arXiv:2603.22463v1 [cs.PL] 23 Mar 2026

用于通用概率编程的可并行化费曼–卡茨模型

MICHELE BOREALE 和 LUISA COLLODI

意大利佛罗伦萨大学。“G. Parenti”统计学、计算机科学与应用系。电子邮件地址：michele.boreale@unifi.it，luisa.collodi@unifi.it

摘要。我们研究概率程序（PPs）形式操作语义下序贯蒙特卡洛（SMC）推断的可证明正确且高效的实例化。我们重点关注具有从任意测度采样，以及在无界循环中进行条件化/重新加权特性的通用概率程序。首先，我们为概率程序图（PPGs）——一种概率程序的自动机理论描述格式——赋予基于无限执行轨迹的期望语义，该语义还纳入了轨迹权重。随后，我们证明一个有限逼近定理，该定理基于对有限固定长度轨迹求取的期望，为此语义提供界限。这使我们能够将该语义纳入费曼–卡茨（FK）模型框架，并确保粒子滤波（PF）算法（作为 SMC 的一种实例）相对于我们的语义具有一致性。在这些结果的基础上，我们提出 VPF，这是一种针对概率程序图及其语义定制的粒子滤波算法向量化版本。使用 VPF 概念验证实现开展的实验表明，与最先进的概率程序推断工具相比，结果非常有前景。

关键词：概率编程、操作语义、SIMD 并行、序贯蒙特卡洛。

1. 引言

概率编程语言（PPL） [34, 7] 为定义任意复杂的概率模型提供了一种系统化方法。人们通常希望基于观测数据对这些模型进行推断；例如，求出在给定观测数据的条件下程序输出的后验分布。在此，我们从概率程序形式操作语义的角度，研究序贯蒙特卡洛（SMC）推断的可证明正确且可并行化的实现。

在概率编程语言的形式语义方面，Kozen[39] 提出的指称方法提供了坚实的数学基础。然而，在概率编程语言推断的实用算法方面，现状似乎较为零散。一方面，符号和静态分析 技术（例如 [51, 31, 48, 8, 10, 50], ）能够得出有坚实语义依据且具有正确性保证的结果，但往往难以扩展。另一方面，实用语言和推断算法主要采用蒙特卡洛（MC）采样 技术（马尔可夫链蒙特卡洛、序贯蒙特卡洛），这类技术可扩展性更强，但往往缺乏

本研究部分由项目 SERICS（PE00000014）、欧盟资助的 NRRP MUR 计划——NextGenerationEU，以及 2023—2027 年卓越中心项目 ReDS——佛罗伦萨大学统计学、计算机科学与应用系资助。

© 用于通用概率编程的可并行化费曼–卡茨模型 知识共享

与形式语义的明确联系 [33, 27, 12]。这一现状也有一些值得注意的例外，例如 [49, 59, 42, 20, 41]，相关工作截面中对此进行了讨论。

相对于概率编程语言的形式语义，确立推断算法的一致性并非仅仅是理论上的追求。在通用概率编程语言的语境下， [32], 将无界循环和条件化与蒙特卡洛采样相结合，需要将计算截断于有限时间，这带来了重大挑战 [10]。此外，这些概率编程语言中的连续分布与离散分布相互作用，可能导致复杂情况，并有可能使现有的基于采样的算法产生错误结果 [59]。在本文中，我们确立了概率程序图（*PPG*）（一种对概率程序进行一般性自动机理论描述的格式）与*Feynman-Kac*（*FK*）模型（一种用于描述有限时间范围内定义的基于状态的概率过程和观测的形式体系）之间的精确联系 [28, Ch.5]。借助这一联系，我们证明了粒子滤波（*PF*）推断算法对于概率程序图的一致性；该算法是序贯蒙特卡洛方法的一种实现形式 [28, Ch.10]。在建立这一联系时，我们采取了如下所述的、明确的操作性视角。

在概率程序图（第3节）中，计算（本质上是采样）按照图的有向边（转移）所规定的连续阶段推进，节点则作为阶段之间的检查点，用于对观测数据进行条件化 ，或更一般地更新计算权重。概率程序图的操作语义以马尔可夫核和评分函数的形式加以形式化。在此基础上，我们引入测度论意义下的无限轨迹语义（第4节；所需的测度论知识在第2节回顾）。随后，有限逼近定理（第5节）使我们能够将这一轨迹语义与有限时间范围内的费曼–卡茨模型（第6节）精确联系起来。众所周知，粒子滤波对FK模型具有一致性：当模拟实例（粒子）的数量趋于无穷时，这些粒子的分布会收敛到由FK模型定义的测度 [28, 第11]章。因此，粒子滤波对概率程序图的一致性也将自动成立。

我们的方法还带来了其他见解。首先，有限逼近定理适用于定义在无限轨迹上的一类前缀闭函数：这类函数的输出仅取决于输入参数的有限初始片段。有限逼近定理意味着，定义在无限轨迹概率空间上的前缀闭函数的期望，可以通过定义在截断轨迹上的函数的期望来近似；后者是相对于某个合适的 FK 模型上定义的测度而言的。由于 FK 模型中的期望可以通过粒子滤波或其他算法有效估计，我们的有限逼近结果为概率程序的统计模型检测奠定了可靠的基础。其次，概率程序图的自动机理论操作语义可转化为粒子滤波的向量化实现，利用粒子层面存在的细粒度 SIMD 并行性。具体而言，在每一步中，概率程序图的转移函数和评分函数都会同时作用于整个由 N 个模拟粒子构成的向量。这在实践中意义重大，因为现代 CPU 和编程语言都广泛支持向量化，因而可能大幅提升速度。我们通过一个原型展示了这一点：该原型使用 TensorFlow[1], 实现了基于概率程序图的粒子滤波算法的向量化版本，称为 VPF。将 VPF 与最先进的概率编程语言在文献中的挑战性示例上进行比较的实验，展现出十分可喜的结果（第 7 节）。最后一节（第 8 节）给出了总结性评论。证明和其他技术材料收录于单独的附录中（A、B、C、D）。

主要贡献。总而言之，我们的主要贡献如下。

(1) 为概率程序图建立清晰的操作语义，该语义基于对无限轨迹取期望，并纳入条件化/重新加权。(2) 提出有限逼近定理，将该语义与有限轨迹及 FK 模型联系起来，从而确立粒子滤波对概率程序图的一致性。(3) 提出一种基于概率程序图的粒子滤波算法向量化版本，并通过实验验证其在实践中的可扩展性和竞争力。

相关工作。 除少数值得注意的例外外，大多数关于概率编程语言语义的研究都遵循 Kozen [39] 开创的指称方法；该领域的代表性研究见 [7, 11, 26, 34, 35, 36, 52, 53, 54, 56] 及其中的参考文献。在此背景下，与我们的目标正交的一项普遍目标是设计能够结合并推理密度的方法。请注意，我们并不要求概率程序在无限轨迹的概率空间上诱导出密度。

与我们的方法相关的是 Lunden 等人关于应用于概率编程语言的序贯蒙特卡罗推断的一系列研究。在 [42], 中，对于一种加入显式重采样原语的 λ 演算，研究表明，在某些限制条件下，无论代码中重采样的位置如何，粒子滤波的一致性都成立。从操作层面看，他们的函数式方法与我们的自动机理论方法截然不同。具体而言，他们通过隐式使用 续延 来处理重采样时粒子的挂起与恢复执行，这种做法沿用了 webPPL [33] 及其他概率编程语言的风格。函数式风格与续延相结合，并不天然适合向量化。例如，确保所有粒子都处于对齐状态，也就是说，其执行都处于重采样点，是一个可能对性能或准确性产生负面影响的问题。相反，在我们的自动机理论模型中，重采样的位置和对齐都不是问题：每次（向量化）转移步之后都会进行重采样，因此所有粒子都会自动对齐。请注意，在概率程序图中，一个转移可以将复杂的、无条件化的计算组合在一起；无论如何，粒子滤波的一致性都有保证。在后续工作 [44, 43], 中，Lunden 等人研究了序贯蒙特卡罗的具体实现问题。在 [44], 中，他们考虑了概率编程语言控制流图 （PCFG），这是一种旨在作为高级概率编程语言（例如他们的 CorePPL）编译目标的数据结构。PCFG 模型在理念上与概率程序图非常相似，但缺少形式语义。Lunden 等人还提供了这一框架的实现，旨在利用粒子层面上存在的潜在并行性。我们在第 7 节将我们的实现与他们的实现进行比较。

Aditya 等人证明了其概率编程语言 R2 [49], 中马尔可夫链蒙特卡罗（MCMC）的一致性。该语言基于一种大步采样语义，并将有限执行路径纳入考量。他们没有提供连接有限轨迹与无限轨迹（因而也包括无界循环）的近似结果。此外，大步语义是否能有效转化为 SIMD 并行算法也尚不明确。Wu 等人 [59]为概率编程语言 Blog 提供了严谨的测度论语义，该语义以贝叶斯网络的形式构建，并实现了一个专为此类网络设计的高效粒子滤波算法。同样，他们没有给出无界循环方面的结果。在我们先前的工作 [20], 中，我们针对含无界循环的概率编程语言，提出了测度论语义，并给出了有限近似结果，以及一种具有保证的 SIMD 并行实现；该实现本质上是一种拒绝采样 算法。拒绝采样对于有限形式的条件化或许有效；但随着条件化要求越来越高，它很快就会变得浪费且低效，例如在循环中反复进行条件化，或观测数据在模型中的似然较低时。最后，SMCP3 [41] 提供了一个丰富的测度论框架，可通过表达能力强的提议分布扩展实用的 Gen 语言 [29]。

概率编程语言领域的一个重要研究方向聚焦于符号化、精确的方法 [51, 31, 48, 8, 10, 50,37] ，旨在获得终止证书、概率程序终止概率的认证界，甚至是后验分布的精确表示；近期在这一方向上的一些研究也可参见 [6, 9, 55, 3, 40, 57]。正如前文所强调的，我们的目标与方法截然不同：我们关注通过采样实现可扩展推断，以及由此产生的一致性问题。这属于一项更广泛的研究计划，旨在开发灵活且可扩展的形式化方法，使其适用于各种与概率相关的领域，包括：涉及安全相关方面的动力系统 [13, 14, 15, 16, 17, 19], 、信息泄漏与安全[25, 24], 、涉及故障与恢复概念的分布式系统 [18], ，以及随机化模型计数与测试 [23, 45]。

本文是会议论文 [22]的扩展修订版。新增内容包括：补充示例（示例 6 和 8）、修订并扩展的实验验证（第 7 节、附录 C、D）、详细证明（附录 A），以及对标准粒子滤波算法更完整的描述（附录 B）。

2. 测度论预备知识

我们参照 [2], 前两章的讲述，回顾测度论中的一些基本概念；对于下文未明确说明的内容，可参考该书。给定一个非空集合 Ω，Ω 上的σ-代数 $\mathcal{F}$ 是 Ω 的子集构成的集合，其中包含 Ω，且对补集和可数不交并封闭。数对 $(\Omega, \mathcal{F})$ 称为可测空间。若对任意 $A \in \mathcal{F}_2$，都有 $f^{-1}(A) \in \mathcal{F}_1$，则（全）函数 $f : \Omega_1 \to \Omega_2$ 相对于 σ-代数 $(\Omega_1, \mathcal{F}_1)$ 和 $(\Omega_2, \mathcal{F}_2)$ 是可测的。我们令 $\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, +\infty\}$ 表示扩展实数集，并假定其运算遵循标准算术规则（参见 [2, 第 1.5 节2]）；令 $\mathbb{R}^+$ 表示包含 $+\infty$ 的非负实数集。$\Omega = \mathbb{R}^m$ 上的博雷尔σ-代数 $\mathcal{F}$ 是包含所有形如 $[a_1, b_1] \times \cdots \times [a_n, b_n]$ 的矩形的最小σ-代数，其中$a_i, b_i \in \mathbb{R}$。可测空间 $(\Omega, \mathcal{F})$ 的一个重要情形是 $\Omega = \mathbb{R}^m$，其中 Ω 取某个 $m \geq 1$，且 $\mathcal{F}$ 是 Ω 上的博雷尔σ-代数。全文中，“可测”均指“博雷尔可测”，这既适用于集合，也适用于函数。对于函数而言，博雷尔可测性在复合及其他基本函数运算下保持不变；连续实函数都是博雷尔可测的。若要明确空间的维数，我们用 $\mathcal{F}_k$ 表示 $\mathbb{R}^k$ $(k \geq 1)$ 上的博雷尔σ-代数。

可测空间 $(\Omega, \mathcal{F})$ 上的测度是一个函数 $\mu \colon \mathcal{F} \to \overline{\mathbb{R}}^+$，它具有可列可加性，即当 A_j’ 是 $\mathcal{F}$ 中两两不交的集合时，$\mu(\cup_{j\geq 1} A_j) = \sum_{j\geq 1} \mu(A_j)$。关于测度 μ [2,（二者均定义在测度空间 $(\Omega, \mathcal{F})$ 上）的博雷尔可测函数 f 的勒贝格积分记作 $\int_\Omega \mu(d\omega) f(\omega)$；在上下文明确时，省略下标 Ω。当 μ 是标准勒贝格测度时，可以省略 μ，将积分写作 $\int_\Omega d\omega f(\omega)$。对于 $A \in \mathcal{F}$，$\int_A \mu(d\omega) f(\omega)$ 表示 $\int_\Omega \mu(d\omega) f(\omega) 1_A(\omega)$，其中$1_A(\cdot)$ 是集合 A 的示性函数。我们用 δ_v 表示集中在 v上的狄拉克测度：对于适当 σ-代数中的每个集合 A，若 $v \in A$，则 $\delta_v(A) = 1$；否则，$\delta_v(A) = 0$。换言之，$\delta_v(A) = 1_A(v)$。另一种测度（与离散分布有关）是计数测度，$\mu_C(A) := |A|$。特别地，对于非负的 f，有等式 $\int_A \mu_C(d\omega) f(\omega) = \sum_{\omega \in A} f(\omega)$。概率测度是在 $\mathcal{F}$ 上定义且满足 $\int \mu(du) = 1$ 的测度 μ。对于定义在 Ω 上的给定非负可测函数 f，其关于概率测度 ν的期望就是它的积分：$\mathrm{E}_\nu[f] = \int \nu(d\omega) f(\omega)$。以下定义至关重要。

定义 *1*（马尔可夫核）。设 *(Ω1, $\mathcal{F}_1$)* 和 *(Ω2, $\mathcal{F}_2$)* 是可测空间。函数$K : \Omega_1 \times \mathcal{F}_2 \longrightarrow \mathbb{R}^+$ 是从 Ω_1 到 Ω_2 的马尔可夫核，当且仅当它满足以下性质：*(1)* 对每个 $\omega \in \Omega_1$, ，函数 $K(\omega, \cdot) : \mathcal{F}_2 \to \mathbb{R}^+$ 是 *(Ω2, $\mathcal{F}_2$)* 上的概率测度；*(2)* 对每个 $A \in \mathcal{F}_2$，函数 $K(\cdot, A) : \Omega_1 \to \mathbb{R}^+$ 是可测的。

在记号上，我们通常将 $K(\omega, A)$ 写作 $K(\omega)(A)$。以下是关于在乘积空间[1] $\Omega^t = \Omega \times \cdots \times \Omega$ (t 次) 上构造有限个测度乘积的一个标准结果，其中 $t \geq 1$ 是整数（更详细的表述见附录中的定理 A.1）。按照惯例，也将该定理定义的测度记为 $\mu^1 \otimes K_2 \otimes \cdots \otimes K_t$。

定理 1（测度乘积，[2],定理 2.6.7）。设 $t \geq 1$ 为整数。设 μ^1 是 Ω 上的概率测度，且 $K_2, ..., K_t$ 是从 Ω 到 Ω 的 $t-1$ 个（不一定互不相同的）马尔可夫核。那么，在 $(\Omega^t, \mathcal{F}^t)$ 上存在唯一的概率测度 μ^t，使得对于每个 $A_1 \times \cdots \times A_t \in \mathcal{F}^t$，都有：$\mu^t(A_1 \times \cdots \times A_t) = \int_{A_1} \mu^1(d\omega_1) \int_{A_2} K_2(\omega_1)(d\omega_2) \cdots \int_{A_t} K_t(\omega_{t-1})(d\omega_t)$。

3. 概率程序

我们首先介绍一种用于指定程序的一般形式体系，它采用某些可视为符号有限自动机的图。对于这一形式体系，我们随后将通过马尔可夫核引入操作语义。

概率程序图。定义概率程序时，我们将依赖一组基本分布：允许使用连续分布、离散分布和混合分布。表达能力的一个关键点在于，测度可以依赖于参数，而参数的值在运行时由程序状态决定。为了确保生成的程序定义可测函数（在适当的空间上），参数与测度之间的依赖关系本身也必须是可测的。我们将用马尔可夫核对这一点进行形式化。此外，我们还将考虑评分函数，它是取值为 0 或 1 的谓词的推广。形式上，我们将考虑下文定义的两个函数族。在这些定义中，我们用 $m \geq 1$ 表示一个固定整数，代表程序中变量 的数量，通常记作 $x_1, ..., x_m$。我们令 v 取值于$\mathbb{R}^m$，即给定状态下程序变量的内容，也称为存储。

- 参数测度：马尔可夫核 ζ： $\mathbb{R}^m \times \mathcal{F}_m \to [0, 1]$。
- 评分函数：可测函数 γ： $\mathbb{R}^m \to [0, 1]$。谓词是评分函数 φ： $\mathbb{R}^m \to \{0, 1\}$ 的一种特殊情况。常使用艾弗森括号记号，例如： $[x_1 \geq 1]$ 是一个谓词，其输入为 v 时，若 $v_1 \geq 1$，则输出 1，否则输出 0。

对于参数测度 ζ 和存储 $v \in \mathbb{R}^m$， $\zeta(v)$ 是一个分布，可用于根据当前程序存储 v采样得到新的存储 $v' \in \mathbb{R}^m$。从解析角度看，例如，可以通过依次对存储的各个组成部分进行采样来表示 ζ。这可以借助参数密度来实现：可测函数 ρ： $\mathbb{R}^m \times \mathbb{R} \to \mathbb{R}^+$，使得对于指定的测度 μ_ρ，函数 $(v, A) \mapsto \int_A \mu_\rho(dr)\, \rho(v, r)$ $(A \in \mathcal{F}_m)$ 是从 $\mathbb{R}^m$ 到 $\mathbb{R}$的马尔可夫核。下面通过一个例子来说明。

[1]我们将不加区分地把语言理论中的词视为元组。

例1。 固定 $m=2$。考虑如下定义的马尔可夫核，对每个 x_1、 $x_2\in\mathbb{R}$和 $A\in\mathcal{F}$

$$\zeta(x_1,x_2)(A):=\int\mu_1(dr_1)\left(\rho_1(x_1,x_2,r_1)\cdot\int\mu_2(dr_2)\rho_2(r_1,x_2,r_2)1_A(r_1,r_2)\right)\tag{3.1}$$

其中：
- $\mu_1=\mu_C$ 是计数测度；
- $\rho_1(x_1,x_2,r)=-\frac{1}{2}1_{\{x_1\}}(r)+\frac{1}{2}1_{\{x_2\}}(r)$ 是 $\{x_1,x_2\}$ 上离散分布的密度；
- $\mu_2=\mu_L$ 是普通的勒贝格测度；
- $\rho_2(x_1,x_2,r)=N(x_1,x_2,r)$：$=\frac{1}{|x_2|\sqrt{2\pi}}\exp(-\frac{1}{2}(\frac{r-x_1}{|x_2|})^2)$ 是均值为 x_1 、标准差为[2] $|x_2|$的正态分布的密度。

函数 ζ 是一个参数测度：具体而言，它对应于先从集合 $\{x_1,x_2\}$中均匀采样r_1，再从均值为 r_1 、标准差为$|x_2|$ 的正态分布中采样 r_2 （如果 $|x_2|$ 为正且有限；否则从默认分布中采样）。我们不通过(3.1)来描述 ζ，而是采用以下更方便的记号：

$$\begin{aligned}&r_1\sim\rho_1(x_1,x_2);\\&r_2\sim\rho_2(r_1,x_2);\\&\zeta(x_1,x_2)(A):=\int\mu_1(dr_1)\left(\rho_1(x_1,x_2,r_1)\cdot\int\mu_2(dr_2)\rho_2(r_1,x_2,r_2)1_A(r_1,r_2)\right)\end{aligned}$$

