{INI} 程序映射为 I 程序，而这些程序只能编写概率独立程序，所以无法翻译 $\times$ 类型构造子。若在NI层中 λ^2_{INI}加入加性函数类型，就可以扩展第一种翻译，使其涵盖整个语言；然而，我们将在下一节中考察的许多具体模型并不支持NI层中的加性函数类型。

5 范畴语义与具体模型

本节将通过对上一节介绍的概率语义进行抽象，给出 λ^2_{INI}的一般范畴语义。然后，我们将基于效应语言已有的语义，为λ^2_{INI}介绍多种具体模型。我们的可靠性定理确保这些模型中均具有自然的分离概念。

5.1 λ^2_{INI}的范畴语义

假设我们有两种效应语言 L_1 和 L_2。第一种语言具有积类型 $\times$，允许共享资源；而第二种语言具有不交积类型 $\otimes$。此外，我们假设 L_2 具有一个连接两种语言的一元类型构造子 M。作出这一决定背后的直觉是，类型 $\mathrm{M}\tau$ 的元素是一种可能共享资源的计算。

从语言设计的角度来看，构造器 M 用于在独立环境中封装可能具有依赖性的计算。

首先要理解的问题是，联结词 $\times$ 和 $\otimes$ 应如何进行范畴论解释。对于 $\times$，我们需要一种余幺半结构来复制和擦除计算。这类结构由 *CD* 范畴刻画：它们是幺半范畴，其中每个对象 A都配备了交换余幺半结构 $A \to A \otimes A$ 和 $A \to I$，使某些图表交换 [Cho 和 Jacobs 2019]。对于 $\otimes$，我们希望限制复制——语言中负责分离的层采用仿射类型系统——因此 $\otimes$ 应当是带有丢弃映射的幺半积。

最后，要为类型构造子 M 建立模型，典型的范畴论思路是将其视为某种从 L_1 到 L_2的函子。我们来看看类型系统提供的一些直觉。类型 $\mathrm{M}\,(\tau_1 \times \tau_2)$ 表示可能共享资源并同时输出 τ_1 和 τ_2 的计算。与此同时，类型 $\mathrm{M}\tau_1 \otimes \mathrm{M}\tau_2$ 表示使用彼此分离的资源、输出 τ_1 和 τ_2 的计算。这种理解表明，不应存在从 $\mathrm{M}\,(\tau_1 \times \tau_2)$ 到 $\mathrm{M}\tau_1 \otimes \mathrm{M}\tau_2$的映射，因为资源一旦共享，就无法再将其分离；但应存在从 $\mathrm{M}\tau_1 \otimes \mathrm{M}\tau_2$ 到 $\mathrm{M}(\tau_1 \times \tau_2)$的映射，因为分离是共享的一种特定情形。

从范畴论的角度看，这些映射的存在性由应用函子（亦称松幺半函子）体现。它们是幺半范畴之间的函子 $F\colon (\mathbf{C}, \otimes_C, I_C) \to (\mathbf{D}, \otimes_D, I_D)$，并配备态射 $\mu_{A,B}\colon F(A) \otimes_D F(B) \to F(A \otimes_C B)$ 和 $\epsilon\colon I_D \to FI_C$，使某些图表交换 [Borceux 1994]。

因此，我们得到了 λ^2_{INI}的范畴模型。

定义5.1。 A λ^2_{INI} 模型是一个三元组 $(\mathbf{C}, \mathbf{M}, \mathrm{M})$，其中 $\mathbf{C}$ 是一个带有余积及态射 $del_A\colon A \to I_C$（关于 A是自然的）的对称幺半闭范畴；$\mathbf{M}$ 是一个带有余积的分配CD范畴，即 $A \otimes_M (B +_M C) \cong (A \otimes_M B) +_M (A \otimes_M C)$；并且 $\mathrm{M} : \mathbf{M} \to \mathbf{C}$ 是松幺半的。

虽然我们需要在 $\mathbf{M}$ 中假设分配律，但 $\mathbf{C}$ 中的分配律会自动成立。

引理5.2。 在每个具有余积的对称幺半闭范畴中，以下同构成立：$A \otimes (B + C) \cong (A \otimes B) + (A \otimes C)$。

证明。根据假设，函子 $A \otimes (-)$ 是一个左伴随，因此保持余积，由此我们可以得出同构 $A \otimes (B + C) \cong (A \otimes B) + (A \otimes C)$。□

指称语义见图 7，大部分等式理论见图 8。由于 $\mathcal{M}$ 的松幺半群方程较长且信息量不大，因此将其留在附录 A 中。

可靠性。 在范畴模型中，λ^2_{INI} 的可靠性定理可以表述如下：

定理 5.3（可靠性）。 令 $\cdot \vdash_I t : \tau_1 \otimes \tau_2$，则 $[\![t]\!] = f \otimes g$，其中 f和 g 分别是态射$I \to [\![\tau_1]\!]$ 和 $I \to [\![\tau_2]\!]$。

从证明论视角来看，可靠性定理指出，对于每个类型为 $\cdot \vdash \tau_1 \otimes \tau_2$ 的证明，我们可以假设最后一条规则是 $\otimes$ 的引入规则。从语义视角来看，可靠性定理意味着，对于每个闭项 $\cdot \vdash t : \tau_1 \otimes \tau_2$，其语义$[\![t]\!]$ 可以分解为两个态射 f_1 和 f_2，使得 $[\![t]\!] = f_1 \otimes f_2$。

证明可靠性需要额外的范畴论工具，因此我们将证明推迟到第 6节。在这里，我们将展示一系列关于 λ^2_{INI}的具体模型。

5.2 具体模型

首先，我们介绍一些基本的概率模型 λ^2_{INI}。虽然先前工作也研究过类似的模型 [Azevedo de Amorim 2023], 我们将这些模型调整为 λ^2_{INI} 并说明我们的可靠性定理如何确保独立性。

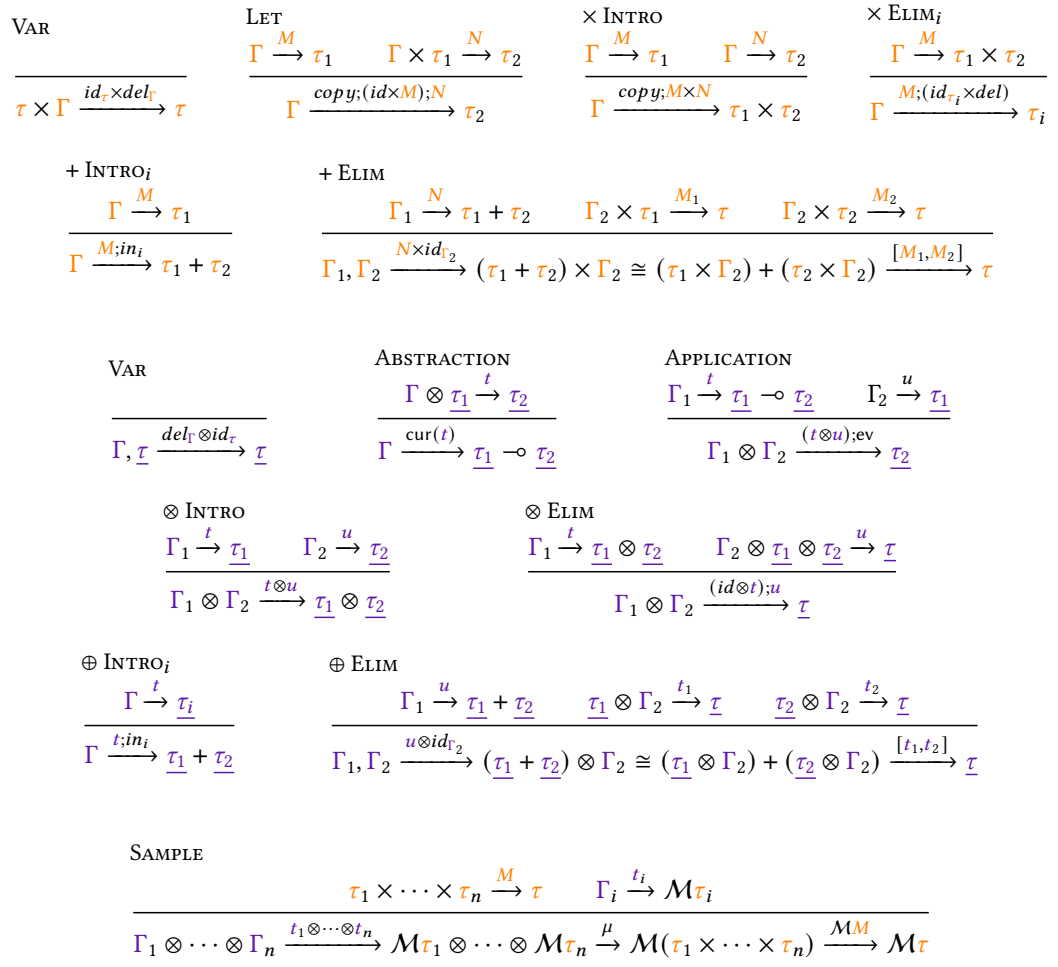


图 7. 范畴语义： λ^2INI

5.2.1 离散概率。我们的第一个具体模型是离散概率的一种不同语义。对于共享范畴，我们采用范畴**CountStoch** 其对象为可数集，态射为转移矩阵，即函数 $f\colon\ A\times B\to[0,1]$ 满足对于每个 $a\in A$， $f(a,\ -)$都是一个（离散）概率分布 [Fritz 2020]。

对于独立范畴，我们采用线性逻辑的概率相干空间模型，这是离散概率语言中一种经过深入研究的语义 [Danos 和 Ehrhard 2011]。该模型最初用于探索概率论与线性逻辑之间的联系，近来则被用于解释递归概率程序和递归类型 [Tasson 和 Ehrhard 2019]；它对于概率 PCF 也具有完全抽象性 [Ehrhard 等人 2018]。