（或 *li* 列出）。请注意，从上到下的采样顺序很重要，　　*re.*

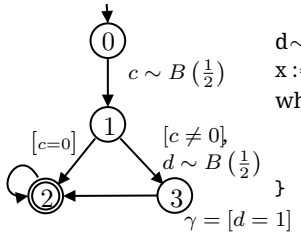
事实上，对于下文介绍的概率程序图（PPG）的形式框架而言，参数测度 ζ's 的解析描述方式并不重要。从实际角度来看，重要的是我们知道如何从测度 $\zeta(v)$ 中（高效地）采样，其中任意 v 均适用，这样才能实际实现推断算法（见第6节）。具体而言， $\zeta(v)$ 可能表示代码片段在输入 v下调用时，其返回的$\mathbb{R}^m$ 输出的分布（可能未知）。参数测度的另一种重要特殊情况如下。对于任意 $v=(v_1,...,v_m)\in\mathbb{R}^m$、 $r\in\mathbb{R}$和 $1\le i\le m$，令 $v[r@i]:=(v_1,...,r,...,v_m)$ 表示将 v_i 替换为 r后的元组。考虑参数测度 $\zeta(v)=\delta_{v[g(v)@i]}$，其中 $g:\mathbb{R}^m\to\mathbb{R}$ 是一个可测函数。用编程术语来说，这对应于将值 $g(v)$ 确定性地赋值 给变量 x_i。我们将这样描述该 ζ： $x_i:=g(x_1,...,x_m)$。

在下文概率程序图的定义中，可以将计算（采样）理解为沿着图的边（转移）在连续阶段进行，而节点则作为阶段之间的检查点 （这个术语借自 [42]），用于对观测数据进行条件化，或者更一般地说，用于重新加权分配给某项计算的评分。边还通过根据源节点存储中的内容计算出的谓词，来表示不同阶段之间的控制流。

定义2 （概率程序图）。 固定 $m\ge1$。 $\mathbb{R}^m$ 上的概率程序图（PPG）是一个四元组$\mathbf{G}=$（$\mathcal{P}$, E, nil,$\mathbf{sc}$），满足以下条件。
- $\mathcal{P}=\{S_1,...,S_k\}$ 是一个有限非空的程序检查点集（简称程序）。
- E 是一个有限非空的转移集，其形式为（S,φ,ζ,S'），其中： $S,S'\in\mathcal{P}$分别称为源程序检查点和目标函数程序检查点；φ： $\mathbb{R}^m\to\{0,1\}$ 是一个谓词； $\zeta:\mathbb{R}^m\times\mathcal{F}_m\to[0,1]$ 是一个参数测度。
- nil $\in\mathcal{P}$是一个特殊的已终止程序检查点，使得(nil, 1, id,nil) (id =恒等映射)是 E 中唯一以nil为源的转移。

[2]假设 p　　viso 是指，当 $x_2=0$或 $|x_1|$， $|x_2|=+\infty$, $N(x_1,\ x_2,\ r)$ 表示一个任意固定的默认概率密　　度。

```
c∼B (1/2);
if (c ==0) skip
else {
     d∼B (1/2);
     观测 (d==1);
     skip
}
```




```
d∼U (0 ,2);    r∼U (0,1);
x :=-1;    y :=1;
while ( |x-y | ≥.1)      {
     x∼N(x,d); y∼N(y,r);
     观测（ |x-y | ≤3）;
}
```

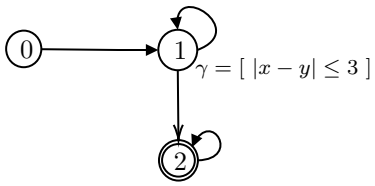


图1. **左侧**。例 2 的概率程序图以及相应的伪代码。空节点 (2) 以双边框标出。转移中不显示常数1谓词、评分函数和恒等函数。评分函数 γ 标注节点3，即 $\mathbf{sc}(3) = \gamma$。**右侧**。例 3 中醉汉与老鼠随机游走的概率程序图以及相应的伪代码。评分函数 γ 标注节点1，即 $\mathbf{sc}(1) = \gamma$。

- $\mathbf{sc}$ 是一个从 $\mathcal{P}$ 到评分函数集合的映射，使得$\mathbf{sc}(\mathsf{nil})$是常数*1*。此外，将 E_S 记为 E 中以 S 为源检查点的转移集合，假设满足以下一致性 条件：对于每个 $S \in \mathcal{P}$，函数$\sum_{(S,\varphi,\zeta,S')\in E_S} \varphi$ 是常数*1*。

我们先用一个简单的例子说明定义2。这也将用于说明下一节中的有限逼近定理。

例 2. 考虑图*1*左侧的概率程序图。这里我们有 $m = 2$ 和 $B(p)$ 是成功概率为 p 的伯努利分布。左侧给出了该程序结果的一种更为常见的伪代码表示法。不过，我们不会系统地探讨从这种程序表示法到概率程序图的形式化转换。

下面的示例说明如何在循环中使用评分函数。虽然这个例子有些刻意，但它与更重要场景的结构相近，例如飞机跟踪示例 [59], （参见第 7 节）。

例 3（老鼠与醉汉）。 在一条街道上，一名醉汉和一只老鼠从常规位置 -1 和1 出发，进行独立随机游走。最初，他们各自从均匀分布中采样一个标准差（*s.d.*）。随后，在每个离散时间步，他们根据以当前位置为中心、标准差为开始时所选数值的正态分布，独立采样各自的下一个位置。当醉汉和老鼠相遇时，过程立即停止；我们将相遇定义为两者之间的距离为 $< 1/10$。

有人提出，在某些城市地区，人和老鼠之间的距离从不会超过 *3* 米 [46]。我们将这一信息视为事实，并通过评分函数 $\gamma := [|x - y| \leq 3]$将其纳入模型。生成的概率程序图（*PPG*）见图*1*右侧（另请参见伪代码）。

注释 1。 评分函数的定义要求 γ 的取值属于 $[0, 1]$。这一限制是必要的，才能使定义在无限轨迹上的权重函数（参见 (4.2)）良定义。例如， [26] 中也有类似限制。实际上，只要 γ 非负且有界，我们总可以除以适当的常数[3]。例如，我们可以不考虑$\gamma(x_1, x_2) = N(x_1, 0.1, x_2)$，而考虑 $\gamma(x_1, x_2) = N(x_1, 0.1, x_2)/N(0, 0.1, 0)$。就轨迹语义而言（第*4*节），这种归一化实际上无关紧要，因为

[3]已提供 程序中出现的所有评分函数都除以相同的常数，以便避免失真 效应。

它会同时作用于分子和分母。另一方面，我们不能使用无界的评分函数，例如 $\gamma(x_1, x_2) = N(x_1, x_2, x_2)$。

概率程序图的操作语义。 对于任意给定的概率程序图 $\mathbf{G}$，我们将定义一个马尔可夫核 $\kappa_{\mathbf{G}}(\cdot,\cdot)$，用于描述其操作语义。从现在起，我们将考虑一个任意固定的概率程序图 $\mathbf{G} = (\mathcal{P}, E, \mathrm{nil}, \mathbf{sc})$，并在记号中省略下标 $\mathbf{G}$。另外需要指出的是，评分函数 $\mathbf{sc}(\cdot)$ 在马尔可夫核的定义中不起作用——它将在第4节的基于轨迹的语义中发挥作用。

下面介绍一些额外的记号简写。首先，我们将 $\mathcal{P}$ 视为有限自然数集 $\{0, ..., |\mathcal{P}| - 1\}$。根据这一约定，我们有 $\mathbb{R}^m \times \mathcal{P} \subseteq \mathbb{R}^{m+1}$。今后，我们将状态空间和σ-代数定义如下：

$$\Omega := \overline{\mathbb{R}}^{m+1} \qquad \mathcal{F} := \text{Borel sigma-field over } \overline{\mathbb{R}}^{m+1}.$$

对于任意 $k \geq 1$，我们保留符号 $\mathcal{F}_k$ 表示 $\mathbb{R}^k$ 上的博雷尔σ-代数。对于任意 $S \in \mathcal{P}$ 和 $A \in \mathcal{F}$，令 $A_S := \{v \in \mathbb{R}^m : (v, S) \in A\}$ 表示 A 在 S 处的**截面**。注意，$A_S \in \mathcal{F}_m$ 作为可测集的截面是可测的，参见 [2, Th.2.6.2, 证明(1)]。

定义3（概率程序图马尔可夫核）。函数 $\kappa : \Omega \times \mathcal{F} \to \mathbb{R}^+$ 定义如下，对于每个 $\omega \in \Omega$ 和 $A \in \mathcal{F}$：

$$\kappa(\omega)(A) := \begin{cases} \delta_\omega(A) & \text{if } \omega \notin \overline{\mathbb{R}}^m \times \mathcal{P} \\ \sum_{(S,\varphi,\zeta,S') \in E_S} \varphi(v) \cdot \zeta(v)\,(A_{S'}) & \text{if } \omega = (v, S) \in \overline{\mathbb{R}}^m \times \mathcal{P}. \end{cases} \tag{3.2}$$

引理 1。函数 κ 是从Ω到Ω的马尔可夫核。

4. 概率程序图的轨迹语义

下文中，我们固定一个任意的概率程序图 $\mathbf{G} = (\mathcal{P}, E, \mathrm{nil}, \mathrm{sc})$，并令 κ 表示由此导出的马尔可夫核，如定义3所述。对于任意 $t \geq 1$，我们将 Ω^t 称为长度为 t的路径集合。现在考虑无限长度路径的集合 Ω^∞，即无限序列$\tilde{\omega} = (\omega_1, \omega_2, ...)$的集合，其中 $\omega_i \in \Omega$。对于任意 $\omega^t \in \Omega^t$和 $\tilde{\omega} \in \Omega^\infty$，我们将对 $(\omega^t, \tilde{\omega})$ 与 Ω^∞ 中的元素等同，该元素的前缀 ω^t 后接 $\tilde{\omega}$。对于 $t \geq 1$ 和可测的 $B_t \subseteq \Omega^t$，令 $\mathfrak{c}(B_t) := B_t \cdot \Omega^\infty \subseteq \Omega^\infty$ 为由 B_t生成的可测柱集。令 $\mathcal{C}$ 为 Ω^∞ 上由所有可测柱集生成的最小σ-代数。在关于测度 μ^1 和核 $K_2, K_3, ...$ 的假设与定理 1 相同的条件下，存在唯一的测度 μ^∞ 定义在 $\mathcal{C}$ 上，使得对于每个 $t \geq 1$ 以及每个可测柱集 $\mathfrak{c}(B_t)$，都有 $\mu^\infty(\mathfrak{c}(B_t)) = \mu^t(B_t)$：参见 [2, Th.2.7.2], ，亦称 Ionescu-Tulcea定理。在下面的定义中，我们令 $0 = (0, ..., 0)$（重复 m 次），并考虑 $\delta_{(0,S)}$，即 Ω 上集中于 $(0, S)$ 的狄拉克测度，其承载全部概率质量。

定义 4 (由 S 诱导的概率测度**)**。设 $S \in \mathcal{P}$。对于每个整数 $t \geq 1$，我们令μ_S^t 为定义在Ω^t 上的概率测度，该测度由定理*1(a)*通过令 $\mu^1 = \delta_{(0,S)}$和 $K_2 = \cdots = K_t = \kappa$ 唯一确定。我们令 μ_S^∞为定义在 $\mathcal{C}$ 上的唯一概率测度，它由 μ_1和 $K_2 = \cdots = K_t = \cdots = \kappa$ 诱导，并由 *Ionescu-Tulcea*定理确定。

换句话说，$\mu_S^t = \delta_{(0,S)} \otimes \kappa \otimes \cdots \otimes \kappa$（$t-1$ 乘以 κ）。按照惯例，若 $t = 1$，则 $\mu_S^t = \delta_{(0,S)}$。测度 μ_S^∞可以非正式地解释为测度 μ_S^t的极限，并表示 S的语义。

回顾一下，（扩展）实值函数 f 的支撑集 是集合 $\mathrm{supp}(f) := \{z : f(z) \neq 0\}$。接下来，我们将专注于非负可测函数 f，以避免积分存在性带来的不必要复杂问题。一般函数可以通过通常的技巧处理：将 f分解为 $f = f^+ - f^-$，其中 $f^+ = \max(0, f)$，而 $f^- = -\min(0, f)$，然后分别处理 f^+和 f^-。下面引入组合评分函数 $\mathrm{sc}: \Omega \to [0,1]$，定义如下，对每个 $\omega = (v, S)$:

$$\mathrm{sc}(\omega) := \begin{cases} \mathbf{sc}(S)(v) & \text{if } \omega = (v,S) \in \overline{\mathbb{R}}^m \times \mathcal{P} \\ 1 & \text{otherwise.} \end{cases} \tag{4.1}$$

将函数 $\mathrm{sc}(\cdot)$ 扩展为定义在无限轨迹上的权重函数 $\mathrm{w}: \Omega^\infty \to [0,1]$，方法是令[4],对任意 $\tilde{\omega} = (\omega_1, \omega_2, \ldots\ldots) \in \Omega^\infty$:

$$\mathrm{w}(\tilde{\omega}) := \Pi_{j \geq 1} \mathrm{sc}(\omega_j)\,. \tag{4.2}$$

对于每个 $t \geq 1$，我们定义截断于时刻 t 的权重函数 $\mathrm{w}_t : \Omega^t \to [0,1]$，其定义为 $\mathrm{w}_t(\omega^t) := \Pi_{j=1}^t \mathrm{sc}(\omega_j)$。$\mathrm{w}$ 和 w_t $(t \geq 1)$ 分别是在各自定义域上的可测函数（见引理 A.2）。由此得到程序语义的定义。我们考虑未归一化语义 $([S]f)$ 与所有轨迹（无论是否终止）的权重 $([S]\mathrm{w})$ 之比。在评分函数表示条件化这一特殊情况下，这种选择相当于除以未失败轨迹的概率。在概率编程语言中，除以未失败状态的概率是较为常见的做法：例如参见 [38, 第 8.3 节.2]。

定义 5（轨迹语义）。设 f 是定义在 Ω^∞ 上的非负可测函数。我们将 S和 f 的未归一化语义记为 $[S]f := \mathrm{E}_{\mu_S^\infty}[f]$（$= \int \mu_S^\infty(d\tilde{\omega})\, f(\tilde{\omega})$）。我们记

$$[\![S]\!]f := \frac{[S](f \cdot \mathrm{w})}{[S]\mathrm{w}} \tag{4.3}$$

前提是上面的分母为 > 0；否则 $[\![S]\!]f$ 未定义。

5. 有限近似

由于可能存在无界循环，概率程序的操作语义是在无限轨迹上定义的。然而在实践中，我们只能对有限轨迹进行推理或采样。本节的主要结果定理 2 提供了一种严格的方法：通过计算有限前缀上的期望，来近似无限轨迹上的期望。直观地说，该定理指出，如果我们将所有轨迹截断至固定长度 t，并将注意力限制在截至该长度时已经终止的轨迹上，那么就能计算 f 期望的下界和上界。此外，如果保证程序会在有限时间内终止，这些界会随着 $t \to +\infty$ 收敛到真实值（定理 3）

更具体地说，我们感兴趣的是 $[\![S]\!]f$ 这样的情形：f 的值可以说由其自变量的一个有限前缀决定。我们将这类函数称为前缀闭合函数，并将在下文进一步定义。首先，我们需要介绍前缀闭语言[5],，为此需要对有限词和无限词的语言引入一些记号。给定两个词 w，$w' \in \Omega^*$，如果 w 是 w'的前缀，也就是说，存在一个词 $w'' \in \Omega^*$ 使得 $ww'' = w'$，我们记作 $w \prec w'$；否则记作 $w \not\prec w'$。对于 L，$L' \subseteq \Omega^*$，如果对所有 $w \in L$ 和 $w' \in L'$都有 $w \not\prec w'$，我们记作 $L \not\prec L'$。若语言序列 L_0，L_1,... 满足对每个 j，$L_j \subseteq \Omega^j$（其中

[4]注意，$\mathrm{w}(\tilde{\omega})$ 定义良好，因为对于每个 $j \geq 0$，$0 \leq \mathrm{sc}(\omega_j) \leq 1$，所以部分乘积序列是单调不增的。[5]在模型检验的语境中，这些语言是安全性性质的补集；例如参见 [5, Def.3.22]。

$\Omega^0 := \{\epsilon\}$，即空序列），则称其为前缀无关序列，条件是对每个 $i \neq j$，$L_i \not\prec L_j$。注意，如果 $L_0 \neq \emptyset$，那么对 $j \geq 1$有 $L_j = \emptyset$。为统一记号，以下约定 $\omega^0 := \epsilon$ 以及 $\mathfrak{c}(\{\epsilon\}) := \Omega^\infty$。若存在一个前缀无关语言序列 L_0，L_1,... 使得 $A = \cup_{j=0}^{\infty} \mathfrak{c}(L_j)$，我们就称 $A \subseteq \Omega^\infty$ 为前缀闭集；称 L_j 为A的一个 j-分支，并将 L_0，L_1,... 统称为 A的分支。对于任意 $t \geq 1$，我们定义Ω^t的以下子集：

$$L^{\leq t} := \cup_{j=0}^{t} L_j \cdot \Omega^{t-j}, \qquad L^{>t} := \{\omega^t : \text{ there is } t' > t \text{ and } \omega_{t'} \in L_{t'} \text{ s.t. } \omega^t \prec \omega^{t'}\}.$$

通俗地说，$L^{\leq t}$ 是长度为 t 且无论如何扩展为无限词都会成为 A成员的路径集合。$L^{>t}$ 是长度为 t 且其某些（但并非全部）无限扩展属于 A 的路径集合——也就是说，它们处于“未决”状态。下文定义的前缀无关语言序列尤其值得关注。

定义 6 (终止)。令 $\mathsf{T} := \mathbb{R}^m \times \{\mathsf{nil}\}$ 为已终止状态的集合，并令 T^{c}表示其补集。我们令 $T_j \subseteq \Omega^j$ $(j \geq 0)$ 表示在时刻终止 j的有限序列集合，即：$T_0 := \emptyset$ 且 $T_j := (\mathsf{T}^{\mathrm{c}})^{j-1} \cdot \mathsf{T}$，其中 $j \geq 1$。我们令 $T_{\mathrm{f}} := \cup_{t \geq 0} \mathfrak{c}(T_t) \subseteq \Omega^\infty$表示在有限时间内终止的无限序列集合。

注意，$\{T_j : j \geq 0\}$ 构成一个前缀无关序列，$T^{\leq t} \subseteq \Omega^t$ 是所有长度为 t 且在时间 t 内终止的路径的集合，而 $\mathfrak{c}(T_t) \subseteq \Omega^\infty$ 是终止时间为 t 的无限执行路径集合。接下来的定义引入前缀闭函数。这类函数 f 具有前缀无关的支撑集，满足条件 (a)，并且还满足另外两个条件。条件 (b)仅说明，f 在其支撑集上的取值由输入序列的某个有限前缀决定。条件 (c)，即 T- 尊重性，意味着在时间 j $(\omega^j \in T_j)$ 终止的轨迹不能在更晚的时间（$\omega^j \notin L^{>j}$）进入 $\mathrm{supp}(f)$。这是一个一致性条件，形式化地表明，f 的取值（可以说）不取决于终止之后 发生的事情。

定义 7（前缀闭函数）。设 $f : \Omega^\infty \to \mathbb{R}^+$ 是一个非负可测函数，且 $(L_0, L_1, \ldots)$ 是一个前缀闭序列。若满足以下条件，我们称 f 为一个具有分支 $L_0, L_1, \ldots$ 的前缀闭函数。

(a) $\mathrm{supp}(f)$ 是前缀无关的，其分支为 L_j $(j \geq 0)$。*(b)* 对于每个 $j \geq 0$ 和 $\omega^j \in L_j$，f 在 $\mathfrak{c}(\{\omega^j\})$上为常数。*(c)* $\mathrm{supp}(f)$ 是尊重 T 的：对于每个 $j \geq 0$，$L^{>j} \cap T_j = \emptyset$。

请注意，可能存在不同的前缀无关序列，使得 f 是前缀闭合的。

例 *4*。示性函数 $1_{T_{\mathrm{f}}}$ 显然是一个前缀闭合的可测函数，其 $\mathrm{supp}(1_{T_{\mathrm{f}}}) = T_{\mathrm{f}}$ 且分支为 $L_j = T_j$。若要了解更有趣的示例，请考虑例 *3* 中的概率程序图，以及函数 f_1（返回过程终止的时间）和 f_2（返回 d 在终止时的值）。这里 $\mathrm{supp}(f_1) = T_{\mathrm{f}}$ 的分支为 $L_j = T_j$ $(j \geq 0)$，而 $\mathrm{supp}(f_2) = \{\tilde{\omega} \in T_{\mathrm{f}} : \tilde{\omega}$中第一个终止状态 ω （如果存在）满足$\omega(1) = d \in (0, 2]\}$，且 $L_0 = \emptyset$, $L_j = (T^{\mathrm{c}})^{j-1} \cdot (T \cap ((0, 2] \times \mathbb{R}^4))$ $(j \geq 1$。

我们现在将研究如何用有限计算（μ_S^t 语义）一致地近似无限计算（μ_S^∞语义）。这将引出本节的主要结果（定理 2）。首先，我们为定义在无限乘积空间 Ω^∞ 上的函数f 引入一种恰当的有限近似概念。固定任意元素 $\star \in \Omega$。对于每个$f : \Omega^\infty \to \mathbb{R}^+$ 和 $t \geq 1$，定义函数 $f_t : \Omega^t \to \mathbb{R}^+$ 如下：$f_t(\omega^t) := f(\omega^t, \star^\infty)$。这里的直观理解是，对于前缀闭函数 f，函数 f_t 能够正确地近似 f

对于 L_j 的分支中所有有限路径， $j \le t$。例如，考虑例 4 中的函数 $f = f_1$。在 $L^{\le t}$ 上，近似 f_t 在某种精确意义上给出了相对于 f 的正确取值：无论 $\star$ 和 $\tilde{\omega}'$ 为何， $f_t(\omega^t) = f(\omega^t, \star^\infty) = f(\omega^t, \tilde{\omega}')$。另一方面，对于有限路径$\omega^t \in L^{>t}$， f_t可能无法正确近似 f：根据具体的 $\star$ 和 $\tilde{\omega}'$，可能会有 $f_t(\omega^t) = f(\omega^t, \star^\infty) \neq f(\omega^t, \tilde{\omega}')$。问题在于，随着 t 不断增大，集合 $L^{>t}$ 会变得越来越稀疏——至少在对测度 μ_S^∞ 作出合理假设的情况下如此。

不难验证，对于任意 t, f_t ，其在 Ω^t 上可测（见附录中的引理 A.2）。接下来的结果展示了如何用仅由 f_t、w_t 和 μ_S^t定义的量来近似 $[\![S]\!]f$ ，这构成了下一节基于采样的推理算法的基础。形式化地，对于 $t \ge 1$ 和可测函数 $h\colon \Omega^t \to \mathbb{R}^+$，我们令

$$[S]^t h \;:= \mathrm{E}_{\mu_S^t}[h] \; (= \textstyle\int \mu_S^t(d\omega^t) h(\omega^t)\,) .$$

定理 2 的证明（附录 A）将通过把以下引理应用于式 (5.2) 中相关表达式的分子和分母得出。直观上，考虑一个前缀闭函数 f ，其分支为 L_0、L_1.....。考虑 $[\![S]\!]f$ 的定义，即定义 5。对于任意时刻 t，不难看出 $\mathfrak{c}(L^{\le t} \cap T^{\le t}) \subseteq \mathrm{supp}(f) \subseteq \mathfrak{c}(L^{\le t} \cap T^{\le t}) \cup (\mathfrak{c}(T^{\le t}))^{\mathrm{c}}$（最后一个包含关系涉及 T-尊重性）。由于 f_t 在 $L^{\le t}$ 上对 f 进行了正确近似，可见第一个包含关系给出了下界$[S]^t f_t \cdot 1_{L^{\le t} \cap T^{\le t}} \cdot \mathrm{w}_t \le [S](f \cdot \mathrm{w})$。至于上界，直观上，在 $(\mathfrak{c}(T^{\le t}))^{\mathrm{c}}$ 上，f 由 M 给出上界。

引理 2。 设 $t \ge 1$ ，且设 $f \le M$ 为一个具有分支 L_j $(j \ge 0)$ 的前缀闭函数：

$$[S]^t f_t \cdot 1_{L^{\le t} \cap T^{\le t}} \cdot \mathrm{w}_t \;\le [S]f \cdot \mathrm{w} \le\; [S]^t f_t \cdot 1_{L^{\le t} \cap T^{\le t}} \cdot \mathrm{w}_t + M \cdot [S]^t (1 - 1_{T^{\le t}}) \cdot \mathrm{w}_t \,. \quad (5.1)$$

在定理上界的表述中，我们发现引入“校正因子” $\alpha_t \ge 1$很方便，它是时间 t 时所有轨迹的权重与已终止轨迹的权重之比。

定理 2 （有限近似）。 考虑 $S \in \mathcal{P}$ 和 $t \ge 1$ ，使得 $[S]^t 1_{T^{\le t}} \cdot \mathrm{w}_t > 0$。那么对于任意具有分支 L_0、L_1、.....的前缀闭函数 f ， $[\![S]\!]f$ 都有良好定义。此外，给定上界 $f \le M$ $(M \in \mathbb{R}^+)$，对于每个足够大的 t ，且$\alpha_t \colon = \frac{[S]^t \mathrm{w}_{[S]^t 1_{T^{\le t}} \cdot \mathrm{w}_t}}{t}$ ，我们有：

$$\frac{[S]^t f_t \cdot 1_{L^{\le t} \cap T^{\le t}} \cdot \mathrm{w}_t}{[S]^t \mathrm{w}_t} \;\le\; [\![S]\!]f \;\le\; \frac{[S]^t f_t \cdot 1_{L^{\le t} \cap T^{\le t}} \cdot \mathrm{w}_t}{[S]^t \mathrm{w}_t} \alpha_t + M \cdot (\alpha_t - 1) \,. \quad (5.2)$$

当 f 是示性函数 $f = 1_A$ 时，当然可以在上述定理中取 $M = 1$ 。我们先用一个简单的例子来说明上述结果。

例 5. 考虑例 2 中的概率程序图（图1，左）。我们想知道该程序终止时 c 的期望值是多少。形式上，我们考虑程序检查点$S = 0$，以及定义在轨迹上的函数 f：若存在终止状态，则返回第一个终止状态上 c 的值，否则返回 0。显然，这个 f 是前缀闭合的，其分支为 $L_j \subseteq T_j$。我们将定理 2 应用于 $[\![S]\!]f$。固定时间 $t = 4$ 后，我们可以轻松计算 (5.2) 中近似 $[\![S]\!]f$ 所涉及的量。为此，我们必须考虑长度为 t 、具有非零概率和权重的有限多条路径（只有两条），它们的权重，以及在终止时其最终状态上的 c 的值 [6]。

$$\begin{array}{ll} [S]^t f_t \cdot 1_{T^{\le t}} \cdot \mathrm{w}_t = 0 \cdot \frac{1}{2} + 1 \cdot \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4} & [S]^t \mathrm{w}_t = \frac{1}{2} + \frac{1}{2} \cdot \frac{1}{2} = \frac{3}{4} \\ [S]^t 1_{T^{\le t}} \cdot \mathrm{w}_t = \frac{1}{2} + \frac{1}{2} \cdot \frac{1}{2} = \frac{3}{4} & \alpha_t = 1 \,. \end{array}$$

[6]这里，我们也用到了这样一个事实： $f_t \cdot 1_{T^{\le t}} = f_t \cdot 1_{L^{\le t} \cap T^{\le t}}$，这是对所有 js 都成立的 $L_j \subseteq T_j$ 的推论。 .

于是，根据 $M=1$，(5.2) 中的上下界相等，并得出 $[\![S]\!]f=\frac{1}{3}$。如果我们移除对节点 *4* 的条件化，那么所有长度为 t 的路径的权重都为 *1*，类似的计算可得 $[\![S]\!]f=\frac{1}{2}$。

在更复杂的情况下，我们可能无法精确计算式 (5.2) 中涉及的量，只能通过采样来估计它们。为此，我们将在下一节介绍 Feynman-Kac 模型和粒子滤波算法。现在，我们先来看下面这个例子。

例 6. 考虑例 *3* 中的概率程序图以及例 *4* 中的 $f=f_2$。取 $S=0$。那么 $[\![S]\!]f$ 就是 d的后验期望，也就是醉汉的标准差。我们可以利用(5.2)计算 $[\![S]\!]f$ 的上界和下界。令 $t=60$固定。通过从 μ_S^t采样，我们可以分别计算(5.2)中各个期望值的以下估计值：$[S]^t f_t\cdot 1_{L\le t\cap T\le t}\cdot \mathrm{w}_t=0.396$、$[S]^t\mathrm{w}_t=0.987$ 和 $\alpha_t=1.497$。按照(5.2)将这些估计值与 $M=2$结合，可得界限：$0.402\le[\![S]\!]f\le 1.596$。考虑更大的 t值，即 α_t更接近 *1*，可以缩小这个相对较大的区间。第 *6* 节将介绍一种更高效、准确地计算(5.2) 中界限的方法：粒子滤波算法。

下面的定理证实，上述界是渐近紧的，至少在程序 $S\in\mathcal{P}$ 以概率 1 终止的假设下是如此。事实上，在这种情况下，$\mathsf{T}^{\le t}$ 之外的概率质量趋于 0，因此式 (5.2) 中的下界和上界趋于一致。此外，在保证程序会在固定时限内终止这一特殊情况下，我们可以得到一个更简单的公式；例如，对于无环[7] 概率程序图的情形。

定理 3（紧性）。假设与定理 *2* 相同的条件成立。进一步假设$\mu_S^\infty(T_\mathrm{f})=1$。那么，(5.2) 中的下界和上界都趋于 $[\![S]\!]f$，当 $t\to+\infty$时。特别地，若对某个 $t\ge 1$ 有 $[S]^t 1_{T\le t}=1$，则

$$[\![S]\!]f \;=\; \frac{[S]^t f_t\cdot \mathrm{w}_t}{[S]^t\mathrm{w}_t}. \tag{5.3}$$

例 7。对于例 *2* 中的概率程序图，有 $\mu_S^\infty(T_\mathrm{f})=1$。正如例 *2* 中所见，当 $t\ge 4$时，下界和上界一致。

一类在实际中很重要的闭前缀函数，其结果 $f(\tilde{\omega})$ 仅取决于对序列 $\tilde{\omega}$ 中第一个终止状态（如果存在）计算定义在 Ω 上的函数 h。这样一来，h 便被提升到 Ω^∞。迄今为止所见的所有示例都属于这种情况。我们将在下文正式引入提升。回顾一下，对于 $t\ge 1$, $T_t=(\mathsf{T}^\mathrm{c})^{t-1}\cdot\mathsf{T}$。

定义 8 (提升)。设 $h:\Omega\to\mathbb{R}^+$ 是一个非负可测函数，满足$\mathrm{supp}(h)\subseteq\mathsf{T}$。$h$ 的提升是可测函数 $\check{h}:\Omega^\infty\to\mathbb{R}^+$，对每个 $\tilde{\omega}=(\omega_1,\omega_2,\ldots)$，定义如下：$\check{h}(\tilde{\omega}):=\sum_{t\ge1}1_{\mathfrak{c}(T_t)}(\tilde{\omega})\cdot h(\omega_t)$。

显然，任意 $\check{h}$ 都是前缀闭合的，其分支为 $L_0=\emptyset$，并且 $L_j=(\mathsf{T}^\mathrm{c})^{j-1}\cdot\mathrm{supp}(h)\subseteq T_j$ 对 $\check{j}\ge1$ 成立。特别地，$\mathrm{supp}(h)\subseteq T_\mathrm{f}$。

$\check{}$**例 8.** 对于最终会终止的路径集合，其示性函数 $h=1_{\mathsf{T}_\mathrm{f}}$显然是 $h=1_\mathsf{T}$的提升。要看更有意思的示例，请参见例 *4*。函数

[7]或者更准确地说，是唯一的循环为空状态上的自环的概率程序图。

f_1 并不是Ω上任何函数的提升，因为它必须统计直到遇到第一个T为止经过了多少个状态。不过，可以修改例 *3* 中的程序，插入一个新的计数器变量 j 来提供这一信息，并在终止时返回 j 的值。函数 f_2 是函数 $h(d, r, x, y, z) = d \cdot [0 \leq d \leq 2] \cdot [z = \mathsf{nil}]$的提升。

6. Feynman-Kac 模型

在序贯蒙特卡罗方法领域，费曼–卡茨（FK）模型 [28, 第9]章 [28, 的特征是使用势函数。费曼–卡茨模型中的势函数，是一个为处于状态 x 、时间为 t的粒子（随机过程的实例）赋予权重 $G_t(x)$的函数。该权重表示基于某种观测量或条件化， x 在时间 t 的合理程度或适配程度。换言之， G_t 会随着系统演化改变粒子的重要性。例如，在一个用于跟踪物体的模型中，势函数可能取决于预测的粒子位置与实际观测位置之间的距离。距离观测位置越近的粒子会获得越高的权重。

FK模型与概率程序语义。如前所述，我们的语义通过在程序检查点应用评分函数来纳入条件化，并将其影响聚合为轨迹上的全局权重 w 。这使得我们能够将程序语义解释为重新加权的期望（定义 5）。在这里，我们将展示如何利用费曼–卡茨框架和粒子滤波器可靠地近似这一期望。我们首先在一般情境下介绍 FK模型。我们的表述紧密遵循 [28, 第9]章 [28, 。在本节及下一节中，我们令 $t \geq 1$ 为任意固定整数。

定义 9 (Feynman–Kac 模型)。 一个费曼–卡茨（FK）模型 是一个元组 $\mathrm{FK} = (\mathcal{X}, t, \mu^1, \{K_i\}_{i=2}^t, \{G_i\}_{i=1}^t)$，其中 $\mathcal{X} = \mathbb{R}^\ell$ 对于某个 $\ell \geq 1$, μ^1 是定义在 $\mathcal{X}$上的概率测度，并且对于 $i = 2, ..., t$: K_i 是从 $\mathcal{X}$ 到 $\mathcal{X}$ 的马尔可夫核，而 $G_i \colon \mathcal{X} \to \mathbb{R}^+$ 是可测函数。

根据定理 *1*，令 μ^t 表示由 $\mu_1, K_2, ..., K_t$ 诱导的定义在 $\mathcal{X}^t$ 上的唯一乘积测度。令 $G := \Pi_{i=1}^t G_i$。若 $0 < \mathrm{E}_{\mu^t}[G] < +\infty$，则由FK诱导的费曼–卡茨测度 定义如下，对每个可测 $A \subseteq \mathcal{X}^t$:

$$\phi_{\mathrm{FK}}(A) := \frac{\mathrm{E}_{\mu^t}[1_A \cdot G]}{\mathrm{E}_{\mu^t}[G]}. \tag{6.1}$$

我们将上述定义中的 G 称为全局势函数。等式 (6.1) 很容易推广到按照 ϕ_{FK} 计算的期望。也就是说，对于 $\mathcal{X}^t$ 上任意可测的非负函数 g ，我们可以很容易地 how that:

$$\mathrm{E}_{\phi_{\mathrm{FK}}}[g] = \frac{\mathrm{E}_{\mu^t}[g \cdot G]}{\mathrm{E}_{\mu^t}[G]}. \tag{6.2}$$

在下文中，如果不会引起混淆，我们将在记号中省略 ϕ_{FK} 中的下标 FK 。将 (6.2) 与定义 (4.3) 对比可知，全局势函数 G 在 FK 模型中所起的作用，类似于概率程序中的权重函数 w。不过请注意，二者存在一个重要的技术差异：FK 模型仅针对由 t 给出的有限时间范围模型定义。借助上一节中的有限逼近定理，可以调和二者之间的差异；下文将进一步阐述这一点（参见定理 4）。

我们尤其关注 ϕ 的第 t 阶边缘分布，即定义在 $\mathcal{X}$ 上的概率测度（$A \subseteq \mathcal{X}$ 可测）：

$$\phi_t(A) := \phi(\mathcal{X}^{t-1} \times A) = \mathrm{E}_\phi[1_{\mathcal{X}^{t-1}\times A}]\,. \tag{6.3}$$

测度 ϕ_t 称为滤波分布（时刻 t），可通过下一小节介绍的粒子滤波算法有效计算。现在，令$\mathbf{G} = (\mathcal{P}, E, \mathrm{nil}, \mathbf{sc})$为任意一个固定的概率程序图。将 (6.2) 与例如 (5.2) 中的下界进行比较，可以看出，考虑与$\mathbf{G}$ 及检查点 S相关联的以下费曼–卡茨模型是合适的。

定义 10 (费曼–卡茨$_S$ 模型)。令 $t \geq 1$ 为一个整数，S 为$\mathbf{G}$的一个程序检查点。我们将费曼–卡茨$_S$ 定义为如下费曼–卡茨模型：$\mathcal{X} = \Omega$、$\mu^1 = \delta_{(0,S)}$、$K_i = \kappa(i = 2, \ldots, t)$以及 $G_i = \mathrm{sc}(i = 1, \ldots, t)$。我们用 ϕ_S表示由费曼–卡茨$_S$诱导的、定义在Ω^t 上的测度。

现在我们将注意力限制在函数 f 上，这些函数是定义在 Ω 上的非负 h的提升。令 $\phi_{S,t}$ 表示通过 (6.3) 得到的 ϕ_S 在时刻 t 的滤波分布。在下述定理中，我们将 (5.2) 中的界用测度$\phi_{S,t}$表示。此结果的关键和意义在于，这些界直接以期望的形式表示；此外，这些期望是关于一维滤波分布 ($\phi_{S,t}$) 计算的，而非关于 t 维滤波分布 (μ_S^t) 计算的。重要的是，已有众所周知的算法可用于估计滤波分布下的期望，正如我们将在下一小节中看到的。

定理 4（滤波分布与提升函数）。在定理 *2* 的相同假设下，进一步假设 f 是 h的提升。则 $\alpha_t = \mathrm{E}_{\phi_{S,t}}[1_\intercal]^{-1}$ 并且

$$\beta_L := \mathrm{E}_{\phi_{S,t}}[h] \;\leq\; [\![S]\!] f \;\leq\; \mathrm{E}_{\phi_{S,t}}[h] \cdot \alpha_t \;+\; M \cdot (\alpha_t - 1) \;=: \beta_U\,. \tag{6.4}$$

例 9。再次考虑例 *2* 中的概率程序图。我们可以依赖定理 *4* 重新计算 $[\![S]\!] f$。固定 $t = 4$。首先，我们根据其定义 （6.3）计算 $\mathcal{X} = \Omega = \mathbb{R}^3$ 上的滤波分布 ϕ_t。与例 *5* 中的做法类似，我们考虑长度为四的非零权重、非零概率轨迹。然后投影到最终（第四个）状态，并计算所得三元组 (c, d, S) 的权重，再进行归一化。非零概率的三元组 (c, d, S) 只有两个：

$$\phi_t(0, 0, 2) = \frac{2}{3} \qquad\qquad \phi_t(1, 1, 2) = \frac{1}{3}\,.$$

例 *5* 中考虑的函数 f是函数 $h(c, d, S) = c \cdot [S = 2]$的提升，该函数定义在 $\mathcal{X} = \mathbb{R}^3$上。应用定理 *4*，得到 $\beta_L = \mathrm{E}_{\phi_t}[h] = \frac{1}{3} \leq [\![S]\!] f$。此外$\mathrm{E}_{\phi_t}[1_\intercal] = 1$，因此根据定理 *4*，$\alpha_t = 1$。因此 $\beta_L = \beta_U = [\![S]\!] f = \frac{1}{3}$。这与示例 *5* 和 *7* 一致。

我们可以将上述定理应用于例 4 中描述的函数以及其他计算难度较大的情况：在下一节介绍粒子滤波算法后，我们将在第 7 节中这样做。

粒子滤波算法。从计算角度来看，我们关注 FK 模型，是因为它们能够以简单、统一的方式呈现一类高效的推断算法，这类算法称为粒子滤波（*PF*） [28, 47, 58]。为便于介绍，这里我们仅介绍基本版本，即*Bootstrap* 粒子滤波，主要遵循[8] [28, 第11]章。给定一个一般的 FK 模型 $\mathrm{FK} = (\mathcal{X}, t, \mu^1, \{K_i\}_{i=2}^t, \{G_i\}_{i=1}^t)$。固定 $N \geq 1$，即要模拟的 $N \geq 1$ 个粒子，它们是由.....表示的随机过程的实例

[8]更多详细信息见附录 B。

算法 1 通用粒子滤波算法

输入：费曼–卡茨 $=(\mathcal{X}, t, \mu^1, \{K_k\}_{k=2}^t, \{G_k\}_{k=1}^t)$，一个费曼–卡茨模型；$N \geq 1$，粒子数。
输出：$X_t^{1:N} \in \mathcal{X}^N$, $W^{1:N}_t \in \mathbb{R}^{+N}$。