定义 5.4 (Danos 和 Ehrhard [2011])。 一个 概率相干空间 （*PCS*）是一个有序对 $(|X|,\mathcal{P}(X))$，其中 $|X|$是一个可数集，且 $\mathcal{P}(X)\subseteq|X|\to\mathbb{R}^+$满足：

- $\forall a\in|X|\ \exists\varepsilon_a>0\ \varepsilon_a\cdot\ \delta_a\in\mathrm{P}(X)$，其中 $\delta_a(a\prime)\ =1$当且仅当 $a=a\prime$，否则为 0；
- $\forall a\in|X|\ \exists\lambda_a\ \forall x\in\mathrm{P}\ (X)\ \ x_a\le\lambda_a$；

$$
\begin{array}{rcl}
\mathsf{case}\ (\mathsf{in}_1 M)\ \mathsf{of}\ (|\mathsf{in}_1 x \Rightarrow N_1 \mid \mathsf{in}_2 x \Rightarrow N_2) & \equiv & N_1\{M/x\} \\
\mathsf{case}\ (\mathsf{in}_2 M)\ \mathsf{of}\ (|\mathsf{in}_1 x \Rightarrow N_1 \mid \mathsf{in}_2 x \Rightarrow N_2) & \equiv & N_2\{M/x\} \\
\\
\mathsf{let}\ x = t\ \mathsf{in}\ x & \equiv & t \\
\mathsf{let}\ x = x\ \mathsf{in}\ t & \equiv & t \\
\mathsf{let}\ y = (\mathsf{let}\ x = M_1\ \mathsf{in}\ M_2)\ \mathsf{in}\ M_3 & \equiv & \mathsf{let}\ x = M_1\ \mathsf{in}\ (\mathsf{let}\ y = M_2\ \mathsf{in}\ M_3) \\
\\
(\lambda x.\, t)\ u & \equiv & t\{u/x\} \\
\mathsf{let}\ x_1 \otimes x_2 = t_1 \otimes t_2\ \mathsf{in}\ u & \equiv & u\{t_1/x_1\}\{t_2/x_2\} \\
\\
\mathsf{case}\ (\mathsf{in}_1 t)\ \mathsf{of}\ (|\mathsf{in}_1 x \Rightarrow u_1 \mid \mathsf{in}_2 x \Rightarrow u_2) & \equiv & u_1\{t/x\} \\
\mathsf{case}\ (\mathsf{in}_2 t)\ \mathsf{of}\ (|\mathsf{in}_1 x \Rightarrow u_1 \mid \mathsf{in}_2 x \Rightarrow u_2) & \equiv & u_2\{t/x\} \\
\\
\mathsf{sample}\ t\ \mathsf{as}\ x\ \mathsf{in}\ x & \equiv & t \\
\mathsf{sample}\ (\mathsf{sample}\ t\ \mathsf{as}\ x\ \mathsf{in}\ M)\ \mathsf{as}\ y\ \mathsf{in}\ N & \equiv & \mathsf{sample}\ t\ \mathsf{as}\ x\ \mathsf{in}\ (\mathsf{let}\ y = M\ \mathsf{in}\ N)
\end{array}
$$

图 8。 等式理论： $\lambda^{2}\mathrm{INI}$

- $\mathcal{P}(X)^{\perp\perp} = \mathcal{P}(X)$, where $\mathcal{P}(X)^{\perp} = \{x \in |X| \to \mathbb{R}^{+} \mid \forall v \in \mathcal{P}(X)\ \sum_{a \in |X|} x_a v_a \leq 1\}$.

我们可以定义一个范畴 **PCoh** ，其对象是概率相干空间，态射$X \multimap Y$是矩阵 f: $|X| \times |Y| \to \mathbb{R}^{+}$使得对于每个 $v \in \mathrm{P}(X)$， $fv \in \mathrm{P}(Y)$，其中 $(fv)_b = \sum_{a \in |X|} f_{(a,b)} v_a$。众所周知，这个范畴是一个对称幺半闭范畴；我们将使用幺半积的显式定义。

定义 5.5。 令 $(|X|, \mathrm{P}(X))$ 和 $(|Y|, \mathrm{P}(Y))$ 为概率相干空间，我们定义 $X \otimes Y = (|X| \times |Y|, \{x \otimes y \mid x \in \mathrm{P}(X)\ y \in \mathrm{P}(Y)\}^{\perp\perp})$，其中 $(x \otimes y)(a, b) = x(a)y(b)$。

现在我们可以定义一个函子 M: **CountStoch** $\to$ **PCoh**。

引理 5.6 （例如，参见 Azevedo de Amorim [2023]）。 令 X为一个可数集，序对 $(X, \{\mu : X \to \mathbb{R}^{+} \mid \sum_{x \in X} \mu(x) \leq 1\})$ 是一个概率相干空间。任何 **CountStoch** 态射 $X \to Y$也是一个 **PCoh** 态射。

定理 5.7。 存在一个松幺半函子 $\mathcal{M}$: **CountStoch** $\to$ **PCoh**。

证明。该函子按上述方式定义。由于该函子在箭头上是恒等映射，因此具有函子性。松弛幺半结构由 $\epsilon = id_1$ 和 $\mu_{X,Y} = id_{X \times Y}$给出。□

总而言之，我们得到了一个基于概率相干空间的$\lambda^{2}_{\mathrm{INI}}$ 模型。

定理 5.8。 三元组 (**PCoh**,**CountStoch**,$\mathcal{M}$) 是一个 λ^{2}_{INI} 模型。

证明。 **CountStoch** 众所周知是一个具有余积的 CD 范畴 [Fritz 2020], 和 **PCoh** 是一个具有余积的对称幺半闭范畴，因为它是线性逻辑的一个模型 [Danos 和 Ehrhard 2011]。态射 del_X 由常量 0 函数给出，其中幺半单位是区间 $[0, 1]$。最后，$\mathcal{M}$ 的松幺半性由前述定理给出。 □

在**PCoh**中，可以证明$\mathcal{M}\tau_1 \otimes \mathcal{M}\tau_2 \subseteq \mathcal{M}(\tau_1 \times \tau_2)$这意味着，类型为$\mathcal{M}\tau_1 \otimes \mathcal{M}\tau_2$的良类型程序由关于$\tau_1 \times \tau_2$的联合分布表示。此外，仔细考察定义5.5 可知， $\mu_A \otimes \mu_B$ 恰好对应于 μ_A 和的乘积分布

μ_B，因此我们的可靠性定理意味着，类型为 $\mathrm{M}\tau_1 \otimes \mathrm{M}\tau_2$ 的闭程序由相互独立的概率分布表示。

5.2.2 连续概率。 接下来，我们考虑连续概率的模型。对于共享层，**CountStoch** 向连续概率的推广是**BorelStoch**，其对象为标准博雷尔空间，态射为马尔可夫核 [Fritz 2020]；详情见附录B。对于分离层，我们需要一个能够解释连续随机性的线性逻辑模型。我们采用基于完备巴拿赫格的模型。

定义 5.9 (Azevedo de Amorim 和 Kozen [2022])。 范畴 $\mathbf{PBanLat}_1$ 的对象是完备巴拿赫格，态射是范数不超过一的序连续线性函数。

直观而言，完备巴拿赫格是配备了格结构和对合线性否定的巴拿赫空间。对于每个可测空间 $(X,\ \Sigma_X)$，其上的有符号测度空间都是完备巴拿赫空间，这意味着它例如可以解释实数直线上的连续概率分布。此外，将 $(X,\ \Sigma_X)$ 映射到其有符号测度空间的映射具有函子性和松幺半群性。

定理 5.10 (Azevedo de Amorim 和 Kozen [2022])。 存在一个松幺半函子 $\mathcal{M}$:$\mathbf{BorelStoch} \to \mathbf{PBanLat}_1$。

定理 5.11。 三元组 $(\mathbf{PBanLat}_1, \mathbf{BorelStoch}, \mathcal{M})$ 是一个 λ^2_{INI} 模型。

证明。 范畴 **BorelStoch** 具有 CD 结构和余积，因为它同构于范畴**Meas** [Fritz 2020]上的交换单子的 Kleisli 范畴。范畴$\mathbf{PBanLat}_1$ 是经典线性逻辑的一个模型，因此是具有余积的对称幺半闭范畴 [Azevedo de Amorim 和 Kozen 2022]。态射 del_V 是常量 0 函数，其中幺半单位为 $\mathbb{R}$。 $\mathcal{M}$ 的松幺半性由前一个定理推出。 □

这个模型可以看作前一个模型的连续推广，因为存在全忠实嵌入 $\mathbf{CountStoch} \hookrightarrow \mathbf{BorelStoch}$ 和 $\mathbf{PCoh} \hookrightarrow \mathbf{PBanLat}_1$ [Azevedo de Amorim 和 Kozen 2022]。在这个模型中，我们的可靠性定理再次确保了概率独立性，即类型为 $\mathcal{M}\tau_1 \otimes \mathcal{M}\tau_2$ 的程序由独立分布表示。

5.2.3 非确定性与通信。 接下来，我们将展示线性逻辑的关系模型如何产生一个 λ^2_{INI} 模型，并将其应用于分布式编程。

语义。 我们的出发点是由集合和二元关系构成的范畴**Rel**，它是线性逻辑最著名的模型之一。将此范畴与幂集单子 P 的 Kleisli 范畴$\mathbf{Set}_{\mathcal{P}}$配对后，我们立即得到一个 λ^2_{INI}的模型。