1: $X_1^{(j)} \sim \mu^1$ ▷状态初始化
2: $W_1^{(j)} := G_1(X_1^{(j)})$ $\Big\}$ $(j = 1, ..., N)$ ▷ 权重初始化
3: **对于** $k = 2, ..., t$ **do**
4: $r_{1:N} \sim R(W_{k-1}^{1:N})$ ▷ 重采样
5: $X_k^{(j)} \sim K_k(X^{(r_j)}_{k-1})$ ▷状态更新
6: $W_k^{(j)} := G_k(X_k^{(j)})$ $\Big\}$ $(j = 1, ..., N)$ ▷ 权重更新
7: **结束循环**
8: **返回** (X_t, W_t)

K_i。设 $W = W^{1:N} = (W^{(1)}, ..., W^{(N)})$ 为一组 N实数非负随机变量，即权重。记 W 为 W的归一化版本，即 $\widehat{W}^{(i)} = W^{(i)}/(\sum_{j=1}^N W^{(j)})$。对于 (N, W)，重采样方案是一个由随机变量 $R = (R_1, ..., R_N)$ 构成的 N 元组，其取值属于 $1..N$且依赖于 W，并且对每个 $1 \leq i \leq N$，都有：$\mathrm{E}[\sum_{j=1}^N 1_{R^{(j)}=i} \,|W] = N \cdot \widehat{W}^{(i)}$。换言之，每个索引 $i \in 1..N$平均会被选中 R 次，其次数与其在 W中的权重成正比。我们用$R(W)$表示 R依赖于给定权重向量 W。文献中提出了多种重采样方案，其中最简单的或许是多项式重采样；参见例如 [28, 第9] 章及其中的参考文献。算法 1 给出了一种通用的粒子滤波算法。这里的重采样在第 4 步进行：其目的是在提取下一代 N 粒子时，赋予权重较高的粒子更多的重要性，同时舍弃权重较低的粒子。

该算法的理论依据和实用价值在于，在温和的假设下，[28],对于定义在 $\mathcal{X}$上的任意可测函数 h，其在时刻 t关于 $\mathcal{X}$ 上的滤波分布 ϕ_t的期望，在极限情况下可以表示为权重为 $\widehat{W}_t^{(j)}$ 的加权和：

$$\sum_{j=1}^{N} \widehat{W}_t^{(j)} \cdot h(X_t^{(j)}) \longrightarrow \mathrm{E}_{\phi_t}[h] \quad \text{a.s. as } N \longrightarrow +\infty\,. \tag{6.5}$$

这里的实际意义是，我们可以相当有效地估计式（6.4）中涉及的期望，对于$\phi_t = \phi_{S,\,t}$，可将其表示为类似式（6.5）的加权和。请注意，在上述一致性结论中，t 保持固定——它是费曼–卡茨模型的一个参数——而粒子数 N 趋于 $+\infty$。

7. 实现与实验验证

7.1. 实现。 PPG 模型天然适合采用向量化方式实现粒子滤波，并利用粒子层面细粒度的 SIMD 并行性。在每次迭代中，N粒子 $\omega^N = (\omega_1, ..., \omega_N)$的状态，其中 $\omega_i = (v_i, z_i) \in \underline{\mathbb{R}}^{m+1}$，将分别存储在形状为 $N \times m$ 和 $N \times 1$ 的一对数组（V, Z）中。权重向量则存储在另一个形状为 $N \times 1$ 的数组 W 中。我们依赖操作的向量化：对于函数 $f\colon \mathbb{R}^k \to \mathbb{R}$ 和一个 $N \times k$数组 U，$f(U)$ 表示对 U 的每一行应用 f 后得到的 $N \times 1$数组。特别地，我们用 $(Z = s)$ 表示（对于

算法 2 VPF：用于概率程序图的向量化粒子滤波算法。

输入：$\mathbf{G} = (\mathcal{P}, E, \text{nil}, \mathbf{sc})$，一个概率程序图；$S \in \mathcal{P}$，初始概率检查点；$t \geq 1$，时间范围；$N \geq 1$，数量。个粒子。

输出：$V \in \mathbb{R}^{-m \times N}$，$Z, W \in \mathbb{R}^{1 \times N}$.

1: $V := S$；$Z := S$ ▷ 状态初始化
2: $W := \gamma_S(Z)$ ▷ 权重初始化
3: **执行** $t-1$ **次**
4: $(V, Z) :=$ 重采样$((V, Z)\ W)$ ▷ 重采样
5: **针对** $(s, \varphi, \zeta, s') \in E$ **do**
6: $M_{s\ \varphi} := \varphi(V) * (Z = s)$ ▷ 掩码计算
7: **结束循环**
8: $V \sim \sum_{(s,\varphi,\zeta,s') \in E} \zeta(V) * M_{s\ \varphi}$ ； $Z := \sum_{(s,\varphi,\zeta,s') \in E} s' \cdot M_{s\ \varphi}$ ▷状态更新
9: $W := \sum_{s \in \mathcal{P}} \gamma_s(V) * (Z = s)$ ▷ 权重更新
10: **结束循环**
11: **返回** (V, Z, W)

任意 $s \in \mathbb{N}$）逐元素将示性函数 $1_{\{s\}}$ 应用于 Z 后得到的 $N \times 1$ 数组；而 $\varphi(V)$ 表示对每一行应用谓词 φ 到 V 后得到的 $N \times 1$ 数组。对于 U一个 $N \times k$数组和 W一个 $N \times 1$ 数组，$U * W$表示将 U 的第 j 行与 W 的第 j 个元素相乘后得到的 $N \times k$数组，其中 $j = 1, ..., N$：当 W 是一个$0/1$ 向量时，这就是布尔掩码的一种形式。通过抽象现代 CPU 和编程语言中的向量化原语，我们将向数组变量赋予向量的操作建模为一条指令，记作 $U := Z$。适用于将标量广播到向量的常规规则同样适用，因此，例如 $V := S$ 对于 $S \in \mathbb{R}$ 意味着用 S 填充 V。同样，对于 ζ 一个参数分布，$U \sim \zeta(V)$ 表示从 $\zeta(v_1), ..., \zeta(v_N)$ 中独立采样 N 次，并将所得矩阵赋给 U：这也算作一条指令。

基于上述理想化的向量化计算模型，我们在算法 2 中给出了 VPF，即适用于概率程序图的粒子滤波算法的向量化版本。这里假设$\mathcal{P} \subseteq \mathbb{N}$，而 $\mathbf{sc}(s) = \gamma_s$。在第 4 行中，Resampling(·) 表示基于权重 W 对当前粒子的状态（由向量对 (V, Z) 表示）应用通用重采样算法所得的结果。对于通用粒子滤波算法 1，返回的输出中，(V, Z) 对应于 X_t，而 W 对应于 W_t。请注意，不存在迭代次数取决于 N的循环；第 5–7 行的 **for** 循环仅遍历转移集合 E，其大小与 N无关。第 8 行只是对从式 (6.1) 中的马尔可夫核函数进行采样的向量化实现。第 9 行是组合评分函数 (4.1) 的向量化实现。在实际的 TensorFlow 实现中，第 8 行和第 9 行中的求和通过布尔掩码和向量化操作来编码。

例 10. 考虑例 *3* 中的概率程序图和例 *4* 中的 $f = f_2$。取 $S = 0$。那么，$[\![S]\!]_f$ 就是 d（醉汉的标准差）取值的后验期望。我们按照定理 *4* 的定义，应用算法 *2*，计算 $[\![S]\!]_f$ 的上界和下界。令 $t = 60$ 和 $N = 10^4$ 个粒子固定。在滤波分布 $\phi_{S\text{下}, t}$，我们得到$E_{\phi_{S, t}}[h] = 0.768$, $\alpha_t = 1.029$。按照 (6.4) 将这些估计值合并，并取 $M = 2$，得到界 $0.768 \leq [\![S]\!]_f \leq 0.847$。$\alpha_t = 1.029 \approx 1$ 的值表明，在 $\phi_{S\text{下}, t}$，几乎所有粒子到时间 $t = 60$ 时都已终止。

7.2. **实验验证**。 我们展示了使用基于 TensorFlow 的概念验证版 [1] 算法 2 实现所获得的一些实验结果。我们仍将这一实现称为 VPF。我们考察了若干具有挑战性的概率程序，这些程序在循环内部包含条件化。对于所有这些程序，给定函数 f，我们将根据定理 4 提供的、以滤波分布下的期望表示的界来估计$[\![S]\!]f$。这些期望将通过 VPF 进行估计。我们还将 VPF 与两种最先进的概率编程语言 webPPL [33] 和 CorePPL [44] 进行比较。webPPL 是一种广受欢迎的概率编程语言，支持多种推断算法，包括序贯蒙特卡洛，其中重采样通过延续传递来处理。我们选择考察 Lunden 等人的 CorePPL，因为它支持高效实现粒子滤波：在 [44]，中，对 CorePPL 与 webPPL、Pyro [12] 及其他概率编程语言的性能比较表明，在多个基准测试中，基于序贯蒙特卡洛的 CorePPL 推断性能更胜一筹。CorePPL 的实现基于编译到一种中间格式，其概念与我们的概率程序图相似。

模型。在实验中，我们考虑了以下程序：飞机跟踪(AT，[59])、醉汉与老鼠 (DMM, 例 3)、兔子与乌龟 (HT，例如 [4])、有界重传协议 (BRP, [40])、非独立同分布循环 (NIID，例如 [40])、*ZeroConf*协议 (ZC，[9])，以及两种不同的随机游走变体：RW1 ([20], 例 2) 和下文中的 RW2。具体而言，AT 是一个在二维空间中利用来自六部雷达的噪声测量跟踪单架飞机的模型。HT 模拟野兔与乌龟在离散线上进行的比赛。BRP 对多个数据包经由有损信道传输的情形进行建模。NIID 描述了一个不断抛掷两枚公平硬币、直到两枚都显示反面的过程。ZC 是同名网络连接协议的理想化版本。RW1 和 RW2 是具有高斯步长的随机游走。这些模型的伪代码见附录 D。这些程序在循环内部使用条件化/评分。具体而言，DMM、HT 和 NIID 含有无界循环：对于这三个程序，在 VPF 的情况下，我们分别在 $k = 1000$, 100, 100 次迭代后截断执行，并据此设置定理 4 的时间参数 t，从而可以推导出 $[\![S]\!]f$ 值的界限。对于其他工具，我们只考虑在 k 次迭代结束时返回的截断估计值。AT、BRP、ZC、RW1 和 RW2 含有有界循环，但仍颇具挑战性。具体而言，AT 在 for 循环中包含多次条件化，需要从连续分布和离散分布的混合中采样，并且包含噪声观测。下文将从准确性和性能两方面讨论实验结果。附录 C 中介绍了这些程序，并提供了实验设置的更多细节；代码可从 [21] 获取。表 1 汇总了实验结果，我们将在下文进行讨论。

准确性。我们针对不同的 N（即粒子数；详情见表 1），在上述示例中比较了 VPF、CorePPL 和 webPPL。至少对于$N \geq 10^5$，这些工具通常会给出非常相似的期望值估计，我们将此视为准确性的经验证据。通过将 VPF 的结果与 webPPL-rejection（若有）的结果直接比较，可以进一步了解准确性；webPPL-rejection 是一种精确推断算法，其估计的期望值始终与 VPF 的估计值一致。我们还考察了有效样本量（ESS），这是衡量样本多样性的指标，数值越高越好 [30]。就 ESS 而言，各工具之间的差异显著：除一个例外（程序 RW1）外，VPF 得出的 ESS

		AT			DMM			HT			BRP			NIID		
		VPF	**CorePPL**	**webPPL-smc**	**VPF**	**CorePPL**	**webPPL-smc**	**VPF**	**CorePPL**	**webPPL-smc**	**VPF**	**CorePPL**	**webPPL-smc**	**VPF**	**CorePPL**	**webPPL-smc**
$N = 10^3$	*time*	**0.009**	0.014	0.190	2.348	**0.050**	0.474	0.274	**0.015**	0.152	0.676	**0.021**	0.155	0.240	**0.010**	0.061
	EV	6.805	6.955	6.696	0.478±0.110	0.501	0.427	32.834	33.683	32.368	0.018	0.016	0.023	3.594	2.694	3.473
	ESS	**1000**	**1000**	999	**994.6**	726.9	9.73	**955.0**	758.9	951.1	**1000**	**1000**	974.5	**1000**	846.6	726.9
$N = 10^4$	*time*	**0.131**	0.194	3.842	1.947	**0.988**	7.190	0.290	**0.180**	3.839	0.786	**0.309**	1.328	0.323	**0.058**	0.490
	EV	6.817	6.967	6.760	0.539±0.115	0.498	0.481	32.725	33.474	32.702	0.029	0.025	0.024	3.364	2.766	3.417
	ESS	$\mathbf{10^4}$	$\mathbf{10^4}$	*9975*	**9984.5**	7797.4	69.417	**9445.0**	7692.9	9476.2	$\mathbf{10^4}$	$\mathbf{10^4}$	9745.8	$\mathbf{10^4}$	8555.6	7560.5
$N = 10^5$	*time*	**0.354**	2.252	-	**2.268**	29.936	-	**0.379**	4.225	361.792	**0.797**	5.010	15.038	**0.445**	1.083	92.419
	EV	6.818	6.970	-	0.537±0.120	0.507	-	33.128	33.545	32.560	0.024	0.025	0.026	3.467	2.772	3.430
	ESS	$\mathbf{10^5}$	9.9e$\mathbf{10^5}$	-	$\approx \mathbf{10^5}$	7.6e10^4	-	$\approx \mathbf{10^5}$	7.7e10^4	$\approx \mathbf{10^5}$	$\mathbf{10^5}$	$\mathbf{10^5}$	9.7e10^4	$\mathbf{10^5}$	8.5e10^4	7.6e10^4
$N = 10^6$	*time*	**2.286**	26.481	-	**38.977**	-	-	**3.749**	49.493	-	**10.155**	58.448	-	**2.916**	14.323	-
	EV	6.834	6.980	-	0.541±0.111	-	-	33.432	33.606	-	0.024	0.025	-	3.413	2.774	-
	ESS	$\mathbf{10^6}$	9.9e10^5	-	$\approx \mathbf{10^6}$	-	-	$\approx \mathbf{10^6}$	7.7e10^5	-	$\mathbf{10^6}$	$\mathbf{10^6}$	-	$\mathbf{10^6}$	8.5e10^5	-
webPPL-rej	*EV*	-			0.494			32.683			0.023			3.414		

		RW1			ZC.1			ZC.2			RW2.1			RW2.2		
		VPF	**CorePPL**	**webPPL**	**VPF**	**CorePPL**	**webPPL**	**VPF**	**CorePPL**	**webPPL**	**VPF**	**CorePPL**	**webPPL**	**VPF**	**CorePPL**	**webPPL**
$N = 10^3$	*time*	0.192	**0.009**	0.045	0.206	**0.018**	0.083	0.262	**0.016**	0.049	0.231	**0.024**	0.232	0.232	**0.021**	0.187
	EV	0.323	0.324	0.343	0.212	0.142	0.250	0.514	0.477	0.483	1.046	0.642	0.912	0.677	0.750	1.092
	ESS	537.0	**1000**	46.739	**1000**	**1000**	392.2	**1000**	**1000**	245.2	**992.0**	780.0	479.9	**999.0**	997.0	133.9
$N = 10^4$	*time*	0.238	**0.031**	0.269	0.349	**0.068**	0.610	0.325	**0.029**	0.207	0.285	**0.186**	3.043	**0.271**	0.275	2.081
	EV	0.334	0.328	0.336	0.242	0.129	0.232	0.483	0.474	0.478	1.367	0.856	1.066	0.982	0.929	1.083
	ESS	5163.0	**10000**	562.795	**10000**	**10000**	4263.0	**10000**	**10000**	2446.8	**9967.0**	7529.0	4026.6	**9701.0**	9350.9	582.5
$N = 10^5$	*time*	0.436	**0.260**	7.956	**0.479**	0.558	5.778	0.378	**0.243**	1.673	**0.405**	3.003	181.802	**0.290**	3.887	149.831
	EV	0.337	0.328	0.332	0.174	0.131	0.233	0.493	0.479	0.479	1.009	0.998	0.982	1.040	0.982	1.037
	ESS	5.1e10^4	$\mathbf{10^5}$	5.7e10^3	$\mathbf{10^5}$	$\mathbf{10^5}$	4.2e10^4	$\mathbf{10^5}$	$\mathbf{10^5}$	2.4e10^4	$\approx \mathbf{10^5}$	7.3e10^4	4.7e10^4	$\approx \mathbf{10^5}$	9.5e10^4	2.0e10^3
$N = 10^6$	*time*	3.422	**2.569**	-	**3.829**	5.790	-	3.928	**2.485**	-	**3.742**	35.271	-	**3.595**	42.134	-
	EV	0.329	0.329	-	0.245	0.130	-	0.479	0.480	-	1.011	1.001	-	1.023	1.002	-
	ESS	5.2e10^5	$\mathbf{10^6}$	-	$\mathbf{10^6}$	$\mathbf{10^6}$	-	$\mathbf{10^6}$	$\mathbf{10^6}$	-	$\approx \mathbf{10^6}$	7.3e10^5	-	$\approx \mathbf{10^6}$	9.4e10^5	-
webPPL-rej	*EV*	0.332			0.235			0.479			1.022			1.061		

TABLE 1. Execution time (*time*) in seconds, estimated expected value (*EV*) and effective sample size ($ESS := (\sum_{i=1}^N W_i)^2/(\sum_{i=1}^N W_i^2)$; the higher the better, see e.g. [30]) as the number of particles (N) increases, for VPF, CorePPL and webPPL, when applied on Aircraft tracking (AT), Drunk man and mouse (DMM), Hare and tortoise (HT), Bounded retransmission protocol (BRP), Non-i.i.d. loops (NIID), ZeroConf (ZC.1, ZC.2) and Random Walks (RW1 and RW2.1, RW2.2). For VPF, with reference to Theorem 4: for the bounded loops AT, BRP, RW1, RW2.1, RW2.2, ZC.1 and ZC.2, we have $EV = \beta_L = \beta_U$ (as $\alpha_t = 1$); for HT and NIID, we only provide β_L, as β_U is vacuous ($M = +\infty$). For DMM we give the midpoint of the interval $[\beta_L, \beta_U] \pm$ its half-width. Best results for *time* and ESS for each example and value of N are marked in **boldface**. Everywhere, '−' means no result due to out-of-memory or timeout (500s). The results for DDM, especially for smaller values of N, exhibit a significant empirical variance: those reported in the table are obtained by averaging over 10 runs of each algorithm. Generally, there is an agreement across the tools about the estimates EV: an exception is NIID, where CorePPL returns values significantly different from the other tools' and from the exact value $\frac{24}{7} = 3.428\cdots$, cf. [40]. Also, for DMM the EV estimates returned by CorePPL and webPPL appear to be consistently lower than the midpoint of the interval returned by VPF.