定理 5.12。 三元组$(\mathbf{Rel}, \mathbf{Set}_{\mathcal{P}}, id)$ 是一个 λ^2_{INI} 模型。

证明。集合上的二元关系 A 和 B 可以表示为子集 $R \subseteq A \times B$，或等价地，表示为函数 $A \to \mathrm{P}(B)$。根据这一观察，可以证明恒等函子是一个同构，由此可轻易推出 id 是松幺半群。由于**Rel** 是线性逻辑的模型，它具有余积；根据同构，$\mathbf{Set}_{\mathcal{P}}$也具有余积。**Rel** 中的自然变换$del$ 是$\mathbf{Set}_{\mathcal{P}}$中的删除操作。 □

面向分布式编程的应用。 尽管这一模型源于线性逻辑，但我们展示了它如何导出一种适用于分布式编程的语言。我们假设采用一种带通信的双层方法进行编程：使用 NI 语言编写局部程序，而使用 I 语言来编排本地代码之间的通信。类型为$\mathcal{M}\tau$ 的程序对应于可由通信语言操作的局部计算。

将 I 语言中的程序解释为形如 $A \to \mathcal{P}(B)$的映射；我们将这些映射视为允许非确定性的 或 有损的 通信。

为使语法与这一解释保持一致，我们将语法示例 t_i as x_i in M to send $t_i$as $x_i$in M调整为：它发送局部程序 t_i计算出的值，将它们绑定到 x_i，然后继续执行局部程序 M。为了了解如何使用这门语言编写分布式程序，我们考虑双方之间一个简单的分布式投票协议。假设有一名领导者接收包含投票结果的两条消息；如果二者相同，则选举结果确定，领导者宣布获胜者。如果投票结果不一致，领导者则输出一个带标签单位值，表示结果为平局。在 λ^2_{INI}中，领导者可以实现为：

$$\begin{aligned}&\text{leader} : \mathcal{M}\mathbb{N} \otimes \mathcal{M}\mathbb{N} \multimap \mathcal{M}(\mathbb{N} \oplus 1)\\&\text{leader} = \lambda\, x_1\, x_2.\, \text{send}\, x_1, x_2\, \text{as}\, n_1, n_2\, \text{in}\, \text{if}\, n_1 = n_2\, \text{then}\, (\text{in}_1\, n_1)\, \text{else}\, (\text{in}_2\, ())\end{aligned}$$

给定一个程序 votes： $\mathcal{M}\mathbb{N} \otimes \mathcal{M}\mathbb{N}$，该程序计算每个参与者将投给谁，完整的分布式程序可以表示为应用 leader votes。

可靠性定理。 在这个模型中，我们的可靠性结果保证：如果有一个类型为 $\mathcal{M}\tau_1 \otimes \mathcal{M}\tau_2$ 的闭程序，那么它可以分解为两个能够在本地运行的局部程序，除了 send 指令之外，不需要任何额外的通信。为了理解这一保证为何并非显然成立，请考虑第 4 节中的问题程序：

$$\text{消息} : \mathcal{M}(1+1) \vdash_I \text{如果消息则 (tt} \otimes \text{tt) 否则 (ff} \otimes \text{ff)} : \mathcal{M}\mathbb{B} \otimes \mathcal{M}\mathbb{B}$$

在我们的解释中，if 语句根据程序变量 消息 的内容进行条件判断，并生成两个输出相同的局部计算。这个程序中有两个潜在的隐式通信来源。首先，消息 的内容是不确定的，因此局部计算必须进行通信，才能就返回哪个值达成一致。其次，由于以相同的值为条件，必须将消息发送给两个局部计算。编舞语言文献已讨论过这些间接通信，例如 Hirsch 和 Garg [2022], ，他们的语言允许对局部计算进行模式匹配，但必须将选定的分支广播给依赖于它的程序。在通信可靠的情形下，这并不会造成问题。

为了说明可靠性保证，我们可以重新审视分布式投票示例。根据可靠性定理，程序 votes 等于 $t_1 \otimes t_2$，对于程序 t_1， $t_2 : \mathcal{M}\mathbb{N}$。因此，唯一需要的通信是显式发送。

表达能力与局限性。直观地说， λ^2_{INI} 中类型为 $\mathbf{M}\tau$ 的闭程序等价于在 M 中将 t_i作为 x_i 发送。我们将其视为一个局部程序 M：它首先接收 n 条不同的消息，然后将收到的消息作为绑定变量来运行其主体 M，并使其输出可供发送给另一个局部计算。因此，每个局部程序在开头只能有一个接收块，并在结尾只能有一次发送，这限制了允许的通信模式。

其他用于分布式程序设计的模态逻辑方法，通过设定一个静态的代理集合，并用该集合中的元素标注模态，以表示由分布式系统中的特定代理执行的计算，解决了这些局限性 [Hirsch 和 Garg2022]。我们推测，通过扩展 λ^2_{INI} 并加入类型构造子 $\mathcal{M}_\ell\tau$，其中 ℓ 是代理的名称，或许能够表示更复杂的通信模式，但我们将此留待未来工作。

相关工作。 分布式编程具有挑战性且容易出错，针对这一领域的语言设计由来已久。两个值得注意的例子是会话类型 [Hüttel 等人 2016]以及编舞式编程 [Montesi 2014]。会话类型采用线性类型纪律，其中

类型构造子用于对所需协议进行建模。另一方面，编舞式编程采用单体式方法：整个系统被编写成单个程序，然后可将其编译为“局部计算”，由编译器添加适当的通信指令。

我们的模型融合了两种方法的特征。它仍然具有次结构通信类型系统，但同时还使用带有双层语言的单一全局程序来表示协议，该双层语言区分局部计算和全局计算。关于这些语言的更深入比较留待未来工作。

5.2.4 交换效应。本节将介绍一大类基于交换单子的模型；交换单子是指，在效应的 Kleisli 语义中，程序等式（let $x = t$ in let $y = u$ in w） ≡ （let $y = u$ in let $x = t$ in w）成立的单子。

交换单子的 Kleisli 范畴具有许多有用的性质。

定理 5.13 （Fritz [2020]）。*Let*绑定 C 是一个笛卡尔范畴，并且 T 是其上的一个交换单子。范畴$\mathbf{C}_T$ 是一个 *CD* 范畴。

引理 5.14。*Let*绑定 C 是一个分配范畴，并且 T 是其上的一个单子。其 *Kleisli* 范畴 $\mathbf{C}_T$ 具有余积，并且也是分配的。

证明。很容易证明，Kleisli 范畴从基范畴继承余积。此外，利用C的分配结构，对其应用 T 并使用函子定律，可以推出$\mathbf{C}_T$ 是分配的。□

⁀另一个有用的代数范畴是由代数和普通映射构成的$\mathbf{C}^T$，其中对象是 T⁀代数，而映射为$\mathbf{C}((A, f), (B, g)) = \mathbf{C}(A, B)$。

定理 5.15 (Simpson [1992])。设 C 是一个笛卡尔闭范畴，且 T 是其上的一个交换单子。T-代数与普通映射构成的范畴是笛卡尔闭的，并且 1 是一个终对象。

因此，我们选择 Kleisli 范畴来解释 NI，并选择 T-代数与普通映射构成的范畴来解释 I。我们只需证明它们之间存在一个应用函子。

⁀**定理 5.16.** 存在一个应用函子 $\iota : \mathbf{C}_T \to \mathbf{C}^T$。

证明。该函子将对象 A 映射到自由代数（TA, μ_A），并将态射$f\colon A \to TB$映射到 f^*。现在，对于松弛幺半结构，考虑自然变换$\mu \circ T\tau \circ \sigma\colon TA \times TB \to T(A \times B)$ 和 $\eta_1\colon 1 \to T1$，其中 τ和 σ是 T的强度。利用 T 是交换的这一事实，以及余幺半群结构 $A \to 1$ 是自然的，可以证明这对应于一个应用函子。□

⁀**定理 5.17.** 三元组 $(\mathbf{C}^T, \mathbf{C}_T, \iota)$ 是一个 λ^2_{INI}模型。

名称生成。 交换效应的简单具体示例包括我们之前见过的概率和非确定性。一个更有趣的示例是用于赋予 ν-演算语义的名称生成单子；这是一种带有生成“新鲜”符号原语的语言 [Stark 1996]。这是一种有用的抽象，例如在密码学中，新符号可能是你不希望与对手分享的秘密。

为 ν-演算给出的具体语义由 Stark [1996] 提出，其中基范畴是函子范畴 [**Inj**，**Set**]，，其中 **Inj** 是有限集与单射函数构成的范畴。在这种情况下，（可交换的）名称生成单子作用于函子，其形式为

$$T(A)(s) = \{(s', a') \mid s' \in \mathbf{Inj}, a' \in A(s + s')\}/\sim$$

其中，当且仅当存在某个 s_0，使得存在单射函数 $f_1\colon s_1 \to s_0$和$f_2\colon s_2 \to s_0$，满足 $A(id_s + f_1)a_1 = A(id_s + f_2)a_2$。直观地说， $T(A)$ 是一种计算

给定一个有限的名称集合 s，它会生成新名称 s'，以及一个值 a'。根据定理 5.17 三元组 $([\mathbf{Inj},\mathbf{Set}]^{\mathcal{T}}, [\mathbf{Inj},\mathbf{Set}]_{\mathcal{T}}, \iota)$ 是一个 λ^2_{INI} 模型。

从语法上讲，我们可以扩展 *NI* 语言的类型语法，加入表示名称的类型名称；还可以扩展 *NI* 语言，加入名称生成操作 $\cdot \vdash \mathsf{fresh} : \text{名称}$。我们的可靠性定理指出，对于类型为 $\mathcal{M}\tau \otimes \mathcal{M}\tau$ 的程序，用于计算第一个分量的名称与用于计算第二个分量的名称互不相交。

也可以利用 Eilenberg-Moore 范畴来定义这一代数模型的一个变体，因为已知在一些较弱的假设下，该范畴是对称幺半闭的[Azevedo de Amorim 2023]。

备注 5.18（值调用与名调用的效应语义）。代数范畴和平凡映射被用作名调用编程语言的指称基础，而 Kleisli 范畴则可用于解释值调用语言 [Simpson 1992]。因此，I语言可视为效应的按名调用解释，而NI可视为效应的按值调用解释。对采样 t 作为 x 在 M 的操作性解释，是强制执行按名调用计算 t，将结果绑定到 x，并在程序中急切地运行它们 M。

5.2.5 仿射分束类型系统。 分束蕴涵逻辑（BI） [O'Hearn 和 Pym 1999] 是一种次结构逻辑，旨在推理资源的共享与分离，例如指向堆内存的指针 [O'Hearn 等人 2001], 或进入并发代码中某个临界区的权限 [O'Hearn 2007]。BI 的证明论导出了带有分束类型系统的函数式语言，其中上下文是树（所谓的分束），而非列表 [O'Hearn 2003]。

自然会想知道 BI 与 λ^2_{INI} 有何关系。从语义上说，分束演算使用双闭范畴（DCC）进行解释；这是一种同时具有笛卡尔闭结构和（通常不同的）幺半闭结构的单一范畴。为了理解这些系统之间的关系，我们来考察分束演算的仿射变体，即幺半单位是语义范畴中的终对象，这意味着存在一种丢弃操作 $A \otimes B \to A$。给定一个仿射 BI 模型C，存在一个态射 $A \otimes B \to A \times B$，它由丢弃态射 $A \otimes B \to A$ 和 $A \otimes B \to B$ 应用积的泛性质得到。此外，根据假设 $I \cong 1$，其中 1 是笛卡尔积的单位对象，而 I 是幺半积的单位对象。最后，这种结构使松幺半性图表成立，从而使恒等函子 $id\colon (\mathbf{C}, \mathbf{1}) \to (\mathbf{C}, I)$ 成为从C上的一种幺半结构到另一种幺半结构的松幺半函子。因此：

定理 5.19。对于每个仿射 *BI* 模型C，三元组 $(\mathbf{C}, \mathbf{C}\ id)$ 都是 λ^2_{INI} 的模型。

注记 5.20。从更抽象的角度来看，根据 λ^2_{INI}(定理 A.3) 所述句法模型的初始性，以及上述定理，我们可以得到一个从 λ^2_{INI} 到分束演算的翻译。因此，仿射分簇演算可以看作我们语言的一个退化版本，其中两层合并为一层。

干扰的句法控制。为了说明仿射分束演算的一个实用模型，我们来考察 O'Hearn 的分束语言 SCI+ [O'Hearn 2003]。这种语言允许分配内存并推理别名，其基础是 Reynolds 的干扰的句法控制（SCI），一种线性类型系统。在 SCI+的指称语义中，类型是函子范畴 $\mathbf{Set}^{\mathcal{P}(Loc)}$ 中的对象，其中 $\mathcal{P}(Loc)$ 是由 Loc 的子集构成的偏序集范畴；是一个由名称（即内存地址）组成的无限集合。直观来说，预层将位置的子集映射到使用这些位置的计算所构成的集合。众所周知，该范畴是仿射 BI 的一个模型：笛卡尔闭结构由预层上的通常构造给出，而幺半闭结构则由预层上的另一种积给出，称为 Day 卷积 [Borceux 1994]。

根据定理 5.19，三元组 $(\mathbf{Set}^{\mathcal{P}(Loc)}, \mathbf{Set}^{\mathcal{P}(Loc)}, id)$ 是一个 λ^2_{INI} 模型，因此满足其可靠性。为了理解在此上下文中这意味着什么，我们来看看该模型是如何

$$
\begin{array}{lll}
\text{类型} & \tau & ::= \quad \text{单元} \mid \text{表达式} \mid \text{命令} \mid \tau \to \tau \mid \tau \multimap \tau \mid \tau \times \tau \\
\text{上下文} & \Gamma & ::= \quad \cdot \mid x\colon \tau \mid \Gamma;\ \Gamma \mid \Gamma,\Gamma
\end{array}
$$

图 9. 类型与项：SCI+

$$
\frac{\Gamma \vdash M\colon \text{命令} \qquad \Gamma \vdash N\colon \text{命令}}{\Gamma \vdash M; N\colon \text{命令}}
\qquad
\frac{\Gamma_1 \vdash M\colon \text{命令} \qquad \Gamma_2 \vdash N\colon \text{命令}}{\Gamma_1,\ \Gamma_2 \vdash M||N\colon \text{命令}}
$$

$$
\frac{\Gamma,\ x\colon \text{单元} \vdash M\colon \text{命令}}{\Gamma \vdash \text{新建}\ x.M\colon\ \text{命令}}
\qquad
\frac{\Gamma \vdash M\colon\ \text{单元} \qquad \Gamma \vdash N\colon\ \text{表达式}}{\Gamma \vdash M := N\colon\ \text{命令}}
$$

$$
\frac{\Gamma \vdash M\colon\ \tau_1 \to \tau_2 \qquad \Gamma \vdash N\colon \tau_1}{\Gamma \vdash M\,N\colon \tau_2}
\qquad
\frac{\Gamma_1 \vdash M\colon \tau_1 \multimap \tau_2 \qquad \Gamma_2 \vdash N\colon \tau_1}{\Gamma_1,\ \Gamma_2 \vdash M\,N\colon\ \tau_2}
$$

图 10. 类型规则：SCI+(selected)

定义的。给定 P（Loc）上的预层 A 和 B，幺半积 $A \otimes B$ 定义为

$$(A \otimes B)(X) \triangleq \{(a,b) \in A(X) \times B(X) \mid support(a) \cap support(b) = \emptyset\}$$
$$(A \otimes B)(f) \triangleq (Af a, Bf b)$$

该 $support$ 函数作用于集合，其定义略显技术性，用于建模 Loc 中哪些资源被用于生成该集合——感兴趣的读者可参阅原论文 [O'Hearn2003]。从较高层面来看，支撑集的不相交性体现了这样一个事实：用于生成 a 的内存位置与用于生成 b的内存位置彼此不相交。因此，我们的可靠性定理保证，类型为$\mathcal{M}\tau_1 \otimes \mathcal{M}\tau_2$ 的闭程序各组件不会共享任何内存位置。

在语法层面上，SCI+ 演算与 λ_{INI}有一些相似之处，例如都具有两种不同的积类型，但也存在许多差异。例如，它有两种上下文拼接操作，因此能够兼容两种不同的箭头类型，如图 9所示。此外，它还具有基本类型 表达式、单元 和 命令，分别用于表示表达式、内存单元和命令，并提供了用于操作它们的原语操作。

就我们的目的而言，我们主要关注图 + 中介绍的 SCI 操作：10。前两条规则分别用于顺序执行或并行执行命令。接下来的两条规则与内存操作有关：第一条分配新的内存位置，第二条则将值赋给某个位置。最后两条是两种应用：第一种允许共享上下文，第二种则不允许。

有状态程序并行执行时，一个众所周知的难题是，可能会对同一个内存位置进行并发写入。SCI+ 通过使用上下文的分离拼接来避免这一问题，从而保证不会发生此类写入冲突。顺序组合程序时，不会出现这类问题，因此可以共享上下文。使用 new $x.M$语法分配新的内存单元时，会在表示新位置的上下文中绑定一个新变量；该位置与现有位置互不相交，因此形成分离的上下文扩展。

SCI+ 中的 λ^2_{INI}。 正如我们所解释的，定理 5.19 的一个直接推论是，可以将 λ^2_{INI} 翻译成 BI 演算。然而，单元和命令

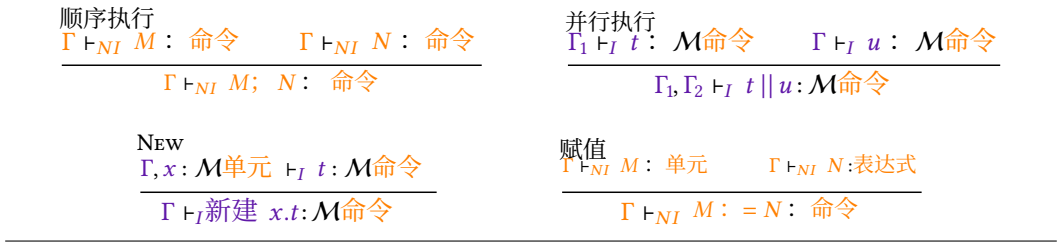


图11。 类型规则：λ^2扩展了 SCI 原语的 INI

操作可以采用与其原始表述类似的类型规则和语义。稍作修改 λ^2_{INI} 后，我们便可以将它们纳入其中，如图 11所示。顺序组合在 NI 语言中完成，而并行组合则在 I 语言中完成。单元赋值规则被加入 NI 语言，因为没有理由要求单元的地址和值必须分别使用不同的位置来计算。对于单元分配，原始规则要求新单元与现有单元互不相交，因此使用 I 语言是自然的。

示例 5.21（O'Hearn [2003]）。考虑程序（$\lambda^2_{\text{INI}}\, \lambda x\, y.\, x := 1; y := 2)\ z\ z$。对于该 λ-抽象，有两种可能的类型。类型 Mcell$\multimap$Mcell$\multimap$Mcomm 要求输入位置 x和 y必须互不相交，而类型 M(cell $\times$ cell)$\multimap$Mcomm 则允许 x和 y共享。前者使该应用不具良类型，因为抽象的参数相同；后者则具有良类型。不过请注意，它之所以具有良类型，仅仅是因为这些赋值采用顺序执行组合。若它们采用并行执行组合，该程序就会不具良类型，正如 SCI+ 中一样，因为并行组合要求内存位置互不相交。