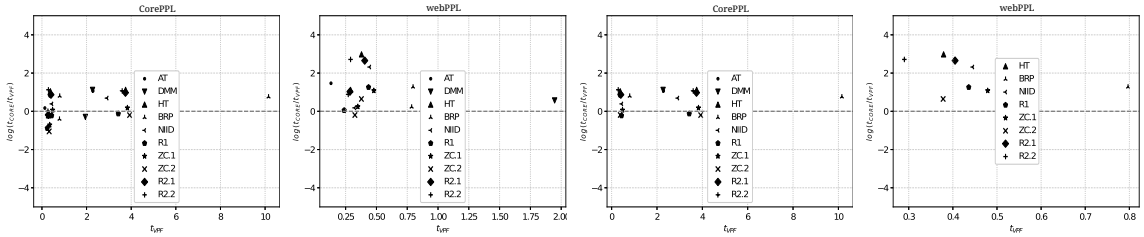


图 2. 对于 $N = 10^5$，10_6，基于表 1 中的数据点绘制的执行时间对数比值散点图，$\log_{10}(\text{time}_{tool}/\text{time}_{\text{VPF}})$。从左到右依次为： $N = 10^5$, $tool =$ CorePPL； $N = 10^5$, $tool =$ WebPPL-smc； $N = 10^6$, $tool =$ CorePPL； $N = 10^6$, $tool =$ WebPPL-smc。对于 $N = 10^6$，绝大多数数据点位于 x 轴上方，表明 VPF 在不同示例中的性能更优。这些图的放大版本见附录 C。

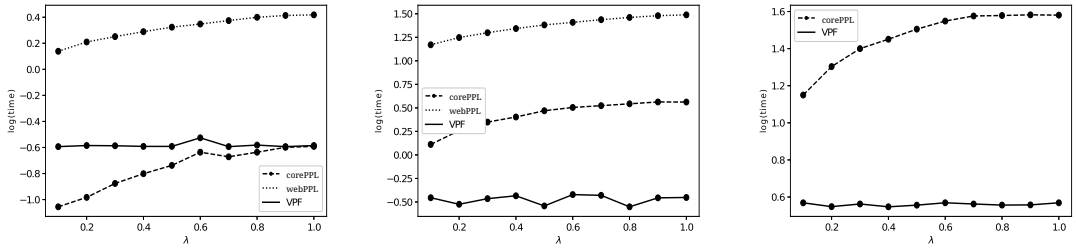
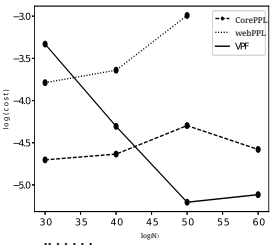


图 3. RW2 程序的执行时间（秒）随条件化外部数据的概率 λ 变化，分别对应 $N = 10^4$ （左）、 $N = 10^5$ （中）和 $N = 10^6$ （右）。由于超时，最右侧图中未包含 webPPL。VPF 的执行时间基本不受 λ 影响。

其执行时间高于或与其他工具相当。关于更多解释，尤其是所提及例外情况的意义，请参阅附录 C。

执行时间。当 N 的值较大时，就执行时间而言，VPF 通常优于其他受考察的工具。对于 $N = 10^6$，差异尤为明显。图 2 通过散点图对执行时间之比（$time_{\text{other-tool}}/time_{\text{VPF}}$）进行了图形化比较，并以对数刻度展示不同示例中的数据（实际数据点见表 1）。对于 WebPPL，几乎所有数据点都位于 x 轴上方，表明 VPF 更优。对于 CorePPL，在 $N = 10^5$ 的情况下，数据点在 x 轴上下基本均匀分布，表明两者总体不相上下。对于 $N = 10^6$，大多数数据点位于 x 轴上方，这再次表明 VPF 更优，有时优势可达数个数量级。

进一步考察 $N = 10^6$ 情形后发现，CorePPL 表现优于 VPF 的程序只有 RW1 和 ZC。这很可能是因为这些程序中进行条件化的概率较低；例如，在 RW1 中只进行了一次最终条件化。由于 CorePPL 仅在条件化之后进行重采样，这或许可以解释它在这些情形下执行时间较短的原因。为进一步研究这一问题，我们考虑 RW2 的一个变体，其中条件化的概率由参数 $\lambda \in [0, 1]$ 控制，并针对 λ 的不同取值运行该程序。所得结果如图 3 所示。我们观察到，对于 CorePPL 和 WebPPL，执行时间往往会随着条件化概率 λ 的增加而增加；相反，VPF 的执行时间似乎不受 λ 影响。这表明，对于条件化频繁的模型，VPF 相较于使用显式重采样的工具具有明显优势。下一段将从可扩展性的角度进一步分析 CorePPL 与 VPF 之间的差异。



可扩展性。 右侧的图以对数刻度展示了 VPF、CorePPL 和 WebPPL 在我们针对 $N = 10^3, ..., 10^6$ 所分析的所有模型上的平均单位成本（每个粒子）。此处，对于每个 N，跨 k 个模型的平均单位成本（以秒计）为 $(t_1 + t_2 + .. + t_k)/(N \cdot k)$，其中 t_i 表示第 i 个模型的执行时间。与图 3 的结果一致，我们观察到，随着样本数量增加，VPF 的成本会降低，而其他工具的成本则保持不变或有所上升（webPPL）。

8. 结论

我们提出了一个用于通用概率程序推断的框架，将严格的基于轨迹的操作语义与实用的算法可实现性相结合。该方法建立在结构化操作模型之上，并与随机过程理论中的经典工具（FK模型）相联系，从而形成了一种理论上可靠且能够高效实现的推断方法。下面总结我们方法的主要见解和可能的未来方向。

通过概率程序图实现结构化语义（第 3 节和第 4 节）。我们的框架以一种面向程序的自动机理论描述格式——概率程序图（PPG）——为核心。在概率程序图中，转移编码采样行为，而节点则通过评分函数表示条件化检查点。这种结构支持严格的无限轨迹语义，并有助于实现计算对齐；这一特性对于向量化实现至关重要。

通过费曼–卡茨模型实现近似（第 5 节和第 6 节）。主要理论贡献是建立了概率程序图的无限轨迹语义与费曼–卡茨（FK）模型之间的新联系；后者是分析基于状态的随机过程的标准工具。一大类前缀闭合函数在无限轨迹上的期望值，可以用有限的截断计算来表示。特别是，定理 2 证明了，这些期望可以由定义在有限长度轨迹上的量进行界定，从而使得在存在无界循环和条件化的情况下仍能进行可处理的推断。定理 3 表明，在温和的假设下，这些界会收敛到精确值。随后，我们建立了与 FK 模型的联系：具体而言，定理 4 将近似界改写为由概率程序图诱导的 FK 模型的滤波分布；这些分布可以通过粒子滤波（PF）进行一致且高效的估计。

高效并行推断（第 7 节）。我们方法的一个核心见解是，PPG 的操作结构使推断能够以自然的方式实现并行化。由于所有粒子都通过同一控制流图（PPG）同步演化（步调一致），且通过评分函数以均匀、对齐的方式进行条件化，我们的粒子滤波算法可直接映射为 SIMD 式向量化执行。这种设计避免了基于延续的语义或 PPL 的函数式语义所带来的粒子错位问题。因此，我们的操作模型很容易映射到支持数据级并行的现代硬件架构上。我们对基于 PPG 的粒子滤波器 VPF 所做的向量化实现，实际展示了该方法的有效性。在涉及嵌套条件化和无界循环的挑战性示例中，VPF 的表现不逊于

甚至优于最先进的概率编程系统，在准确性和运行时间两方面均是如此。

未来研究方向。在实践方面，开发从高级概率编程语言到概率程序图的编译器，并将这一框架扩展到更丰富的类型系统和数据结构，是自然的下一步。将基于采样的方法与符号推理或基于约束的推理技术相结合，是一个颇具挑战性的理论方向。

参考文献

[1] Martín Abadi、Paul Barham、Jianmin Chen、Zhifeng Chen、Andy Davis、Jeffrey Dean、Matthieu Devin、Sanjay Ghemawat、Geoffrey Irving、Michael Isard、Josh Levenberg Manjunath Kudlur、Rajat Monga、Sherry Moore、Derek G. Murray、Benoit Steiner、Paul Tucker、Vijay Vasudevan、Pete Warden、Martin Wicke、Yuan Yu 和 Xiaoqiang Zheng。Tensorflow：一种用于大规模机器学习的系统。第 *12* 届 *USENIX* 操作系统设计与实现研讨会论文集（*OSDI '16*）。*arXiv:1605.08695*，2016。[2] Robert B. Ash。实分析与概率。*Academic Press, Inc., New York, NY, USA*，1972。[3] Martin Avanzini、Georg Moser 和 Michael Schaper。递归程序的自动化期望值分析。*roc. ACM* 程序语言，*7*，*PLDI*，2023。[4] Alexander Bagnall、Gordon Stewart 和 Anindya Banerjee。对含循环和条件化的概率程序进行形式化验证的采样器。*ACM* 程序语言论文集，*7*，*1–24*，2023。[5] Christel Baier 和 Joost-Pieter Katoen。模型检验原理。麻省理工学院出版社，2008。[6] Jialu Bao、Nitesh Trivedi、Drashti Pathak、Justin Hsu 和 Subhajit Roy。概率程序的数据驱动不变式学习。载于：*CAV*。《计算机科学讲义》，第 *13371* 卷，第 *33–54* 页。施普林格，2022。[7] Gilles Barthe、Joost-Pieter Katoen 和 Alexandra Silva。概率编程基础。剑桥大学出版社，2020。[8] Ezio Bartocci、Laura Kovács 和 Miroslav Stankovic。Mora——基于矩的不变式自动生成。系统构造与分析工具和算法国际会议。施普林格，*492–498*，2020。[9] Kevin Batz、Mingshuai Chen、Sebastian Junges、Benjamin Lucien Kaminski、Joost-Pieter Katoen 和 Christoph Matheja。通过归纳综合归纳不变式验证概率程序。载于 *Sankaranarayanan, S.*、*Sharygina, N.*（编）《系统构造与分析工具和算法》。*TACAS 2023*。《计算机科学讲义》，第 *13994* 卷。施普林格，2023。[10] Raven Beutner、Luke Ong 和 Fabian Zaiser。通用概率编程中后验推断的保证界限。载于第 *43* 届 *ACM SIGPLAN* 编程语言设计与实现国际会议论文集（*PLDI 22*），2022。[11] Sooraj Bhat、Johannes Borgström、Andrew D. Gordon 和 Claudio Russo。从概率函数式程序推导概率密度函数。载于系统构造与分析工具和算法国际会议论文集（*TACAS 2013*），2013。[12] Eli Bingham、Jonathan P Chen、Martin Jankowiak、Fritz Obermeyer、Neeraj Pradhan、Theofanis Karaletsos、Rohit Singh、Paul Szerlip、Paul Horsfall 和 Noah D Goodman。Pyro：深度通用概率编程。机器学习研究杂志 *20*，*1*，*973–978*，2019。[13] Michele Boreale。多项式连续系统的精确与近似线性抽象算法。载于第 *21* 届混合系统：计算与控制国际会议（*HSCC*）论文集，第 207–216 页。ACM，2018。[14] Michele Boreale。多项式常微分方程中代数最强后置条件和最弱前置条件的完备算法。载于*SOFSEM 2018*：计算机科学理论与实践，第 10706 卷《计算机科学讲义》，第 442–455 页。施普林格，2018。[15] Michele Boreale。多项式微分方程中的代数、余代数与最小化。《计算机科学逻辑方法》，15(1)，2019。[16] Michele Boreale。关于偏微分方程的余代数。载于第 *44* 届计算机科学数学基础国际研讨会（*MFCS*），第 138 卷*LIPIcs*，第 24:1–24:13 页。达格施图尔城堡–莱布尼茨信息学中心，2019。

[17] Michele Boreale。偏微分方程的自动前置条件与后置条件。Information and Computation，285:104860，2022。[18] Michele Boreale、Roberto Bruni、Rocco De Nicola 和 Michele Loreti。Caspis：会话、流水线与服务演算。Mathematical Structures in Computer Science，25(3):666–709，2015。[19] Michele Boreale 和 Luisa Collodi。通过区间分析与模拟实现有保证的贝叶斯参数估计。载于 VMCAI 2023：验证、模型检测与抽象解释，第 13881 卷《计算机科学讲义》，第 106–128 页。施普林格，2023。[20] Michele Boreale 和 Luisa Collodi。概率程序的有保证推断：一种可并行化的小步操作方法。《计算机科学讲义》，第 14500 卷，施普林格，2023。[21] Michele Boreale 和 Luisa Collodi。本文所述实验的代码。https: //github.com/Luisa-unifi/Vectorized-Particle-Filtering。2025。[22] Michele Boreale 和 Luisa Collodi。面向通用概率编程的可并行化费曼–卡茨模型。载于 Giorgio Bacci 和 Adrian Francalanza（编），第十六届博弈、自动机、逻辑与形式化验证国际研讨会论文集，GandALF 2025，马耳他瓦莱塔，2025年9月16–17日，第 428 卷电子理论计算机科学出版物，第 91–110 页，2025年9月。[23] Michele Boreale 和 Daniele Gorla。近似模型计数、稀疏异或约束与最小距离。载于《计算系统建模的艺术：从逻辑与并发到安全与隐私的探索》，第 11760 卷《计算机科学讲义》，第 363–378 页。施普林格，2019。[24] Michele Boreale 和 Francesca Pampaloni。通用泄漏函数和自适应对手下的定量信息流。载于 FORTE 2014：分布式对象、组件与系统的形式化技术，第 8461 卷《计算机科学讲义》，第 166–181 页。施普林格，2014。[25] Michele Boreale 和 Michela Paolini。通过统计估计对信息泄漏进行形式化界定。载于 ISC 2014：信息安全会议，第 8783 卷《计算机科学讲义》，第 216–236 页。施普林格，2014。[26] Johannes Borgström、Ugo Dal Lago、Andrew D. Gordon 和 Marcin Szymczak。通用概率编程的 λ 演算基础。ACM SIGPLAN Notices 51.9：33–46，2016。[27] Bob Carpenter、Andrew Gelman、Matthew D. Hoffman、Daniel Lee、Ben Goodrich、Michael Betancourt、Marcus Brubaker、Jiqiang Guo、Peter Li 和 Allen Riddell。Stan：一种概率编程语言。Journal of Statistical Software 76(1)，2017。[28] Nicolas Chopin 和 Omiros Papaspiliopoulos。序贯蒙特卡洛导论。Springer Nature Switzerland AG，2021。[29] Marco F. Cusumano-Towner、Feras A. Saad、Alexander K. Lew 和 Vikash K. Mansinghka。Gen：一种具有可编程推断功能的通用概率编程系统。载于第 40 届 ACM SIGPLAN 编程语言设计与实现会议论文集（PLDI），第 221–236 页。美国纽约，2019。[30] Víctor Elvira、Luca Martino 和 Christian P. Robert。重新审视有效样本量。arXiv：1809. 04129，2018。[31] Timon Gehr、Sasa Misailovic 和 Martin T. Vechev。Psi：概率程序的精确符号推断。第 28 届计算机辅助验证国际会议论文集（CAV 2016），多伦多，第 62–83 页，2016。[32] Noah D. Goodman、Vikash Mansinghka、Daniel Roy、Keith Bonawitz 和 Joshua Tenenbaum。Church：生成模型语言。人工智能中的不确定性会议论文集，2008。[33] Noah D. Goodman 和 Andreas Stuhlmüller。概率编程语言的设计与实现。第 28 届计算机辅助验证国际会议论文集（CAV 2016），多伦多，第 62–83 页，2016。[34] Andrew D. Gordon、Thomas A. Henzinger、Aditya V. Nori 和 Sriram K. Rajamani。概率编程。《软件工程的未来》会议论文集（FOSE2014 181，2014 167），第 – 页。[35] Maria I. Gorinova、Andrew D. Gordon 和 Charles Sutton。SlicStan 中具有密度的概率编程：高效、灵活且确定。《ACM 编程语言论文集》3，POPL.1-30，2019。[36] Maria I. Gorinova、Andrew D. Gordon、Charles Sutton 和 Matthijs Vákár。通过类型系统表达条件独立性。ACM编程语言与系统汇刊（TOPLAS）44.1：1–54，2021。[37] Michikazu Hirata、Yasuhiko Minamide 和 Tetsuya Sato。Isabelle/HOL 中高阶概率程序的语义基础。第 14 届交互式定理证明国际会议（ITP），

2023。《莱布尼茨国际信息学论文集》（LIPIcs），第268卷，第18:1–18:18页，Schloss Dagstuhl – Leibniz-Zentrum für Informatik，2023。[38] Benjamin L. Kaminski。《概率程序的高级最弱前置条件演算》。博士论文（Ph.D），亚琛工业大学，2019。[39] Dexter Katoen。概率程序的语义。《计算机与系统科学杂志》，22(3):328–350，1981。[40] Lutz Klinkenberg、Christian Blumenthal、Mingshuai Chen、Darion Haase 和 Joost-Pieter Katoen。使用生成函数对含环概率程序进行精确贝叶斯推断。载于《ACM 编程语言论文集》（OPSLA），第8卷：127,923–953，2024。[41] Alexander K. Lew、George Matheos、Tan Zhi-Xuan、Matin Ghavamizadeh、Nishad Gothoskar、Stuart Russell 和 Vikash K. Mansinghka。SMCP3：采用概率程序提议的序贯蒙特卡洛。第26届国际人工智能与统计学会议论文集（AISTATS）。PMLR 206:7061–7088，2023。[42] Daniel Lundén、Johannes Borgström 和 David Broman。概率编程语言序贯蒙特卡洛推断的正确性。《编程语言与系统》，第404–431页。施普林格国际出版公司，2021。[43] Daniel Lundén、David Broman、Fredrik Ronquist 和 Lawrence M. Murray。高阶概率程序中序贯蒙特卡洛推断的自动对齐。载于 ESOP，第535–563页，2023。[44] Daniel Lundén、Joey Ohman、Jan Kudlicka、Viktor Senderov、Fredrik Ronquist 和 David Broman。编译通用概率编程语言，实现高效的并行序贯蒙特卡洛推断。载于 ESOP，第29–56页，2022。[45] Héctor D. Menéndez、Michele Boreale、Daniele Gorla 和 David Clark。通过输出采样提升自动单元测试生成的输出多样性。《IEEE 软件工程汇刊》，48 308，2022 2 295( ): – 。[46] Tshepo Mokoena。都市传说：在城市里，离老鼠从来不会超过6英尺远吗?《卫报》在线版，https: // www. theguardian. com/ cities/ 2014/ feb/ 13/ urban-myths-6ft-from-a-rat ., 2014。[47] Pierre Del Moral、Arnaud Doucet 和 Ajay Jasra。序贯蒙特卡洛采样器。《英国皇家统计学会杂志》。B辑（统计方法学），第68卷，第3期，第411–436页，2006。[48] Praveen Narayanan、Jacques Carette、Chungchieh Shan Wren Romano 和 Robert Zinkov。Hakaru中的程序变换概率推断（系统描述）。载于第13届国际函数式与逻辑编程研讨会（FLOPS 2016）论文集，第62–79页，2016。[49] Aditya V. Nori、Chung-Kil Hur、Sriram K. Rajamani 和 Selva Samuel。R2：一种用于概率程序的高效马尔可夫链蒙特卡洛采样器。第二十八届 AAAI 人工智能会议论文集，第2476–2482页，2014。[50] Francesca Randone、Luca Bortolussi、Emilio Incerto 和 Mirco Tribastone。采用矩匹配高斯混合的概率程序推断。《ACM 编程语言论文集》（POPL）：1882–1912，2024。[51] Sriram Sankaranarayanan、Aleksandar Chakarov 和 Sumit Gulwani。概率程序的静态分析：从有限条路径推断全程序性质。第34届 ACM SIGPLAN 编程语言设计与实现会议论文集，2013。[52] Adam Scibior、Ohad Kammar、Matthijs Vákár、Sam Staton、Hongseok Yang、Yufei Cai、Klaus Ostermann、Sean K. Moss、Chris Heunen 和 Zoubin Ghahra mani。高阶贝叶斯推断的指称语义验证。《ACM 编程语言论文集》2。POPL。1–29，2017。[53] Sam Staton。概率编程的交换语义。第26届欧洲编程研讨会（ESOP2017）论文集，瑞典乌普萨拉，2017。[54] Sam Staton、Frank Wood、Hongseok Yang、Chris Heunen 和 Ohad Kammar。概率编程的语义：高阶函数、连续分布与软约束。第31届 ACM/IEEE 计算机科学逻辑年会（LICS）论文集，2016。[55] Zachary Susag、Sumit Lahiri、Justin Hsu 和 Subhajit Roy。随机化程序的符号执行。ACM 程序语言论文集，6，OOPSLA，1583–1612，2022。[56] Joseph Tassarotti 和 Jean-Baptiste Tristan。概率编程语言的经验证密度编译。《ACM 编程语言论文集》7，PLDI。615–637，2023。[57] Peixin Wang、Tengshun Yang、Hongfei Fu、C.-H. Luke Ong 和 Guanyan Li。通过多项式求解对贝叶斯概率编程进行静态后验推断。《ACM 编程语言论文集》，第8卷，第PLDI’24期，202，1361–1386，2024。

[58] Frank Wood、Jan Willem Meent 和 Vikash Mansinghka。概率编程推断的新方法。第十七届人工智能与统计国际会议论文集，*PMLR 33:1024-1032*，2014。[59] Yi Wu、Siddharth Srivastava、Nicholas Hay、Simon Du 和 Stuart Russell。概率编程中的离散-连续混合：广义语义与推断算法。第*35*届国际机器学习会议论文集，*PMLR 80:5343-5352*，2018。