表达能力更强 λ^2_{INI}。SCI+ 支持更细粒度的共享/不相交策略，可交错使用 $\times$ 和 $\otimes$ 类型构造子——这些程序难以用 λ^2_{INI}表达。例如，在我们的语言中无法表示类型 M($A \otimes B$) $\times$M ($C \otimes D$)。这是因为只有一个模态可将 NI 语言映射到 I 语言，而没有模态可反向映射。这一局限也体现在下面这个无法用 λ^2_{INI}表达的简单程序中：$x := 1; (y := 2)\ ||\ (z := 3)$。该程序类型错误，因为只有 NI 程序才能顺序组合，只有 I 程序才能并行组合。然而，在具体模型中，松幺半函子是恒等函子，因此可以在 NI 类型语法中添加子句 $\tau := \tau\ |\cdots$，使下面的类型规则成立：

$$\frac{\Gamma \vdash_I t : \tau}{\Gamma \vdash_{NI} t : \tau}$$

这使得对上文中棘手的程序进行类型检查成为可能。

6 可靠性定理

到目前为止，我们已经看到了两种可靠性证明。对于 λ_{INI}，我们使用逻辑关系证明了可靠性（定理 3.3）。对于 λ^2_{INI} 采用概率语义的情形，我们利用了关于分布单子代数的一条观察结果（定理 4.1）。这个证明很巧妙，但这种策略无法推广到其他 λ^2_{INI}模型。

因此，为了证明关于 λ^2_{INI}的一般健全性定理，我们将回到逻辑关系。我们的可靠性定理陈述如下。

定理 6.1. 如果 $\cdot \vdash_I t : \mathcal{M}\tau_1 \otimes \mathcal{M}\tau_2$ 那么 $[\![t]\!]$ 可以分解为两个态射 $[\![t]\!] = f_1 \otimes f_2$，其中 $f_1 : I \to \mathcal{M}[\![\tau_1]\!]$ 且 $f_2 : I \to \mathcal{M}[\![\tau_2]\!]$。

逻辑关系常用于证明类型理论和编程语言的元理论性质。然而，它们通常应用于具体情形，也就是说，针对某个具体模型，我们可以在其中显式定义逻辑关系。但在我们的情形中，这种方法还不够，因为我们研究的是 λ^2_{INI}的抽象范畴语义。因此，我们将借助逻辑关系的范畴化处理，称为*Artin* 粘合，这是一种最初用于托扑理论的构造 [Hyland 和 Schalk 2003年；Johnstone 等人 2007]。

本文不在此详细介绍这一技术，但会在此强调其中一些关键方面。我们已经介绍了适用于 λ^2_{INI}的模型类别。Let $\cdot \vdash_I t :_ \tau$ 是一个类型正确的程序。对于每个具体模型(C，M，$\mathcal{M}$)，我们希望证明该模型中的解释 $[\![t]\!]$满足某些性质。概括而言，粘合论证分为三个步骤：

(1) 定义一个 λ^2_{INI}的模型范畴，并证明每个解释 $[\![\cdot]\!]$ 都可以编码为从句法模型Syn 到（C，M，M）的映射；其中，句法模型以类型为对象，以类型推导（模去 λ^2_{INI}的等式理论）为态射。证明句法模型是初始模型，即可推出这一性质。(2) 定义一个三元组（Gl(C)，M，M）——其中，范畴Gl(C)的对象是序对（A，$X \subseteq$C（I，A）），子集X被视为A上的谓词，态射保持这些谓词

并证明这一结构是 λ^2_{INI}的模型。我们称之为粘合模型，并且存在一个显然的遗忘模型态射（Gl(C), M, M）→（C, M, M）。(3) 利用初始性，定义一个映射 $(\!|\cdot|\!)$，从句法模型Syn到粘合模型。该映射的数据将 λ^2_{INI} 中的每个I-类型 τ关联到一个对象（A_τ，$X_\tau \subseteq$ C（I，A_τ））；直观而言，$A_\tau \in$C是 τ 在 $[\![\cdot]\!]$下的解释，而子集 X_τ 编码了类型τ处的逻辑关系，因此，该映射定义了一个逻辑关系。函子 τ 及其陪域编码了逻辑关系证明。

最后，我们可以将句法范畴中的任意态射，即类型正确的项 $(\!|t|\!)$，映射为 $\cdot \vdash_I t : \tau$ 的一个元素。由的初始性可知，Syn，$[\![t]\!]$ 也是 X_τ 的一个元素，从而通过逻辑关系证明完成证明。详细内容见附录 A。

7 相关工作

线性逻辑与概率程序。近年来，一系列研究将线性逻辑用作强大的框架，为概率编程语言提供语义。值得注意的是，Ehrhard 等人[2018] 证明，线性逻辑相干空间语义的一个概率版本，对于带离散选择的概率 PCF 是完全抽象的；而 Ehrhard 等人 [2017] 为一种支持连续随机采样的高阶概率语言提供了受线性逻辑启发的指称语义；此外，也有人开发了按推值的概率版本 [Tasson 和 Ehrhard2019]。此外，针对几乎必然终止等概率性质 [Dal Lago 和 Grellois 2019]，以及差分隐私 [Azevedo de Amorim 等人 2019；Reed 和 Pierce 2010]，研究者也开发了线性类型系统。

正如我们提到的，我们的范畴模型 λ^2_{INI} 受到基于幺半群伴随的线性逻辑模型的启发，其中最著名的是 Benton 的 LNL [Benton 1994]。从编程语言的角度来看，这些模型将带指数模态的线性 λ演算分解为两种语言，每种语言都有不同的积类型：一种是笛卡尔积，另一种是对称幺半群积。伴随关系通过在每种语言中加入函子型类型构造子来体现，这与我们的 $\mathcal{M}$ 模态类似。这些双层语言与 λ^2_{INI}非常相似，而且确实可以

证明每个 LNL 模型都是一个 λ^2_{INI} 模型。同时，λ^2_{INI} 的模型类别比 LNL 广泛得多——第 5.2 节中介绍的模型都不是 LNL 模型。

高阶程序与效应。关于高阶程序效应，已有大量研究成果，我们无法在此一一概述。λ_{INI} 的语义是 Moggi 的 Kleisli 语义的一个实例，该语义源自他关于单子效应的奠基性工作 [Moggi 1991]；不同之处在于，我们的单层语言使用线性类型系统来强制保证概率独立性。

该领域的另一项著名工作是按推值（CBPV） [Levy 2001]。它是一种用于处理效应的两层元语言，同时涵盖值调用和名调用语义。每一层都有一种模态，将其中一层映射到另一层。它与 λ^2_{INI}有相似之处，但二者之间的确切关系尚不清楚——我们的具体模型中没有一个是 CBPV 模型。

我们的两层语言 λ^2_{INI} 也可以被视为对一种新颖的线性逻辑资源解释的应用，该解释由 Azevedo de Amorim [2023],提出，它使用应用式模态来保证线性限制仅适用于计算，而不适用于值。我们关注分离和效应：我们展示了如何在这一框架中自然地容纳带效应计算的不同和类型，考虑了更一般的范畴模型，并证明了一个可靠性定理，以确保带效应计算的分离性。

分束类型系统。我们对共享与分离的关注，与另一种次结构逻辑——分块蕴涵逻辑（BI）——的动机相似 [O'Hearn 和 Pym 1999]。与我们的系统一样，BI 包含两种合取，分别用于建模资源分离和资源共享。与 λ_{INI}中的情况一样，BI 的这两种合取属于同一种语言。与我们的工作不同，BI 还包含两种蕴涵，分别对应每一种合取。BI 的主要应用是用于并发程序和堆操作程序的分离逻辑 [O'Hearn 2007； O'Hearn 等人 2001], 其中前置条件和后置条件取自 BI。

BI 的大多数应用采用真值函数式的克里普克式语义 [Pym 等人 2004]。通过考察 BI 的证明论模型，O'Hearn [2003] 为一种高阶语言开发了一种分束类型系统。其范畴语义由一个双闭范畴给出：一个笛卡尔闭范畴，并带有独立的对称幺半群闭结构。 O'Hearn [2003] 展示了这种语言的不同模型，用于对堆中的共享与分离进行推理，但已知的其他具体模型寥寥无几。目前尚不清楚如何将效应纳入分束类型系统；相比之下，我们的模型能够对广泛的单子效应进行推理。

这两种语言都与我们的语言有着自然的联系。我们的语言 λ_{INI} 与 O'Hearn 的系统相似，但有两点不同。首先， λ_{INI} 只有乘法箭头，没有加法箭头——正如我们在第 3 节所述，目前尚不清楚如何在 λ_{INI} 中支持加法箭头，同时不破坏我们的主要可靠性性质。其次， λ_{INI} 中的上下文是扁平列表，而非树形分束；使用 bunched 上下文来表示更复杂的依赖关系将会很有意思。

我们的分层语言 λ^2_{INI} 也与 O'Hearn 的系统相似。尽管我们的范畴模型只有单一的乘法箭头，但在 I 层中，我们的许多（但并非全部）具体模型也在 NI 层中支持加法箭头。此外，在 BI 中，若假设使用单一范畴，而非像我们的方法那样使用两个范畴，就可以将联结词分层 $\times$ 和 $\otimes$，从而构造复杂的依赖结构。相比之下，我们的两层语言只允许构造以下形式的依赖关系： $\mathcal{M}(\tau \times \cdots \times \tau) \otimes \cdots \otimes \mathcal{M}(\tau \times \cdots \times \tau)$。与此同时，目前尚不清楚 λ^2_{INI} 中的两种和类型在 bunched 类型系统中将如何运作。

高阶语言中的概率独立性。有一些概率函数式语言配备了能够对概率独立性进行建模的类型系统。其中最为复杂的例子可能要数 Darais 等人 [2019], 提出的系统，他们为概率函数式语言设计了一种结合线性性、信息流控制和概率区域的类型系统。Darais 等人 [2019]

展示了如何使用他们的系统，为不经意随机存取机（ORAM）的实现实现并验证安全属性。我们的工作旨在构建一种捕获独立性的核心演算，并配备简洁的范畴模型。

Lobo Vesga 等人 [2021] 提出了一种嵌入 Haskell 的概率函数式语言，旨在验证程序中与差分隐私相关的准确性属性。他们的系统采用污点分析来确立独立性，这是可靠应用切尔诺夫界等集中界所必需的。与我们的工作不同，Lobo Vesga 等人 [2021] 没有在核心演算中形式化其独立性属性。

概率分离逻辑。近期一系列工作针对一阶命令式概率程序发展了分离逻辑，并使用分束蕴涵逻辑中的公式表示前置条件和后置条件。这些系统能够推理概率独立性 [Barthe 等人 2019], , 也能推理条件独立性等细化属性 [Bao 等人 2021],以及负关联 [Bao 等人 2022]。这些系统利用不同的 Kripke 风格模型来表示逻辑断言；目前尚不清楚如何将这些思路改编为类型系统或高阶语言。此外，还有定量概率分离逻辑 [Batz 等人 2022, 2019]。

8 结论与未来方向

我们提出了两种线性高阶语言，其类型能够刻画概率独立性，以及效应程序中的其他分离概念。我们认为，未来有几个值得进一步研究的自然方向。

独立性的其他变体。从某种意义上说，概率独立性是依赖性的一种平凡情形：它刻画了两个随机量之间完全不存在依赖性的情况。统计学和人工智能领域的研究者还考虑了其他能够建模更精细依赖关系的概念，例如条件独立性、正关联和负依赖（例如， [Dubhashi 和 Ranjan 1998]）。其中一些概念已被推广到概率之外的其他模型；例如，Pearl 和 Paz [1986]提出了一套图拟阵理论，用以公理化条件独立性的性质。若能探究这些概念中是否有任何一种可以在类型系统中刻画，将会很有意思。

用于独立性的分束类型系统。我们的工作与分束逻辑方面的研究有许多相似之处；最值得一提的是，分束逻辑包含加法合取和乘法合取。尽管分束逻辑已在霍尔风格逻辑中得到广泛应用，但据我们所知，唯一的 bunched 类型系统由O'Hearn [2003]提出。这种语言包含单层结构，其中有两种积类型和两种函数类型；其类型上下文是树形分束，而非扁平列表。为具有更丰富上下文结构的语言构建概率模型，将是一个值得探索的未来研究方向。

非交换效应。我们的具体模型涵盖了多种单子效应，但我们只支持由交换单子建模的效应。许多常见效应由非交换单子建模，例如全局状态单子。我们或许可以扩展语言以处理非交换效应，但很可能需要泛化我们的模型，并考虑非交换逻辑。

迈向效应分离的一般理论。 我们已经看到，在存在效应时，和类型、积等构造会以两种形式出现，我们将其解释为共享与分离。共享与分离的概念早已在编程语言和逻辑领域得到研究，尤其催生了分离逻辑。我们认为，效应程序的分离（及共享）应当有一个更广泛的理论，而这一理论仍有待发展。

参考文献

Arthur Azevedo de Amorim、Marco Gaboardi、Justin Hsu 和 Shin-ya Katsumata。2019。通过度量进行概率关系推理。载于 ACM/IEEE计算机科学逻辑研讨会（LICS），温哥华，不列颠哥伦比亚省。IEEE，1–19。DOI：http://dx.doi.org/10.1109/LICS.2019.8785715 Pedro H. Azevedo de Amorim。2023。马尔可夫核与线性算子的高阶语言。载于软件科学与计算结构基础（FoSSaCS），法国巴黎。Pedro H Azevedo de Amorim 和 Dexter Kozen。2022。完美巴拿赫空间中的经典线性逻辑。预印本（2022）。Jialu Bao、Simon Docherty、Justin Hsu 和 Alexandra Silva。2021 2021 36 1 14 . 一种用于条件独立性的分簇逻辑。载于第 th 届 ACM/IEEE计算机科学逻辑研讨会（LICS）。IEEE，–。Jialu Bao、Marco Gaboardi、Justin Hsu 和 Joseph Tassarotti。2022。用于负依赖的分离逻辑。ACM《编程语言》论文集 6，POPL（2022），1–29。Gilles Barthe、Justin Hsu 和 Kevin Liao。2019。概率分离逻辑。ACM《编程语言》论文集 4，POPL（2019），1–30。Kevin Batz、Ira Fesefeldt、Marvin Jansen、Joost-Pieter Katoen、Florian Keßler、Christoph Matheja 和 Thomas Noll。2022。定量分离逻辑蕴涵检验基础。载于编程语言与系统——第 31 届欧洲编程研讨会（ESOP 2022），作为欧洲软件理论与实践联合会议（ETAPS 2022）的一部分举行，德国慕尼黑，2022年4月2日至7日，论文集（《计算机科学讲义》），Ilya Sergey（编），第 13240 卷。施普林格，57–84。DOI：http://dx.doi.org/10.1007/978-3-030-99336-8_3 Kevin Batz、Benjamin Lucien Kaminski、Joost-Pieter Katoen、Christoph Matheja 和 Thomas Noll。2019。定量分离逻辑：用于推理概率指针程序的逻辑。ACM 程序设计语言会议录 3，POPL（2019），34:1–34:29。DOI：http://dx.doi.org/10.1145/3290347 P. N. Benton。1994。一种混合线性与非线性逻辑：证明、项与模型（扩展摘要）。载于计算机科学逻辑国际研讨会（CSL），波兰卡齐米日（《计算机科学讲义》），Leszek Pacholski 和 Jerzy Tiuryn（编），第 933 卷。施普林格，121–135。DOI：http://dx.doi.org/10.1007/BFb0022251 Francis Borceux。1994。《范畴代数手册：第 2 卷，范畴与结构》。第 2 卷。剑桥大学出版社。G. E. P. Box 和 Mervin E. Muller。1958年。正态随机偏差生成注记。《数学统计年刊》29，2（1958年），610–611。DOI：http://dx.doi.org/10.1214/aoms/1177706645 Kenta Cho 和 Bart Jacobs。2019。通过字符串图进行析解与贝叶斯反演。《数学结构与计算机科学》29，7（2019），938–971。DOI：http://dx.doi.org/10.1017/S0960129518000488 Ugo Dal Lago 和 Charles Grellois。2019。通过单子仿射大小类型系统实现概率终止。《ACM 程序语言汇刊》41，2（2019），10:1–10:65。DOI：http://dx.doi.org/10.1145/3293605 Vincent Danos 和 Thomas Ehrhard。2011。作为高阶概率计算模型的概率相干空间。《信息与计算》209，6（2011），966–991。David Darais、Ian Sweet、Chang Liu 和 Michael Hicks。2019。一种用于概率无关计算的语言。ACM《编程语言》论文集 4，POPL（2019），1–31。Devdatt P. Dubhashi 和 Desh Ranjan。1998 13，2 1998 99 124 . 球与箱：负依赖研究。《随机结构与算法》（），–。Thomas Ehrhard、Michele Pagani 和 Christine Tasson。2017。可测锥与稳定可测函数：概率高阶编程模型。载于编程语言原理（POPL）。Thomas Ehrhard、Michele Pagani 和 Christine Tasson。2018。概率 PCF 的完全抽象。《美国计算机学会会刊》65，4（2018），23:1–23:44。DOI：http://dx.doi.org/10.1145/3164540 Tobias Fritz。2020。关于马尔可夫核、条件独立性及充分统计量定理的综合方法。《数学进展》370（2020），107239。Andrew K Hirsch 和 Deepak Garg。2022。Pirouette：高阶类型化函数编排。ACM《编程语言》论文集 6，POPL（2022），1–27。Hans Hüttel、Ivan Lanese、Vasco T. Vasconcelos、Luís Caires、Marco Carbone、Pierre-Malo Deniélou、Dimitris Mostrous、Luca Padovani、António Ravara、Emilio Tuosto、Hugo Torres Vieira 和 Gianluigi Zavattaro。2016。会话类型与行为契约基础。《ACM 计算调查》49，1，文章 3（2016年4月），36 页。DOI：http://dx.doi.org/10.1145/2873052 Martin Hyland 和 Andrea Schalk。2003。线性逻辑模型的粘合与正交性。《理论计算机科学》294，1-2（2003），183–231。Peter T Johnstone、Stephen Lack 和 Paweł Sobociński。2007。拟拓扑范畴、拟粘合范畴与阿廷粘合。载于计算机科学代数与余代数国际会议。施普林格，312–326。Tom Leinster。2014。范畴论基础。第 143 卷。剑桥大学出版社。Paul Blain Levy。2001。按推值。博士学位论文。Elisabet Lobo Vesga、Alejandro Russo 和 Marco Gaboardi。2021。一种带精度估计的数据隐私编程语言。《ACM 程序语言汇刊》43，2（2021），6:1–6:42。DOI：http://dx.doi.org/10.1145/3452096