附录 A. 证明

以下结果涵盖了定理 1，是测度论中的一个经典结论。下面的表述是将 [2, Th.2.6.7] 专门应用于马尔可夫核和非负函数的结果。(a) 给出了一种构造乘积空间 Ω^t 上测度的方法，所需起点是一个初始测度 μ^1 和 $t-1$ 个马尔可夫核。直观而言，该乘积空间是马尔可夫链长度为 t 的路径的样本空间。特别地，长度为 $t=1$ 的路径仅由一个初始状态构成——尚未发生任何状态转移。(b) 是富比尼定理的推广，它允许将关于 (a) 中测度的乘积空间上的积分表示为分量空间上的迭代积分。下文中，我们令 ω^t 遍历 Ω^t。

定理 A.1（测度乘积）。设 $t \geq 1$ 为一个整数。设 μ^1 为 Ω 上的一个概率测度，$K_2, \ldots, K_t$ 为从 Ω 到 Ω 的 $t-1$ 个（不一定互异的）马尔可夫核。*(a)* 存在唯一的概率测度 μ^t 定义在 $(\Omega^t, \mathcal{F}^t)$ 上，使得对于每个 $A_1 \times \cdots \times A_t \in \mathcal{F}^t$，我们都有：

$$\mu^t(A_1 \times \cdots \times A_t) \;=\; \int_{A_1} \mu^1(d\omega_1) \int_{A_2} K_2(\omega_1)(d\omega_2) \cdots \int_{A_t} K_t(\omega_{t-1})(d\omega_t)\,. \tag{A.1}$$

(b)（富比尼）设 f 为定义在 Ω^t 上的非负可测函数。那么，令 $\omega^t = (\omega_1, \ldots, \omega_t)$，我们有

$$\int \mu^t(\omega^t) f(\omega^t) \;=\; \int \mu^1(d\omega_1) \int K_2(\omega_1)(d\omega_2) \cdots \int K_t(\omega_{t-1})(d\omega_t) f(\omega^t)\,. \tag{A.2}$$

特别地，在右侧，对于每个 $j = 1, \ldots, t-1$ 以及 $(\omega_1, \ldots, \omega_{j-1})$，函数 $\omega_j \mapsto \int K_{j+1}(\omega_j)(d\omega_{j+1}) \cdots \int K_t(\omega_{t-1})(d\omega_t) f(\omega^t)$ 在 Ω 上可测。

现在我们开始证明论文中陈述的结果。

引理 *1* 的证明。作为一个函数，$\mathbb{R}^{m+1} \times \mathcal{F} \to \mathbb{R}^{+}$，$\kappa$ 可写如下

$$\kappa(v, z)(A) = [z \notin \mathcal{P}] \cdot \delta_v(A_z) \;+\; \sum_{(S,\varphi,\zeta,S') \in E} [z = S] \cdot \varphi(v) \cdot \zeta(v)(A_{S'})\,. \tag{A.3}$$

现在我们来验证定义 1 所要求的两个性质。

- 对于每个 $(v, z) \in \Omega$，函数 $A \mapsto \kappa(v, z)(A)$ 都是一个概率测度。考虑 (A.3) 的右侧：由于对于每个 S，$\sum_{(S,\varphi,\zeta,S') \in E} \varphi(v) = 1$，给定 (v, z) 后，恰好只有一个加项不等于常值零函数。因此，存在一个定义在 $\mathcal{F}_m$ 上的概率测度 ν，使得对于每个 $A \in \mathcal{F}$，都有 $\kappa(v, z)(A) = \nu(A_z)$。接下来，我们注意到可测集截面的以下一般性质，这一性质可通过初等集合论推理得出：对于任意可测集 $A \in \mathcal{F}$，若 $A = \cup_{j \geq 0} A_j$（可测集的不交并）且 $z \in \mathbb{R}$，则有 $A_z = \cup_{j \geq 0} (A_j)_z$（可测集的不交并）。应用上述两个事实以及测度 ν 的可加性，可得：$\kappa(v, z)(A) = \nu((\cup_{j \geq 0} A_j)_z) = \nu(\cup_{j \geq 0} (A_j)_z) = \sum_{j \geq 0} \nu((A_j)_z) = \sum_{j \geq 0} \kappa(v, z)(A_j)$。这表明 $\kappa(v, z)$ 是一个测度。此外，$\kappa(v, z)(\Omega) = \nu(\Omega_z) = \nu(\mathbb{R}^m) = 1$，由此完成了 $\kappa(v, z)$ 是概率测度的证明。
- 对于每个 $A \in \mathcal{F}$，函数 $(v,\ z) \mapsto \kappa(v,\ z)(A)$ 都是非负可测函数。再次考虑 (A.3) 的右端，但将 $\delta_v(A_z)$ 写作示性函数 $1_{A_z}(v)$：将其视为 v 的函数时，它是可测的（因为 A_z 是可测集）。此外，对于任意 A 和 S'，$\zeta(v)(A_{S'})$ 是关于 v 的可测函数（因为 ζ 是马尔可夫核）；因此对每个 φ，$\varphi(v) \cdot \zeta(v)(A_{S'})$ 也是关于 v 的可测函数。但任何仅依赖于 v 的可测函数，例如 $h(v)$，也同样是关于 $(v,\ z)$ 的可测函数（即通过复合得到的函数

投影 $(v,\ z) \mapsto v$ 与 $v \mapsto h(v)$）。同样，谓词 $[z \notin \mathcal{P}]$ 和 $[z = S]$（对任意固定的 $S \in \mathcal{P}$）是关于 z 的可测函数，因此也是关于 $(v,\ z)$ 的可测函数。由于 $\kappa(\cdot)(A)$ 是通过对关于 $(v,\ z)$ 的非负可测函数进行乘积和求和得到的，因此它是关于 $(v,\ z)$ 的可测函数 [2, 第1章，定理1.5。6]。

□

下面给出一个一般性引理，它有助于将无限轨迹集合上的测度与有限轨迹集合上的测度联系起来。在引理陈述中，我们用 μ^∞ 表示柱集 σ-代数上的一般测度，它是按照 Ionescu-Tulcea 定理的意义，由核的无限乘积得到的；用 μ^t 表示相应的有限乘积测度。我们将使用以下记号。对于 $\tilde{\omega} = (\omega_1,\ \omega_2,\ \ldots\ldots) \in \Omega^\infty$，记 $\tilde{\omega}_{1:t} := (\omega_1,\ \ldots\ldots,\ \omega_t) \in \Omega^t$。对于 $h : \Omega^t \to \mathbb{R}^+$ 上的非负函数，定义 $\tilde{h} : \Omega^\infty \to \mathbb{R}^+$ 如下，对每个 $\tilde{\omega} \in \Omega^\infty$：

$$\tilde{h}(\tilde{\omega}) := h(\tilde{\omega}_{1:t})\,. \tag{A.4}$$

接下来，我们将使用可测柱集的以下性质：对于可测的 $A,\ B \subseteq \Omega^t (t \geq 1)$，有 $\mathfrak{c}(A \cup B) = \mathfrak{c}(A) \cup \mathfrak{c}(B)$，且 $\mathfrak{c}(A \cap B) = \mathfrak{c}(A) \cap \mathfrak{c}(B)$。

引理 A.1。设 $h : \Omega^t \to \mathbb{R}^+$ 为非负可测函数。则由 (A.4) 定义的 $\tilde{h}$ 是可测的。此外，对每个可测柱集 $\mathfrak{c}(B_t) \subseteq \Omega^\infty$ $(B_t \subseteq \Omega^t)$，都有 $\int_{\mathfrak{c}(B_t)} \mu^\infty(d\tilde{\omega}) \tilde{h}(\tilde{\omega}) = \int_{B_t} \mu^t(d\omega^t) h(\omega^t)$。

证明。首先，考虑示性函数 $h = 1_{A_t}$ 的情形，其中 $A_t \subseteq \Omega^t$ 是可测的。则 $h = 1_{\mathfrak{c}(A_t)}$ 是由 A_t 生成的可测柱集的示性函数，结论显然成立，因为 h 是可测的，并且 $\int_{\mathfrak{c}(B_t)} \mu^\infty(d\tilde{\omega}) \tilde{h}(\tilde{\omega}) = \int \mu^\infty(d\tilde{\omega}) \tilde{h}(\tilde{\omega}) 1_{\mathfrak{c}(B_t)}(\tilde{\omega}) = \mu^\infty(\mathfrak{c}(B_t) \cap \mathfrak{c}(A_t)) = \mu^\infty(\mathfrak{c}(B_t \cap A_t)) = \mu^t(B_t \cap A_t) = \int_{B_t} \mu^t(d\omega^t) h(\omega^t)$。一般情形下 h 的结论随即由标准的测度论论证（线性性、控制收敛）得出。

□

现在我们可以轻松证明本文中使用的各种函数的可测性。

引理 A.2（函数的可测性）。设 $t \geq 1$。以下函数均可测：*(1)* $\mathrm{w}_t : \Omega^t \to [0,1]$；*(2)* $\mathrm{w} : \Omega^\infty \to [0,1]$；*(3)* $f_t : \Omega^t \to \mathbb{R}^+$，条件是 $f : \Omega^\infty \to \mathbb{R}^+$ 可测。

证明。对于第 1 和第 2 部分，首先注意到，可以将 Ω 上的评分函数（定义（4.1））写为：$\mathrm{sc}(v, z) = [z \notin \mathcal{P}]_ + \sum_{S \in \mathcal{P}} [z = S] \cdot \mathrm{sc}(S)(v)$，其中 $\gamma = \mathrm{sc}(S)$ 是 $\mathbb{R}^m$ 上的可测评分函数。这很容易推出 $\mathrm{sc}(\cdot)$ 在 Ω 上可测（另见引理 1 证明的第二项）。由于 $\mathrm{w}_t(\omega^t) = \mathrm{sc}(\omega_1) \cdots \mathrm{sc}(\omega_t)$ 是 Ω 上可测函数的乘积，因此它是 Ω^t 上的可测函数。现在考虑 $\tilde{\mathrm{w}}_t : \Omega^\infty \to [0,1]$：应用引理 A.1，并取 $h = \mathrm{w}_t$，可推出 $\tilde{\mathrm{w}}_t$ 可测。最后，由于 $t \to +\infty$，可见 $\tilde{\mathrm{w}}_t \to \mathrm{w}$ 逐点收敛：于是 w 也可测，因为它是可测函数序列的逐点极限，参见 [2, 定理 1.5.4]。

关于第 3 部分，定义 t-截面 $C \in \mathcal{C}$ 于 $\tilde{\omega} \in \Omega^\infty$ 为 $C^t_{\tilde{\omega}} := \{\omega^t \in \Omega^t : (\omega^t, \tilde{\omega}) \in C\}$。与 [2, 定理 2.6.2(1)] 中针对有限乘积给出的证明非常相似的证明表明，$\mathcal{C}$ 中元素的 t-截面是可测的。现在令 A 为 $\mathbb{R}$中的任意可测集。根据定义，$f^{-1}(A) = \{(\omega^t, \tilde{\omega}) :\ f(\omega^t, \tilde{\omega}) \in A\}$ 是可测的。该集合在 $\tilde{\omega} = \star^\infty$ 处的 t-截面恰好是 $f_t^{-1}(A)$，因此它是可测的。这说明 f_t 是可测的。

□

现在我们来证明定理 2。首先，我们证明一些关于测度μ_S^∞和 μ_S^t的结果。下面将使用记号 $\int f\,d\mu$ 表示 $\int \mu(d\omega) f(\omega)$。这两种记号可以互换使用；第二种记号更便于表示迭代积分。我们需要以下定义。

定义 A.1 (相容路径)。 我们定义以下可测子集：Ω^t $(t \geq 1)$ 和 Ω^∞:

$$\Theta^{\leq t} := \cup_{j=0}^{t-1} (\mathsf{T}^{\mathrm{c}})^j \cdot \mathsf{T}^{t-j} \qquad \Theta_t := \Theta^{\leq t} \cup (\mathsf{T}^{\mathrm{c}})^t \qquad \Theta := \left(\cup_{j\geq 0} (\mathsf{T}^{\mathrm{c}})^j \cdot \mathsf{T}^\infty\right) \cup (\mathsf{T}^{\mathrm{c}})^\infty . \tag{A.5}$$

引理 A.3。 *(a)* $\mu_S^\infty(\Theta) = 1$。因此，对于任意可测集 A 以及定义在 Ω^∞ 上的非负可测f，有 $\int_A f\,d\mu_S^\infty = \int_{A\cap\Theta} f\,d\mu_S^\infty$。

(b) 令 $t \geq 1$。则 $\mu_S^t(\Theta_t) = 1$。因此，对于任意可测集 A 以及定义在 Ω^t 上的非负可测 f，有 $\int_A f\,d\mu_S^t = \int_{A\cap\Theta_t} f\,d\mu_S^t$。

证明。 我们考虑 (a) 部分。考虑 $\Theta^{\mathrm{c}} = \cup_{j\geq 0} (\Omega^j \cdot \mathsf{T} \cdot \mathsf{T}^{\mathrm{c}})$；注意，该并集通常并非不交并，但这与证明的其余部分无关。对于任意 $j \geq 0$，我们将证明 $\mu_S^\infty(\complement(\Omega^j \cdot \mathsf{T} \cdot \mathsf{T}^{\mathrm{c}})) = 0$，由此即可得证。事实上，根据乘积测度 μ_S^∞ 的定义，$\mu_S^\infty(\complement(\Omega^j \cdot \mathsf{T} \cdot \mathsf{T}^{\mathrm{c}})) = \mu_S^{j+2}(\Omega^j \cdot \mathsf{T} \cdot \mathsf{T}^{\mathrm{c}})$。定理 A.1(a)（富比尼）给出

$$\mu_S^{j+2}(\Omega^j \mathsf{T}\mathsf{T}^{\mathrm{c}}) = \int \delta_{(0,S)}(d\omega_1) \int \kappa(\omega_1)(d\omega_2) \int \cdots \int_{\mathsf{T}} \kappa(\omega_j)(d\omega_{j+1}) \int_{\mathsf{T}^{\mathrm{c}}} \kappa(\omega_{j+1})(d\omega_{j+2}) 1 . \tag{A.6}$$

考虑上述表达式中最内层的两个积分，令 $J(\omega_j) := \int_{\mathsf{T}} \kappa(\omega_j)(d\omega_{j+1}) \int_{\mathsf{T}^{\mathrm{c}}} \kappa(\omega_{j+1})(d\omega_{j+2}) 1 = \int \kappa(\omega_j)(d\omega_{j+1}) \int \kappa(\omega_{j+1})(d\omega_{j+2}) 1_{\mathsf{T}}(\omega_{j+1}) \cdot 1_{\mathsf{T}^{\mathrm{c}}}(\omega_{j+2})$。假设 $\omega_{j+1} = (v, \mathrm{nil}) \in \mathsf{T}$：那么，考虑 $J(\omega_j)$ 中最内层的积分，根据 κ 的定义，我们有 $\int \kappa(\omega_{j+1})(d\omega_{j+2}) 1_{\mathsf{T}}(\omega_{j+1}) \cdot 1_{\mathsf{T}^{\mathrm{c}}}(\omega_{j+2}) = \int \kappa(v, \mathrm{nil})(d\omega_{j+2}) 1_{\mathsf{T}^{\mathrm{c}}}(\omega_{j+2}) = \int \delta_{(v,\mathrm{nil})}(d\omega_{j+2}) 1_{\mathsf{T}^{\mathrm{c}}}(\omega_{j+2}) = 1_{\mathsf{T}^{\mathrm{c}}}(v, \mathrm{nil}) = 0$。类似地，如果 $\omega_{j+1} \in \mathsf{T}^{\mathrm{c}}$，最内层的积分为 0。这意味着 $J(\omega_j) = 0$，因此 (A.6) 中的积分为 0。

（b）部分的证明类似。 □

引理 A.4. 令 $S \in \mathcal{P}$。当 $t \to +\infty$ 时，我们有 $[S]^t 1_{\mathsf{T}^{\leq t}} \cdot \mathrm{w}_t \longrightarrow [S] 1_{\mathsf{T}_{\mathrm{f}}} \cdot \mathrm{w}$。此外，序列 $[S]^t 1_{\mathsf{T}^{\leq t}} \cdot \mathrm{w}_t$ $(t \geq 1)$ 单调非递减。

证明。 对每个 $t \geq 1$，考虑函数 $h_t = 1_\Theta \cdot (1_{\mathsf{T}^{\leq t}} \cdot \mathrm{w}_t)$，它定义在 Ω^∞ 上且可测，因为它是两个可测函数的乘积（Θ 的可测性由引理 1 得出）。容易验证：（1）$(h_t)_{t\geq 1}$ 是单调非递减的函数序列；（2）当 $t \to +\infty$ 时，$h_t \to 1_\Theta \cdot 1_{\mathsf{T}_{\mathrm{f}}} \cdot \mathrm{w}$ 逐点成立。根据单调收敛定理 [2, Th.1.6.2]，$\int h_t d\mu_S^\infty \longrightarrow \int 1_\Theta \cdot 1_{\mathsf{T}_{\mathrm{f}}} \cdot \mathrm{w} d\mu_S^\infty$，其中左侧的积分序列单调非递减。现在，应用引理 A.3(a) 和引理 A.1，并令 $B_t = \Omega^t$，得到 $\int h_t d\mu_S^\infty = \int_\Theta (1_{\mathsf{T}^{\leq t}} \cdot \mathrm{w}_t) d\mu_S^\infty = \int (1_{\mathsf{T}^{\leq t}} \cdot \mathrm{w}_t) d\mu_S^\infty = \int 1_{\mathsf{T}^{\leq t}} \mathrm{w}_t d\mu_S^t$，其中最后一项按定义为 $[S]^t 1_{\mathsf{T}^{\leq t}} \mathrm{w}_t$。同理，根据引理 A.3(a)，$\int 1_\Theta \cdot 1_{\mathsf{T}_{\mathrm{f}}} \cdot \mathrm{w} d\mu_S^\infty = \int 1_{\mathsf{T}_{\mathrm{f}}} \cdot \mathrm{w} d\mu_S^\infty = [S] 1_{\mathsf{T}_{\mathrm{f}}} \cdot \mathrm{w}$。证明完毕。

□

我们需要一个关于前缀闭函数支撑集的引理。

引理 A.5. 设 f 为一个具有分支 L_j $(j \geq 0)$ 和 $t \geq 1$的前缀闭函数。则$L^{>t} \cap T^{\leq t} = \emptyset$。

证明。 对于每个 $0 \leq j \leq t$，我们都有 $L^{>j} \cap T_j = \emptyset$ （T-尊重性），这意味着 $L^{>t} \cap T_j \cdot \Omega^{t-j} = \emptyset$ （因为 $L^{>t} \subseteq L^{>j} \cdot \Omega^{t-j}$）。因此，回想起 $T^{\leq t} = \cup_{j=0}^{t} T_j \cdot \Omega^{t-j}$，我们有$L^{>t} \cap T^{\leq t} = \emptyset$。 □

引理 *2* 的证明 。我们分别证明式 (5.1) 中的上界和下界。

- （上界）。首先，我们来证明包含关系 $\mathrm{supp}(f) \subseteq \mathfrak{c}(L^{\leq t} \cap T^{\leq t}) \cup (\mathfrak{c}(T^{\leq t}))_{\mathrm{c}}$。确实，考虑 $\tilde{\omega} \in \mathrm{supp}(f)$，使得 $\tilde{\omega} \notin \mathfrak{c}(L^{\leq t} \cap T^{\leq t}) = \mathfrak{c}(L^{\leq t}) \cap \mathfrak{c}(T^{\leq t})$。那么，要么$\tilde{\omega} \in \mathfrak{c}(T^{\leq t})_{\mathrm{c}}$，此时无需再证明。要么 $\tilde{\omega} \in \mathfrak{c}(L_j)$ 对某个 $j > t$ 成立，因此$\tilde{\omega} \in \mathfrak{c}(L^{>t})$；根据引理 A.5，$L^{>t} \cap T^{\leq t} = \emptyset$，因此 $\mathfrak{c}(L^{>t}) \cap \mathfrak{c}(T^{\leq t}) = \emptyset$；这意味着$\tilde{\omega} \notin \mathfrak{c}(T^{\leq t})$，即 $\tilde{\omega} \in \mathfrak{c}(T^{\leq t})_{\mathrm{c}}$，从而证明了所需的包含关系。作为刚才证明的包含关系的推论，

$$[S]f \cdot \mathrm{w} \leq \underbrace{[S]f \cdot 1_{\mathfrak{c}(L^{\leq t} \cap T^{\leq t})} \cdot \mathrm{w}}_{K_1} + \underbrace{[S]f \cdot 1_{\mathfrak{c}(T^{\leq t})^{\mathrm{c}}} \cdot \mathrm{w}}_{K_2} . \tag{A.7}$$

现在我们分别对 K_1 和 K_2给出上界。
–的上界 K_1。利用 (A.4) 中引入的记号，我们首先验证

$$\text{on } \mathfrak{c}(L^{\leq t}), \text{ hence on } \mathfrak{c}(L^{\leq t} \cap T^{\leq t}), \text{ we have } f \cdot \tilde{\mathrm{w}}_t = \widetilde{(f_t \cdot \mathrm{w}_t)} . \tag{A.8}$$

~事实上，对于任意 $\omega^t \in L^{\leq t}$和 $\tilde{\omega} \in \Omega^{\infty}$，我们有：$(f_t \cdot \mathsf{w}_t)(\omega^t, \tilde{\omega}) = (f_t \cdot \mathsf{w}_t)(\omega^t) = f_t(\omega^t) \cdot \mathsf{w}_t(\omega^t) = f(\omega^t, \star^{\infty}) \cdot \tilde{\mathsf{w}}_t(\omega^t, \tilde{\omega}) = f(\omega^t, \tilde{\omega}) \cdot \tilde{\mathsf{w}}_t(\omega^t, \tilde{\omega})$，其中等式 $f(\omega^t, \star^{\infty}) = f(\omega^t, \tilde{\omega})$ 成立，是因为 f 是前缀闭合的，且对于某个$0 \leq j \leq t$，$(\omega^t, \tilde{\omega})$、$(\omega^t, \star^{\infty}) \in \mathfrak{c}(L_j)$；这就证明了 (A.8)。现在我们有

$$\begin{aligned}
K_1 &= \int_{\mathfrak{c}(L^{\leq t} \cap T^{\leq t})} f \cdot \mathrm{w} \, d\mu_S^{\infty} && \text{(A.9)} \\
&\leq \int_{\mathfrak{c}(L^{\leq t} \cap T^{\leq t})} f \cdot \tilde{\mathrm{w}}_t \, d\mu_S^{\infty} && \text{(A.10)} \\
&= \int_{\mathfrak{c}(L^{\leq t} \cap T^{\leq t})} \widetilde{(f_t \cdot \mathrm{w}_t)} \, d\mu_S^{\infty} && \text{(A.11)} \\
&= \int_{L^{\leq t} \cap T^{\leq t}} f_t \cdot \mathrm{w}_t \, d\mu_S^{t} && \text{(A.12)} \\
&= [S]^t f_t \cdot 1_{L^{\leq t} \cap T^{\leq t}} \cdot \mathrm{w}_t && \text{(A.13)}
\end{aligned}$$

其中：(A.10) 来源于 $\mathsf{w} \leq \tilde{\mathsf{w}}_t$；在 (A.11) 中，我们使用了 (A.8)；在 (A.12) 中，我们应用了引理 A.1，其中 $h = f_t \cdot \mathsf{w}_t$和 $B_t = L^{\leq t} \cap T^{\leq t}$。
– 的上界 K_2。由 $f \leq M$和$\mathsf{w} \leq \tilde{\mathsf{w}}_t$，我们得到$K_2 \leq M \cdot [S]1_{\mathfrak{c}(T^{\leq t})^{\mathrm{c}}} \tilde{\mathsf{w}}_t = M \cdot (\int \tilde{\mathsf{w}}^t d\mu_S^{\infty} - \int_{\mathfrak{c}(T^{\leq t})} \tilde{\mathsf{w}}_t d\mu_S^{\infty})$。现在，对 $h = \mathsf{w}_t$应用引理 A.1：首先令$B_t = \Omega^t$，得到 $\int \tilde{\mathsf{w}}_t d\mu_S^{\infty} = \int \mathsf{w}_t d\mu_S^{t} = [S]^t \mathsf{w}_t$；然后令 $B_t = T^{\leq t}$，得到$\int_{\mathfrak{c}(T^{\leq t})} \tilde{\mathsf{w}}_t d\mu_S^{\infty} = \int_{T^{\leq t}} \mathsf{w}_t d\mu_S^{t} = [S]^t \mathsf{w}_t \cdot 1_{T^{\leq t}}$。总之，我们有：

$$K_2 \leq M \cdot ([S]^t \mathrm{w}_t - [S]^t 1_{T^{\leq t}} \cdot \mathrm{w}_t) = M \cdot [S]^t (1 - 1_{T^{\leq t}}) \cdot \mathrm{w}_t . \tag{A.14}$$

• （下界）。回顾一下，对于任意 $j \geq 1$， $T_j = (\mathsf{T}_{\mathfrak{c}})^{j-1} \cdot \mathsf{T}$，且 $T^{\leq t} = \cup_{j=1}^{t} T_j \cdot \Omega^{t-j} \subseteq \Omega^t$。现在考虑 $\Theta^{\leq t} = \cup_{j=1}^{t} T_j \cdot \mathsf{T}^{t-j} \subseteq \Omega^t$。为简洁起见，我们采用以下缩写：

$$A_t := \Theta^{\leq t} \cdot \mathsf{T}^{\infty} \cap \mathfrak{c}(L^{\leq t} \cap T^{\leq t}) \,. \tag{A.15}$$

显然， $[S]f\mathsf{w} \geq [S]f\mathsf{w}1_{A_t} = \int_{A_t} f\mathsf{w}\, d\mu_S^{\infty}$。现在我们来验证：

$$\text{on } A_t, \text{ we have } f\mathrm{w} = \widetilde{(f_t \mathrm{w}_t)} \,. \tag{A.16}$$

事实上，对于任意 $(\omega^t, \tilde{\omega}) \in A_t$，我们有：$(f\mathsf{w})(\omega^t, \tilde{\omega}) = f(\omega^t, \tilde{\omega}) \cdot \mathsf{w}(\omega^t, \tilde{\omega}) = f(\omega^t, \star^{\infty}) \cdot \widetilde{\mathsf{w}(\omega^t, \tilde{\omega})} = f_t(\omega^t) \cdot \mathsf{w}_t(\omega^t) = (f_t \cdot \mathsf{w}_t)(\omega^t) = (f_t \cdot \mathsf{w}_t)(\omega^t, \tilde{\omega})$，其中：(i) $f(\omega^t, \tilde{\omega}) = f(\omega^t, \star^{\infty})$ 源于 f 是前缀闭合的，并且对于某个 $0 \leq j \leq t$，$(\omega^t, \tilde{\omega})$、$(\omega^t, \star^{\infty}) \in \mathfrak{c}(L_j)$；(ii) $\mathsf{w}(\omega^t, \tilde{\omega}) = \mathsf{w}_t(\omega^t)$ 源于对于某个 $0 \leq j \leq t$，$(\omega^t, \tilde{\omega}) \in T_j \cdot \mathsf{T}^{\infty}$；回想一下，定义在 Ω 上的基本权重函数 $\mathsf{sc}(\cdot)$ 在 T 上取值为 1；由此证明了 (A.16)。现在我们有：

$$\begin{aligned}
[S]f \cdot \mathrm{w} &\geq \int_{A_t} f\mathrm{w}\, d\mu_S^{\infty} && \text{(A.17)}\\
&= \int_{A_t} \widetilde{(f_t \mathrm{w}_t)}\, d\mu_S^{\infty} && \text{(A.18)}\\
&= \int_{\mathfrak{c}(L^{\leq t} \cap T^{\leq t})} \widetilde{(f_t \mathrm{w}_t)}\, d\mu_S^{\infty} && \text{(A.19)}\\
&= \int_{L^{\leq t} \cap T^{\leq t}} (f_t \mathrm{w}_t)\, d\mu_S^{t} && \text{(A.20)}\\
&= [S]^t f_t \mathrm{w}_t 1_{L^{\leq t} \cap T^{\leq t}} && \text{(A.21)}
\end{aligned}$$

其中：第二步使用了 (A.16)；第三步应用了引理 A.3 以及 $\mathfrak{c}(L^{\leq t} \cap T^{\leq t}) \cap \Theta = A_t$ 这一事实；倒数第二步则使用了引理 A.1，其中 $h = f_t \mathsf{w}_t$ and $B_t = L^{\leq t} \cap T^{\leq t}$。

(A.13)、(A.14) 和 (A.21) 中的界推出了所需的界 (5.1)。 □

定理 *2* 的证明 我们首先证明 $[\![S]\!]f$ 定义良好，也就是说，$[S]\mathsf{w} > 0$。事实上，由引理 A.4 以及至少对某个 t 有 $[S]^t 1_{\mathsf{T}^{\leq t}} \cdot \mathsf{w}_t > 0$，可得 $[S]1_{\mathsf{T}_{\mathrm{f}}} \cdot \mathsf{w} > 0$；由于 $1_{\mathsf{T}_{\mathrm{f}}} \cdot \mathsf{w} \leq \mathsf{w}$，因此可得 $[S]1_{\mathsf{T}_{\mathrm{f}}} \cdot \mathsf{w} \leq [S]\mathsf{w}$，从而得到所需结论。

现在考虑 $[\![S]\!]f = \frac{[S]f \cdot \mathsf{w}}{[S]\mathsf{w}}$，其中 f 如假设所示，并考虑 (5.2) 中的不等式。考虑该分数分子和分母的以下界。

$$[S]^t f_t \cdot 1_{L^{\leq t} \cap T^{\leq t}} \cdot \mathrm{w}_t \leq [S]f \cdot \mathrm{w} \leq [S]^t f_t \cdot 1_{L^{\leq t} \cap T^{\leq t}} \cdot \mathrm{w}_t + M([S]^t \mathrm{w}_t - [S]^t 1_{T^{\leq t}} \cdot \mathrm{w}_t) \tag{A.22}$$

$$[S]^t 1_{T^{\leq t}} \cdot \mathrm{w}_t \leq [S]\mathrm{w} \leq [S]^t \mathrm{w}_t \,. \tag{A.23}$$

(A.22) 中的界与引理 2 中的界相同，只是将项 $M \cdot (\cdots)$ 写成了等价形式。至于 (A.23)，首先将引理 2 的界应用于常值函数 $f = 1$。注意，此 f 是可测的，并且对于前缀无关语言序列 $L_0 = \{\epsilon\}$ 和 $L_j = \emptyset$，对 $j > 0$而言，它显然是前缀闭合的。因此，对于 $t \geq 1$，在 Ω^t 上有$L^{\leq t} = \Omega^t$，故有 $1_{L^{\leq t} \cap T^{\leq t}} = 1_{T^{\leq t}}$。此外， $f_t = M = 1$恒等成立。由此可立即看出，引理 2 的界 (5.1) 特化后即为 (A.23)。根据上文已为 $[\![S]\!]f = \frac{[S]f \cdot \mathrm{w}}{[S]\mathrm{w}}$ 的分子和分母建立的界 (A.22) 和 (A.23)，

可得

$$\frac{[S]^t f_t \cdot 1_{L^{\le t}\cap T^{\le t}} \cdot \mathrm{w}_t}{[S]^t \mathrm{w}_t} \quad \le \quad [\![S]\!] f \quad \le \quad \frac{[S]^t f_t \cdot 1_{L^{\le t}\cap T^{\le t}} \cdot \mathrm{w}_t}{[S]^t 1_{T^{\le t}} \cdot \mathrm{w}_t} + M \cdot \left(\frac{[S]^t \mathrm{w}_t}{[S]^t 1_{T^{\le t}} \cdot \mathrm{w}_t} - 1 \right). \quad \text{(A.24)}$$

现在，将上述上界的第一项乘以并除以 $[S]^t \mathrm{w}_t$；根据假设，该量为正。再回忆 α_t 的定义，即可得到所需的 (5.2)。□

证明 定理 3。将 (5.2) 写成 (A.24) 的形式。我们讨论其中三个不同项分别在 $t \to +\infty$ 时的极限。

(1) $[S]^t f_t 1_{L^{\le t}\cap T^{\le t}} \mathrm{w}_t \to \int f \mathrm{w}\, d\mu_S^\infty = [S] f \mathrm{w}$。考虑 (A.15) 中引入的集合 $A_t = \Theta^{\le t} \cdot \mathrm{T}^\infty \cap \mathfrak{c}(L^{\le t} \cap T^{\le t})$。我们有以下等式

$$\begin{aligned} [S]^t f_t \cdot 1_{L^{\le t}\cap T^{\le t}} \cdot \mathrm{w}_t &= \int_{A_t} f \cdot \mathrm{w}\, d\mu_S^\infty \\ &= \int f \cdot \mathrm{w} \cdot 1_{\mathfrak{c}(T^{\le t}\cap L^{\le t})}\, d\mu_S^\infty \end{aligned} \quad \text{(A.25)}$$

其中：第一个等式已在 (A.17)–(A.21) 中证明，第二个等式则由引理 A.3 以及等式 $A_t \cap \Theta = \mathfrak{c}(T^{\le t} \cap L^{\le t}) \cap \Theta$ 得出。现在，根据 $\mathrm{supp}(f) = \cup_{j\ge 0} \mathfrak{c}(L_j)$ 的定义，$1_{\mathfrak{c}(T^{\le t}\cap L^{\le t})}$ 逐点收敛于 $1_{T_{\mathrm{f}}\cap \mathrm{supp}(f)}$：特别地，对于每个 $\tilde{\omega}$，当 t 足够大时，有 $1_{\mathfrak{c}(T^{\le t}\cap L^{\le t})}(\tilde{\omega}) = 1_{T_{\mathrm{f}}\cap \mathrm{supp}(f)}(\tilde{\omega})$。这进而意味着，$f \cdot \mathrm{w} \cdot 1_{\mathfrak{c}(T^{\le t}\cap L^{\le t})}$ 逐点收敛于 $f \cdot \mathrm{w} \cdot 1_{T_{\mathrm{f}}\cap \mathrm{supp}(f)} = f \cdot \mathrm{w} \cdot 1_{T_{\mathrm{f}}}$（即使 f 取值为 $+\infty$）。此外，函数序列 $f \cdot \mathrm{w} \cdot 1_{\mathfrak{c}(T^{\le t}\cap L^{\le t})}$ 单调不减。由单调收敛定理 [2, Th.1.6.2]，(A.25) 中的积分随后收敛于 $\int f \cdot \mathrm{w} \cdot 1_{T_{\mathrm{f}}}\, d\mu_S^\infty = \int f \cdot \mathrm{w}\, d\mu_S^\infty$，其中最后一个等式源于 $\mu_S^\infty(T_{\mathrm{f}}) = 1$，以及再次应用引理 A.3。

(2) $[S]^t \mathrm{w}_t \to [S]\mathrm{w}$。根据引理 A.1 中的 $B_t = \Omega^t$，我们得到：$[S]^t \mathrm{w}_t = \int \mathrm{w}_t\, d\mu_S^t = \int \widetilde{\mathrm{w}}_t\, d\mu_S^\infty$。此外，函数序列 $\widetilde{\mathrm{w}}_t$ 逐点收敛于 w，且这些函数均被一个可积函数（例如 1）所控制。应用控制收敛定理 [2, 1.6.9],，我们得到 $\int \widetilde{\mathrm{w}}_t\, d\mu_S^\infty \to \int \mathrm{w}_t\, d\mu^\infty = [S]\mathrm{w}$，这正是所需结论。

(3) $[S]^t 1_{T^{\le t}} \cdot \mathrm{w}_t \to [S]\mathrm{w}$。对常值函数 $f = 1$应用上面的第一项。注意，这个 f 显然是前缀闭合的，且 $L_0 = \{\epsilon\}$，因此对每个 $t \ge 1$都有 $L^{\le t} = \Omega^t$。

至于 (5.3)，我们有 $[S]^t 1_{T^{\le t}} \mathrm{w}_t = \int_{T^{\le t}} \mathrm{w}_t\, d\mu_S^t = \int \mathrm{w}_t\, d\mu_S^t = [S]^t \mathrm{w}_t$，这由假设 $\mu_S^t(T^{\le t}) = 1$ 以及初等测度论推理得出。因此 $\alpha_t = 1$ 并且由 (5.2) 可知，$[\![S]\!] f$ 的上下界与 $\frac{[S]^t f_t \cdot 1_{L^{\le t}\cap T^{\le t}} \cdot \mathrm{w}}{[S]^t \mathrm{w}_t}$ 一致。现在我们来验证

$$f_t \cdot 1_{T^{\le t}} = f_t \cdot 1_{L^{\le t}\cap T^{\le t}}.$$

事实上，由于 $\mathrm{supp}(f_t) \subseteq L^{\le t} \cup L^{>t}$，我们可以针对 $\omega^t \in \mathrm{supp}(f_t)$ 考虑两种情况。若 $\omega^t \in \mathrm{supp}(f_t) \cap L^{\le t}$：根据定义，我们有 $(f_t \cdot 1_{T^{\le t}})(\omega^t) = (f_t \cdot 1_{L^{\le t}\cap T^{\le t}})(\omega^t)$。若 $\omega^t \in \mathrm{supp}(f_t) \cap L^{>t}$：那么根据引理 A.5，我们有 $\omega^t \notin T^{\le t}$，因此同样有 $(f_t \cdot 1_{T^{\le t}})(\omega^t) = (f_t \cdot 1_{L^{\le t}\cap T^{\le t}})(\omega^t) = 0$。最后，我们可以如下计算：$[S]^t f_t \cdot \mathrm{w}_t = [S]^t f_t \cdot 1_{T^{\le t}} \cdot \mathrm{w}_t = [S]^t f_t \cdot 1_{L^{\le t}\cap T^{\le t}} \cdot \mathrm{w}_t$，其中第一个等式由 $\mu(T^{\le 1}) = 1$ 以及初等的测度论推理得出，第二个等式则由上文已证明的等式得出。至此，(5.3) 的证明完成。□

下面的结果有助于将我们的语义与费曼–卡茨模型的滤波分布联系起来。这里，我们令 $\mathrm{pr}_j : \Omega^t \to \Omega$ $(1 \le j \le t)$ 表示到第 j 个分量的投影；这是一个可测函数。该结果基本上说明，在长度为 t 的路径上对 f_t 取期望，与对 $h \circ \mathrm{pr}_t$ 取期望相同，后者只考察路径的最后一个状态。

引理 A.6. 设 $f = h$ 为定义在 Ω 上的非负可测函数。对于每个 S 和 $t \ge 1$，我们有 $[S]^t f_t \cdot 1_{L^{\le t}} \cdot \mathrm{w}_t = [S]^t (h \circ \mathrm{pr}_t) \cdot \mathrm{w}_t$，其中 L_j $(j \ge 0)$ 是 f 的分支。

证明。我们首先证明关于测度 μ_S^t 的一个一般性结论。回顾一下，我们用 ω^t 表示元组 $(\omega_1, \ldots, \omega_t)$。设 $g \colon \Omega^t \to \mathbb{R}^+$ 是一个可测函数。对于每个 $1 \le j \le t$，令 $\eta_j(\omega^t) := [\omega_{j+1} = \omega_j] \cdot \cdots \cdot [\omega_t = \omega_j]$ 为一个谓词，当且仅当 $\omega_j = \cdots = \omega_t$ 时取值为 1。于是，回想起 $T_j = (\mathsf{T}^c)^{j-1} \cdot \mathsf{T}$，我们有

$$\int_{T_j \cdot \Omega^{t-j}} \mu_S^t(d\omega^t)\, g(\omega^t) = \int_{T_j \cdot \Omega^{t-j}} \mu_S^t(d\omega^t)\, g(\omega^t) \cdot \eta_j(\omega^t)\,. \tag{A.26}$$

此等式可按如下方式验证。首先，利用定理 A.1(b)（富比尼），将 (A.26) 左侧（分别为右侧）的积分写成迭代积分：在所得表达式中，将对应于 $t - j$ 个最内层迭代积分的表达式记为 H_j（分别为 K_j）。对于每个 $\omega_1, \ldots\ldots, \omega_j$，我们有以下等式

$$\begin{aligned}
H_j &= \int \kappa(\omega_j)(d\omega_{j+1}) \int \kappa(\omega_{j+1})(d\omega_{j+2}) \cdots \int \kappa(\omega_{t-1})(d\omega_t)\, g(\omega^t) \cdot 1_{T_j \cdot \Omega^{t-j}}(\omega^t) \\
&= g(\omega_1, \ldots \omega_{j-1}, \omega_j, \ldots, \omega_j) \cdot 1_{\mathsf{T}^c}(\omega_1) \cdots 1_{\mathsf{T}^c}(\omega_{j-1}) \cdot 1_{\mathsf{T}}(\omega_j) && \text{(A.27)} \\
&= g(\omega_1, \ldots \omega_{j-1}, \omega_j, \ldots, \omega_j) \cdot 1_{\mathsf{T}^c}(\omega_1) \cdots 1_{\mathsf{T}^c}(\omega_{j-1}) \cdot 1_{\mathsf{T}}(\omega_j) \cdot \eta_j(\omega_1, \ldots \omega_{j-1}, \omega_j, \ldots, \omega_j) && \text{(A.28)} \\
&= \int \kappa(\omega_j)(d\omega_{j+1}) \int \kappa(\omega_{j+1})(d\omega_{j+2}) \cdots \int \kappa(\omega_{t-1})(d\omega_t)\, g(\omega^t) \cdot \eta_j(\omega^t) \cdot 1_{T_j \cdot \Omega^{t-j}}(\omega^t) && \text{(A.29)} \\
&= K_j
\end{aligned}$$