Saunders Mac Lane。2013年。《工作数学家的范畴论》。第5卷。施普林格科学与商业媒体。Eugenio Moggi。1991年。《计算概念与单子》。Inf. Comput. 93, 1 (1991), 55–92。DOI:http://dx.doi.org/10.1016/0890-5401(91)90052-4 Fabrizio Montesi。2014年。编舞式编程。博士学位论文。丹麦。Peter W. O'Hearn。2003年。关于分束类型系统。J. Funct. Program. 13, 4 (2003), 747–796。DOI:http://dx.doi.org/10.1017/S0956796802004495 Peter W. O'Hearn。2007年。分离逻辑与并发资源管理。载于《第六届国际内存管理研讨会论文集》，ISMM 2007，蒙特利尔，魁北克，加拿大，2007年10月21—22日，Greg Morrisett 和 Mooly Sagiv（编）。ACM，1。DOI:http://dx.doi.org/10.1145/1296907.1296908 Peter W. O'Hearn 和 David J. Pym。1999年。分束蕴涵逻辑。Bull. Symb. Log. 5, 2 (1999), 215–244。DOI:http://dx.doi.org/10.2307/421090 Peter W. O'Hearn、John C. Reynolds 和 Hongseok Yang。2001年。关于改变数据结构的程序的局部推理。载于《计算机科学逻辑》，第15届国际研讨会 CSL 2001。EACSL第10届年度会议，法国巴黎，2001年9月10—13日，论文集（《计算机科学讲义》），Laurent Fribourg（编），第2142卷。施普林格，1–19。DOI:http://dx.doi.org/10.1007/3-540-44802-0_1 Judea Pearl 和 Azaria Paz。1986年。图拟阵：用于推理相关关系的基于图的逻辑，或者，当你已经知道 z 时，x 何时会告诉你更多关于 y 的信息？载于欧洲人工智能会议（ECAI），英国布莱顿，Benedict du Boulay、David C. Hogg 和 Luc Steels（编）。North-Holland，357–363。David J. Pym、Peter W. O'Hearn 和 Hongseok Yang。2004年。可能世界与资源：BI语义。Theor. Comput. Sci. 315, 1 (2004), 257–305。DOI:http://dx.doi.org/10.1016/j.tcs.2003.11.020 Jason Reed 和 Benjamin C. Pierce。2010年。距离使类型更强大：一种差分隐私演算。载于ACM SIGPLAN国际函数式编程会议（ICFP），马里兰州巴尔的摩，Paul Hudak 和 Stephanie Weirich（编）。ACM，157–168。DOI:http://dx.doi.org/10.1145/1863543.1863568 Alex K Simpson。1992年。Kleisli 范畴中的递归类型。未发表手稿，爱丁堡大学（1992）。Ian Stark。1996年。局部名称的范畴模型。Lisp与符号计算 9, 1 (1996), 77–107。Christine Tasson 和 Thomas Ehrhard。2019年。概率化推值调用。计算机科学中的逻辑方法（2019）。

范畴论可靠性证明：细节

A.1 模型范畴

λ^2_{INI} 的一个模型由一个带余积的 CD 范畴$\mathbf{M}$ 、一个带余积的对称幺半闭范畴$\mathbf{C}$ 以及一个松幺半函子 $\mathbf{M} : \mathbf{M} \to \mathbf{C}$给出。两个模型（$\mathbf{M}_1$,$\mathbf{C}_1$,$\mathbf{M}_1$）和（$\mathbf{M}_2, \mathbf{C}_2, \mathbf{M}_2$）之间的态射是一对保持逻辑联结词的函子（$F\colon \mathbf{M}_1 \to \mathbf{M}_2, G\colon \mathbf{C}_1 \to \mathbf{C}_2$）。通过逐分量定义态射复合，并将（$id_{\mathbf{C}}, id_{\mathbf{M}}$）作为恒等态射，该结构构成一个范畴，我们称之为$\mathbf{Mod}$。

在类型理论的范畴论处理中，证明等式理论是范畴语义的可靠近似非常重要。对于 λ^2_{INI}，由于该语言不使用任何复杂的类型理论构造，因此可靠性可以通过对类型推导进行归纳法而直接证明。

定理 A.1。 令 $(\mathbf{C}, \mathbf{M}, \mathcal{M})$ 是一个 λ^2_{INI} 模型。如果 $\Gamma \vdash_{NI} M \equiv N : \tau$ 那么 $[\![M]\!] = [\![N]\!]$ 并且如果$\Gamma \vdash_I t \equiv u : \tau$ 那么 $[\![t]\!] = [\![u]\!]$。

主要的微妙之处在于，我们必须更精确地呈现 I 语言的等式理论。请注意，sample 结构可以同时从任意多个分布中进行采样，而应用函子只提供二元采样算子。从形式上说，只需将 sample 限制为两个参数，并将以下规则添加到

等式理论。

$$\frac{\Gamma_i \vdash_I t_i : \mathrm{M}\tau_i \qquad i \in \{1,2,3\}}{\begin{array}{c}\Gamma_1,\ \Gamma_2,\ \Gamma_3 \vdash_I \text{采样 } t_1,\ (\text{采样 } t_2,\ t_3 \text{作为 } x_2,\ x_3 \text{在 } (x_2,\ x_3)) \text{作为 } x_1,\ y \text{在 } (x_1,\ \pi_1\, y,\ \pi_2\, y) \equiv \\ \text{采样 } (\text{采样 } t_1,\ t_2 \text{作为 } x_1,\ x_2 \text{在 } (x_1,\ x_2))\ t_3 \text{作为 } y,\ x_3 \text{在 } (\pi_1\, y,\ \pi_2\, y,\ x_3) : \mathrm{M}(\tau_1 \times \tau_2 \times \tau_3)\end{array}}$$

$$\frac{\Gamma \vdash_I t : \mathrm{M}\tau}{\Gamma \vdash_I \text{采样 } t,\ (\text{采样 } _ \text{作为 } _ \text{于 } ()) \text{作为 } x,\ y \text{于 } x \equiv t : \mathrm{M}\tau}$$

$$\frac{\Gamma \vdash_I t : \mathrm{M}\tau}{\Gamma \vdash_I \text{采样 } (\text{采样 } _ \text{作为 } _ \text{于 } ())\ t \text{作为 } x,\ y \text{于 } y \equiv t : \mathrm{M}\tau}$$

请注意，尽管这条规则看起来令人望而生畏，但它本质上是松弛幺半群交换图的语法形式，它说明 sample 操作满足结合律，因此，定义 n 元运算 sample $t_1, \ldots t_n$ 为 $x_1, \ldots\ldots, x_n$ in M（$n \geq 2$）的方式是唯一的。

一个重要的 λ^2_{INI} 模型是句法对象 **Syn**，它是一个三元组 $(\mathbf{Syn}^{lin}, \mathbf{Syn}^{CD}, \mathcal{M})$，其中 $\mathbf{Syn}^{CD}$ 是具有余积的 CD 范畴的句法范畴，而 $\mathbf{Syn}^{lin}$ 是具有余积和应用模态的对称幺半闭范畴的句法范畴，并且 $\mathcal{M}$ 是该模态的类型构造子。具体而言，这些范畴的对象都是类型，而态射是带有一个自由变量的程序，并且按图 8中给出的等式理论取模。简单检查可知，**Syn** 是一个 λ^2_{INI} 模型。

定理 A.2. Syn 是一个λ^2_{INI} 模型。

定理 A.3。 **Syn** 是**Mod**的初始对象。

证明。设 (C, M, M) 是一个模型。通过定义两个函子 $[\![\cdot]\!]_1 : \mathrm{Syn}_{lin} \to \mathrm{C}$ 和 $[\![\cdot]\!]_2 : \mathrm{Syn}_{CD} \to \mathrm{M}$，可以构造一个态射 $[\![\cdot]\!] : \mathrm{Syn} \to (\mathrm{C}, \mathrm{M}, \mathrm{M})$。由于 Syn^{lin} 和 Syn^{CD} 是自由生成的，函子对对象的作用可通过对类型进行简单归纳来刻画。对态射的作用则通过对类型推导进行归纳来定义，具体见图 7。

证明此函数定义良好的过程遵循定理A.1。唯一性可通过假设存在两个语义，并利用类型推导归纳证明它们相等来得到。 □

A.2 粘合范畴

我们通过使用逗号范畴来构造逻辑关系范畴。形式上，沿函子 $F : \mathrm{C}_1 \to \mathrm{D}$ 和 $G : \mathrm{C}_2 \to \mathrm{D}$ 的逗号范畴，其对象为三元组 (A, X, h)，其中 A 是一个 C_1 对象，X 是一个 C_2 对象，且 $h : FA \to GX$；其态射 $(A, X, h) \to (A', X' h')$ 是序对 $f : A \to A'$ 和 $g : X \to X'$，使某些图表交换。在计算机科学中的粘合应用里，通常假设 F 是恒等函子，且 $\mathrm{D} = \mathbf{Set}$。此外，为简化问题，有时也假设我们处理的是粘合范畴的满子范畴；例如，可以假设我们只关心满足 $A \to GB$ 是单射的对象，从而有效地表示 GB 的一个子集。

因此，在我们所关注的情形中，沿函子 G 构造的粘合范畴：$\mathrm{C} \to \mathbf{Set}$ 以序对（$A,\ X \subseteq G(A)$）为对象，其态射（$A,\ X$）$\to (B,\ Y)$ 是一个C态射 $f : A \to B$，使得 $G(f)(X) \subseteq Y$。请注意，这一条件可以看作是对箭头类型通常的逻辑关系解释的一种更抽象的表述：将相关的事物映射为相关的事物。在直观层面上，我们希望利用函子 G 将类型映射为其元素所满足的谓词。

现在，我们已准备好定义粘合范畴，并证明它构成该$\widetilde{\ }$语言的一个模型。给定一个三元组 $(\mathbf{M}, \mathbf{C}, \mathbf{M})$，我们定义三元组 $(\mathbf{M}, \mathbf{Gl}(\mathbf{C}), \mathbf{M})$，其中 $\mathbf{Gl}(\mathbf{C})$ 的对象

是序对 $(A \in \mathbf{C}\ X \subseteq \mathbf{C}(I, A))$，而态射是保持 X 的 C 态射，即我们沿着全局截面函子 $\mathbf{C}(I, -)$ 粘合 C。函子 $\mathbf{M} : \mathbf{M} \to \mathbf{C}$ 被提升为一个函子 $\widetilde{\mathbf{M}} : \mathbf{C} \to \mathbf{Gl}(\mathbf{C})$：它将对象 X 映射到 $(\mathbf{M}\ X, \mathbf{C}(I, \mathbf{M}\ X))$，并将态射 f 映射到 $\mathbf{M}\ f$。[1] 现在我们必须证明，该三元组确实是我们语言的一个模型。

有一点能简化我们的证明：Gl(C)中的态射就是带有额外结构的C 中的态射，而复合保持不变。因此，一旦我们证明某个C 态射也是Gl(C)中的态射，要证明某个Gl(C)图表交换，只需证明相应的C图表交换。

定理 A.4. Gl(C) 是一个具有余积和自然变换 *del*的对称幺半闭范畴。

证明。令(A, X)和(B, Y)为Gl(C)中的对象，我们定义$(A, X) \otimes (B, Y) = (A \otimes B, \{f \colon I \to A \otimes B \mid f = f_A \otimes f_B, f_A \in X, f_B \in Y\})$。幺半单位由$(I, \mathbf{C}(I, I))$给出，而自然变换 del 与C中的自然变换相同；它是Gl(C)中的态射，因为 $X_I = \mathbf{C}(I, I)$。

设 (A, X) 和 (B, Y) 是 Gl(C) 中的对象，我们定义 $(A, X) \multimap (B, Y) = (A \multimap B, \{f \colon I \to (A \multimap B) \mid \forall f_A \in X_A, \epsilon_B \circ (f_A \otimes f) \in X_B\})$，其中 $\epsilon_B : (A \multimap B) \otimes A \to B$ 是幺半闭伴随的余单位。

为证明 $A \otimes (-) \dashv A \multimap (-)$，我们可以使用伴随的（余）单位刻画，它对应于存在两个自然变换 $\epsilon_B \colon A \otimes (A \multimap B) \to B$ 和 $\eta_B \colon B \to A \multimap (A \otimes B)$，使得 $1_{A \otimes -} = \epsilon (A \otimes -) \circ (A \otimes -)\eta$ 且 $1_{A \multimap -} = (A \multimap -)\epsilon \circ \eta (A \multimap -)$，其中 1_F是 F到其自身的恒等自然变换。由于按定义，这些自然变换在 C 中满足伴随方程，因此只需通过展开定义，证明它们也是 Gl(C) 的态射。

最后，我们可以证明 Gl(C) 有余积。令 (A_1, X_1) 和 (A_2, X_2) 为 Gl(C) 对象，我们定义 $(A_1, X_1) \oplus (A_2, X_2) = (A_1 \oplus A_2, \{\mathsf{in}_i\ f_i \mid f_i \in X_i\})$。接下来证明它满足和类型的泛性质。令 $f_1 \colon A_1 \to B$ 和 $f_2 \colon A_2 \to B$ 为 Gl(C) 态射。考虑 C 态射 $[f_1, f_2]$。我们要证明这个态射也是 Gl(C) 态射。考虑$g \in X_{A_1 \oplus A_2}$，根据假设，$g =$属于$_1 g_1$ 或 $g =$属于$_2$。通过分情况分析以及事实 $f_i \circ g_i \in$和 $[f_1, f_2] \circ \mathsf{in}_i g_i = f_i \circ g_i$，我们可以得出结论：$[f_1, f_2]$ 确实是 Gl(C) 态射。□

这些构造在范畴逻辑文献中已有所论述 [Hyland 和 Schalk 2003], 不过，由于它足够简单，我们认为在此也加以介绍是有帮助的。迄今为止的每个构造都使用与C相同的对象，因此可以证明遗忘函子$U : \mathbf{Gl}(\mathbf{C}) \to \mathbf{C}$ 保留每个类型构造子，并且是一个 **Mod** 态射。接下来，我们必须证明 $\mathcal{M}$ 是松幺半群的，这可由以下事实推出：μ 和 ϵ 保留图集，只需简单展开定义即可。现在很容易得出结论：松幺半性图表是可交换的，因为复合相同，且 $\mathcal{M}$ 是松幺半群的。

因此，粘合范畴是一个模型。

定理 A.5. 三元组 (M,Gl(C), $\mathcal{M}$) 是一个 **Mod** 对象。

存在一个从粘合模型到原模型的遗忘映射。

定理 A.6. 存在一个 **Mod** 态射 $U : (\mathbf{M}, \mathbf{Gl}(\mathbf{C}), \mathcal{M}) \to (\mathbf{M}, \mathbf{C}, \mathcal{M})$。

最后，利用Syn的初始性，我们可以证明

定理 A.7. 存在一个Mod 态射 $(\!|\cdot|\!) : \mathbf{Syn} \to (\mathbf{M}, \mathbf{Gl}(\mathbf{C}), \mathcal{M})$。

[1]请注意，其谓词集是每个 C 态射 $I \to \mathbf{M}\ X$，这类似于在通常的逻辑关系证明中对基本类型的解释方式。

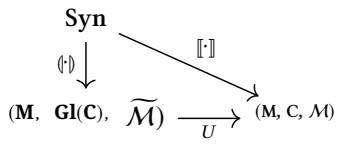


图12. 可靠性证明的本质

有了这个映射，我们现在可以构造一个函子 $U \circ (|\cdot|) : \mathbf{Syn} \to (\mathbf{M}, \mathbf{C}, \mathcal{M})$ 根据**Syn**的初始性，该函子等于 $[\![\cdot]\!]$，如图 12所示。

A.3 一般可靠性定理

定理 A.8。 若 $\cdot \vdash_I t : \tau, _$ 则 $[\![t]\!] \in X_{\tau \circ _}$

证明。我们知道 $[\![\cdot]\!] = U \circ (|\cdot|)$，且 $(|t|)$ 是一个 Gl(C) 态射。因此，由定义可知 $[\![t]\!] = (|t|) = (|t|) \circ id_I \in X_\tau$，因为 $id_I \in X_I$。□

定理 5.3 可立即作为推论得出。

推论 A.9。 若 $\cdot \vdash_I t : \mathcal{M}\tau_1 \otimes \mathcal{M}\tau_2$，则 $[\![t]\!]$ 可分解为两个态射 $[\![t]\!] = f_1 \otimes f_2$，其中 $f_1 : I \to \mathcal{M}[\![\tau_1]\!]$ 且 $f_2 : I \to \mathcal{M}[\![\tau_2]\!]$。

证明。根据定理 A.8，如果 $\cdot \vdash_I t : \mathrm{M}\tau_1 \otimes \mathrm{M}\tau_2$，那么 $[\![t]\!] \in X_{\mathcal{M}\tau_1 \otimes \mathcal{M}\tau_2}$，根据展开定义，这意味着存在 $f_1 : I \to \mathrm{M}\,[\![\tau_1]\!]$ 以及 $f_2 : I \to \mathrm{M}\,[\![\tau_2]\!]$，使得 $[\![t]\!] = f_1 \otimes f_2$。□

B 可测集与马尔可夫核

可测空间将一个集合与其子集的集合结合起来，用于描述那些可以赋予明确定义的测度或概率的子集。

定义 B.1。 给定一个集合 X，一个 σ-代数 $\Sigma_X \subseteq \mathrm{P}(X)$ 是满足以下条件的子集集合：(i) $X \in \Sigma_X$，以及 (ii) Σ_X对补集运算和可数并运算封闭。可测空间 是一个二元组 (X, Σ_X)，其中 X是一个集合， Σ_X是一个 σ-代数。

可测空间 $(X,\ \Sigma_X)$ 和 $(Y\ \Sigma_Y)$ 之间的 可测函数 是一个函数 $f\colon X \to Y$ 满足对每个 $A \in \Sigma_Y$， $f^{-1}(A) \in \Sigma_X$，其中 f^{-1}是逆像函数。可测空间与可测函数构成一个范畴 **Meas**。

定义 B.2. 标准博雷尔空间 $(X,\ \Sigma_X)$ 是指满足以下条件的空间： X可赋予一个度量，使得 X作为度量空间是完备且可分的，并且 Σ_X是由该度量生成的 σ-代数。

例 B.3. 对于每个 $n \in \mathbb{N}$， $\mathbb{R}^n$ 连同其标准 σ-代数都是标准博雷尔空间。

定义 B.4. 概率测度是一个函数 $\mu_X\colon \Sigma_X \to [0,1]$，满足：(i) $\mu(\emptyset) = 0$，(ii) $\mu(X) = 1$，且 $\mu(\uplus A_i) = \sum_i \mu(A_i)$。

定义 B.5. 可测空间 (X, Σ_X) 和 $(Y\Sigma_Y)$ 之间的一个马尔可夫核 是一个函数$f\colon X \times \Sigma_Y \to [0,1]$，满足：

- 对每个 $x \in X$， $f(x, -)$ 都是一个概率分布。
- 对每个 $B \in \Sigma_Y$， $f(-, B)$ 都是一个可测函数。

马尔可夫核 $f\colon X \times \Sigma_Y \to [0,1]$和 $g\colon Y \times \Sigma_Z \to [0,1]$可以按以下公式复合

$$(g \circ f)(x, C) = \int g(-, C)\, df(x, -)$$

狄拉克核 $\delta(a,\ A) = 1$若 $a \in A$，否则为 0，是上述定义的复合的单位元；该结构可组织成一个范畴 **BorelStoch**，其中标准博雷尔空间为对象，马尔可夫核为态射。

边缘分布与概率独立性。我们需要构造一些关于积空间上的分布和测度的结果。

定义 B.6。给定一个在 $X \times Y$上的分布 μ，其边缘分布 μ_X是由 $\mu_X(A) = \int_Y d\mu(A,\ \text{–})$ 定义的、在 X 上的分布。直观地说，这是通过从 μ 中采样一对元素并投影到其第一个分量而得到的分布。另一个边缘分布 μ_Y 的定义类似。

定义 B.7。在 $A \times B$ 上的概率测度 μ 若为其边缘分布 μ_A 和 μ_B 的乘积，即 $\mu(X,\ Y) = \mu_A(X) \cdot \mu_B(Y)$，$X \in \Sigma_A$ 和 $Y \in \Sigma_B$，则称其概率独立。