其中：(A.27) 在 $\omega_j \notin \mathsf{T}$ 时显然成立，因为此时两边均为 0；若 $\omega_j \in \mathsf{T}$，例如 $\omega_j = (v, \mathsf{nil})$，则根据 κ 的定义，有 $\kappa(\omega_j)(\cdot) = \delta_{\omega_j}(\cdot)$，于是对狄拉克测度的性质 $\int \delta_\omega(d\omega') q(\omega') = q(\omega)$ 反复应用，即可得到 (A.27)；根据 η_j 的定义可得 (A.28)；根据与 (A.27) 相同的推理可得 (A.29)。现在，对 $\omega_1, \ldots, \omega_j$ 积分 H_j 和 K_j（注意，根据富比尼定理，两边都是 $\omega_1, \ldots, \omega_j$ 的可测函数），然后对两边应用富比尼定理，将所得的迭代积分改写为

关于 Ω^t 的积分。现在我们有

$$\int_{T_j\cdot\Omega^{t-j}} \mu_S^t(d\omega^t)\,(f_t\cdot 1_{L^{\le t}}\cdot \mathrm{w}_t)(\omega^t) = \int_{T_j\cdot\Omega^{t-j}} \mu_S^t(d\omega^t)\,(f_t\cdot \mathrm{w}_t)(\omega^t) \tag{A.30}$$

$$= \int_{T_j\cdot\Omega^{t-j}} \mu_S^t(d\omega^t)\,(f_t\cdot \mathrm{w}_t\cdot \eta_j)(\omega^t) \tag{A.31}$$

$$= \int_{T_j\cdot\Omega^{t-j}} \mu_S^t(d\omega^t)\,(\mathrm{w}_t\cdot \eta_j)(\omega^t)\cdot h(\omega_j) \tag{A.32}$$

$$= \int_{T_j\cdot\Omega^{t-j}} \mu_S^t(d\omega^t)\,(\mathrm{w}_t\cdot \eta_j)(\omega^t)\cdot h(\omega_t) \tag{A.33}$$

$$= \int_{T_j\cdot\Omega^{t-j}} \mu_S^t(d\omega^t)\,(\mathrm{w}_t\cdot \eta_j\cdot (h\circ \mathrm{pr}_t))(\omega^t) \tag{A.34}$$

$$= \int_{T_j\cdot\Omega^{t-j}} \mu_S^t(d\omega^t)\,((h\circ \mathrm{pr}_t)\cdot \mathrm{w}_t)(\omega^t) \tag{A.35}$$

其中利用了以下等式，对于 $\omega^t \in T_j\cdot\Omega^{t-j}$：在 (A.30) 中，$(1_{L^{\le t}}\cdot f_t)(\omega^t) = f_t(\omega^t)$，因为 $\mathrm{supp}(f_t) \cap T_j\cdot\Omega^{t-j} \subseteq L^{\le t}\cap T_j\cdot\Omega^{t-j}$；在 (A.31) 中，使用 (A.26)，其中 $g = f_t\cdot \mathrm{w}_t$；在 (A.32) 中，$f_t(\omega^t) = h(\omega_j)$；在 (A.33) 中，$\eta_j(\omega^t)h(\omega_j) = \eta_j(\omega^t)h(\omega_t)$；在 (A.34) 中，使用 $h(\omega_t)$ 的定义以及 $\mathrm{pr}_t(\omega^t)$；在 (A.35) 中，再次使用 (A.26)，其中 $g = (h\circ \mathrm{pr}_t)\cdot \mathrm{w}_t$。现在，我们有：

$$[S]^t f_t\cdot 1_{L^{\le t}}\cdot \mathrm{w}_t = \int f_t\cdot 1_{L^{\le t}}\cdot \mathrm{w}_t\, d\mu^t$$

$$= \int_{T^{\le t}} f_t\cdot 1_{L^{\le t}}\cdot \mathrm{w}_t\, d\mu^t \tag{A.36}$$

$$= \sum_{j=1}^{t}\int_{T_j\cdot\Omega^{t-j}} f_t\cdot 1_{L^{\le t}}\cdot \mathrm{w}_t\, d\mu^t \tag{A.37}$$

$$= \sum_{j=1}^{t}\int_{T_j\cdot\Omega^{t-j}} (h\circ \mathrm{pr}_t)\cdot \mathrm{w}_t\, d\mu^t \tag{A.38}$$

$$= \int_{T^{\le t}} (h\circ \mathrm{pr}_t)\cdot \mathrm{w}_t\, d\mu^t \tag{A.39}$$

$$= \int_{\Theta^{\le t}} (h\circ \mathrm{pr}_t)\cdot \mathrm{w}_t\, d\mu^t \tag{A.40}$$

$$= \int_{\Theta^{\le t}} (h\circ \mathrm{pr}_t)\cdot \mathrm{w}_t\, d\mu^t + \int_{(\mathrm{T}^{\mathrm{c}})^t} (h\circ \mathrm{pr}_t)\cdot \mathrm{w}_t\, d\mu^t \tag{A.41}$$

$$= \int_{\Theta_t} (h\circ \mathrm{pr}_t)\cdot \mathrm{w}_t\, d\mu^t \tag{A.42}$$

$$= \int (h\circ \mathrm{pr}_t)\cdot \mathrm{w}_t\, d\mu^t \tag{A.43}$$

其中：(A.36) 由 $L^{\le t}\subseteq T^{\le t}$ 得出；(A.37) 由 $T^{\le t} = \cup_{j=1}^t T_j\cdot\Omega^{t-j}$（不交并）以及积分的基本性质得出；(A.38) 由 (A.30)—(A.35) 中建立的等式得出；(A.39) 再次由 $T^{\le t} = \cup_{j=1}^t T_j\cdot\Omega^{t-j}$ 得出；(A.40) 由引理 A.3(b) 和 $\Theta_t\cap T^{\le t} = \Theta^{\le t}$ 得出；(A.41) 由以下事实得出：$h\circ \mathrm{pr}_t$ 在 $(\mathrm{T}^{\mathrm{c}})^t$ 上恒等于 0，因为 $\mathrm{supp}(h)\subseteq \mathrm{T}$，因此此处第二个积分为 0；(A.42) 由 $\Theta_t = \Theta_{\le t}\cup(\mathrm{T}^{\mathrm{c}})^t$ 的定义（不交并）得出；(A.43) 再次由引理 A.3(b) 得出。 □

证明 定理 4。下列等式很容易验证，并且在下文中会用到。

$$\mathrm{E}_{\phi_t}[h] = \frac{\mathrm{E}_{\mu^t}[(h \circ \mathrm{pr}_t) \cdot G]}{\mathrm{E}_{\mu^t}[G]}\,. \tag{A.44}$$

至于实际证明，首先注意到，根据提升的定义（定义 8），$L^{\leq t} \subseteq T^{\leq t}$，因此 $f_t \cdot 1_{L^{\leq t} \cap T^{\leq t}} = f_t \cdot 1_{L^{\leq t}}$。关于 β_L，由（5.2）中的下界可得：

$$[\![S]\!] f \geq \frac{[S]^t f_t \cdot 1_{L^{\leq t} \cap T^{\leq t}} \cdot \mathrm{w}_t}{[S]^t \mathrm{w}_t} = \frac{[S]^t f_t \cdot 1_{L^{\leq t}} \cdot \mathrm{w}_t}{[S]^t \mathrm{w}_t} = \frac{[S]^t (h \circ \mathrm{pr}_t) \cdot \mathrm{w}_t}{[S]^t \mathrm{w}_t} = \mathrm{E}_{\phi_{S,t}}[h]$$

其中，在倒数第二步中，我们将引理 A.6 应用于分式的分子；在最后一步中，我们将（A.44）应用于模型 FK_S，并以 $G = \mathsf{w}_t$ 作为全局势函数。

关于 β_U，考虑函数 $f = \check{1}_{\mathsf{T}}$（函数 $h = 1_{\mathsf{T}}$ 的提升），其分支为 $L_j = T_j$：不难验证，对于每个 $t \geq 1$，$1_{T^{\leq t}} \cdot \mathsf{w}_t = f_t \cdot 1_{T^{\leq t}} \cdot \mathsf{w}_t$。然后，我们可以对 β_L 使用上述推理，并利用该函数 f 证明 $\frac{[S]^t 1_{T^{\leq t}} \cdot \mathrm{w}[S]^t}{\mathrm{w}_t} = \mathrm{E}_{\phi_{S,t}}[1_{\mathsf{T}}]$，即 $\alpha_t = \mathrm{E}_{\phi_{S,t}}[1_{\mathsf{T}}]^{-1}$。随后，（5.2）中的上界使我们能够完成证明。□

附录 B. 粒子滤波算法

从计算角度来看，我们关注 FK 模型，是因为它们能够以简单、统一的方式呈现一类高效的推断算法，这类算法称为粒子滤波 *(PF)* [47, 28]。下文将介绍一种计算滤波分布 ϕ_t 的算法。我们将按照 [28, 第11]章的思路，在下文介绍一种通用的粒子滤波算法方案。

固定一个通用的 FK 模型，$\mathsf{FK} = (\mathcal{X}, t, \mu^1, \{K_i\}_{i=2}^t, \{G_i\}_{i=1}^t)$。固定 $N \geq 1$，即粒子数，也就是我们要模拟的由 K_i 表示的随机过程的实例。对于任意一组 $W = W^{1:N} = (W^{(1)}, \ldots, W^{(N)})$ 非负实随机变量（即权重），将 W 的归一化版本记为 $\widehat{W}$，也就是[9] $\widehat{W}^{(i)} = W^{(i)} / (\sum_{j=1}^N W^{(j)})$。对于 (N, W)，重采样方案是取值于 $1..N$ 的 N 元随机变量组，记作 $R = (R_1, \ldots, R_N)$，满足：对于每个 $i \in 1..N$，令 F_i 表示 R 中 i 出现的次数，则有

$$\mathrm{E}[F_i | W] = N \cdot \widehat{W}_i\,.$$

换句话说，R 是一个随机选择过程：从 $1..N$个索引中有放回地选取 N 个索引，使得平均而言，每个索引 $i \in 1..N$ 被选取的次数与其在 W中的权重成正比。我们用 $R(W)$表示 R取决于给定的权重向量 W。文献中提出了多种重采样方案。或许最简单的方法是令 R 为 N 个独立同分布的随机变量，且每个变量均服从 W：这称为多项式重采样。有关详情和高效的实现方法，请参阅粒子滤波方面的专业文献，例如 [28,第9] 章及其中的参考文献。

算法 1 是一个通用的粒子滤波算法。在第 k次迭代时，对于 $k = 1$、..... t，提取两个 N元组：

- 一个状态元组 $X_k = X^{1:N}_k = (X_{(1)k}, \ldots, X_{(N)k}) \in \mathcal{X}^N$；

[9] 但需注意，例如在所有 $W_{(i)}$均为 0 的情况下，$\widehat{W}^{(i)} := 1/N$。在粒子滤波算法的执行过程中，该事件发生的概率为 $\to 0$，因为 $N \to +\infty$。

• 一个（未归一化的）权重元组 $W_k = W^{1:}{}_k{}^N = (W_{(1)k}, ..., W_{(N)k}) \in (\mathbb{R}^+)^N$。

$X_k^{1:N}$中的元素取决于前一次迭代的元组 $X_{k-1}^{1:N}$，W_{k-1}。重采样步骤 4 的目的是在提取下一组粒子元组时，提高权重较高的粒子的重要性，同时丢弃权重较低的粒子。若 R 是多项式重采样，则步骤 4–5 相当于从（经验）分布 $\sum_{j=1}^N \widehat{W}_{(j)k-1}\delta_{K_k(X_{k-1}^{(j)})}$中抽取每个 $X_k^{(j)}$。权重 $W_k^{1:N}$通过势函数G_k计算得出；若 $k < t$，则会在第 $k+1$ 次迭代的重采样步骤中使用这些权重，否则将其作为算法输出的一部分返回。

以下定理从渐近意义上说明了粒子滤波算法相对于滤波分布 ϕ_t 在 $\mathcal{X}$上的一致性。其实际意义在于，我们可以将期望估计为相对于滤波分布的加权和。请注意，在定理陈述中，t 保持不变——它是费曼–卡茨模型的一个参数——而粒子数 N 趋于 $+\infty$。

定理 B.1（粒子滤波的收敛性，[28]）。*考虑算法 1 返回的随机变量（X_t, W_t）($t \geq 1$)。假设 Feynman–Kac 测度 ϕ 在 $\mathcal{X}^t$上定义良好。进一步假设 R（·）为多项式重采样，且势函数 G_k 均有有限上界。对于定义在 $\mathcal{X}$上的每个非负可测函数 h，有：$\sum_{j=1}^N \widehat{W}_t^{(j)} \cdot h$（$X_t^{(j)}$）$\longrightarrow \mathrm{E}_{\phi_t}[h]$ 几乎必然地[10] 当 $N \to +\infty$时。*

附录 C. 实验的更多细节

我们对所考虑的示例作更详细的文字描述。

(1)飞机跟踪(AT) [59]。飞机被建模为一个点，依据高斯过程在二维平面上移动。六部雷达对飞机进行跟踪：在每个离散时间步，每部雷达都会带有噪声地测量飞机与自身位置之间的距离；观测到的是特定距离。我们的目标是求飞机最终水平位置的后验期望值。这是迄今为止所考虑的示例中最复杂的一个；本节末尾将详细介绍如何用概率程序图（PPG）对其进行编码。(2) 醉汉与老鼠(DMM)，例 3。我们的目标是求醉汉方差的后验期望值。(3) 兔子与乌龟(HT)，参见例如 [4]。该模型模拟野兔与乌龟沿一维直线进行赛跑：乌龟每个时间步前进一步，步长为 1；而野兔偶尔迈出一步，其步长服从高斯分布。此外，在每个时间步，观测结果都表明野兔与乌龟之间的距离始终不超过 10。野兔一超过乌龟，比赛便立即终止。我们的目标是求野兔最终位置的后验期望值。(4) 有界重传协议，[40]。若干数据包需要通过有损信道传输，每个数据包丢失的概率为 0.02。只有在传输最后 80 个数据包时，才能观测到丢包情况。如果没有任何数据包需要重传超过 4 次，则认为传输成功。我们的目标是求失败概率的后验期望值。

[10]参见 [28, 第11] 章，其中给出了该断言成立所需概率空间的精确定义　　e.

(5) 非独立同分布循环，[40]。该模型描述了一个离散采样器的行为：它不断抛掷两枚公平硬币，直到两枚硬币都出现反面。此外，观测结果表明，在每次迭代中，至少有一枚硬币的结果与上一次迭代相同。这一观测在连续循环迭代之间引入了数据依赖。我们的目标是求直到终止所需迭代次数的后验期望值。

对于每个模型，我们使用所考虑的每种工具/算法，抽取 N粒子（$N \in R = \{10^3, 10^4, 10^5, 10^6\}$）及其对应的权重。利用抽取的样本和权重，这些工具会计算感兴趣量的（后验）期望值。对于每种工具，我们关注评估以下方面：

- 计算所得期望值的准确性；
- 抽取样本的质量；
- 执行时间方面的性能。

关于准确性，我们不知道目标期望值的确切值（但有一种情况例外，见下文），因此无法直接比较。尽管如此，正如第 6 节小节中所讨论的，粒子滤波和其他序贯蒙特卡罗采样算法具有渐近一致性，这保证了随着 $N \to +\infty$ 增大，样本估计值将收敛到真实期望值。对于特定的 N 值，我们没有明显的方法来判断结果离收敛还有多远。实际操作中，我们将所考察范围 R 内连续 N 值对应估计值之间的差异较小，以及不同工具给出的估计值彼此非常接近，视为收敛和准确性的经验证据。如上所述，与其他返回截断点估计值的工具不同，VPF 原则上可根据定理 4 给出 $[\![S]\!]f$ 的上下界。只要目标函数 f 无界，上界就会是无效的（$+\infty$）；这里的 HT 和 NIID 正是如此。此外，示例 AT 和 BRP 都是有界循环，因此 $\alpha_t = 1$，进而 $\beta_L = \beta_U = [\![S]\!]f$。

我们通过相应权重 W_1，…，W_N的有效样本量（*ESS*）[30] 来实证衡量抽取样本的质量：

$$ESS := \frac{(\sum_{i=1}^{N} W_i)^2}{\sum_{i=1}^{N} W_i^2}.$$

ESS 是衡量所采样粒子效率的经验指标，数值越高越好。具体而言，ESS 表示从目标分布中抽取多少个独立同分布样本，才能使估计量的方差与加权样本所得的方差相同。因此，ESS 接近 N 表明这些 N 粒子似乎是从目标分布中独立同分布抽取的。

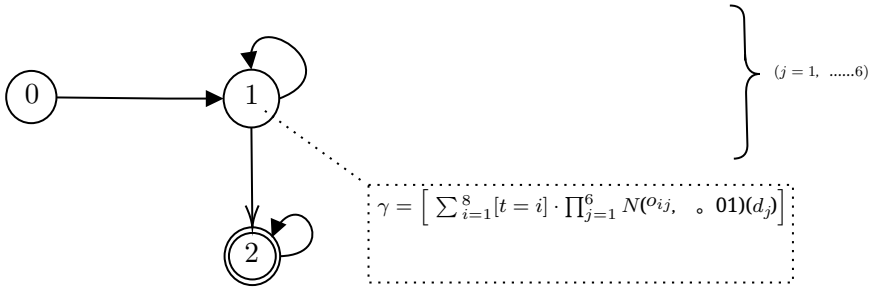
实验在一台配备 16GB RAM 和 Nvidia T500 GPU 的 2.8 GHz 英特尔酷睿 i7 PC 上运行。VPF 和 webPPL 在 Windows 10 操作系统下运行，使用 CUDA 工具包 v. 11.8 和驱动程序 v. 522.06。CorePPL 和 RootPPL 在 Ubuntu 22 下运行，使用 CUDA 工具包 v. 12.2 和驱动程序 v. 535.86。

在表 1（第 7 节）中，我们报告了随着粒子数量 N 增加，VPF、CorePPL、RootPPL 和 webPPL 的执行时间、期望值估计值以及有效样本量。对于 VPF，除 DMM 外的所有示例均报告了单个值估计：对于程序 AT 和 BRP，该估计值为 $[\![S]\!]f$；对于示例 HT 和 NIID（无界 f），该估计值为下界 β_L，因为如上所述，上界是空泛的。我们还指出，对于 NIID，已知 $[\![S]\!]f = \frac{24}{7} = 3.428\cdots$[40]。我们注意到，至少对于 $N \geq 10^5$，这些工具倾向于给出非常相似的

期望值[11], ，这是粒子滤波和其他序贯蒙特卡洛算法渐近一致性的结果；我们将此视为准确性的实证。

我们以对飞机跟踪（AT）的概率程序图（PPG）进行明确描述，作为本节的结束。飞机被建模为一个点，按照高斯过程在二维平面上移动，时间离散为 $t = 1$，.....，8 个时刻。在这些时刻，六个雷达会跟踪飞机。每个雷达由一个半径表征：在每个时刻，如果飞机处于雷达半径范围内，雷达就会返回测得的飞机距离（带有噪声）；否则，雷达只会返回自身半径的含噪版本。我们的目标是根据从六个雷达在全部八个时刻获得的实际观测数据，推断飞机的最终水平位置，即时刻 $t = 8$ 的 x 值。

在下方的概率程序图中，o_{ij} 是时刻 i 雷达 j 测得的距离，其中 $1 \leq i \leq 8$ 和 $1 \leq j \leq 6$ 分别表示.....；($^{r}x_j$, $^{r}y_j$) 和 $^{r}_j$ 分别是雷达 j 的坐标和半径。实际数值数据见 [59]。此外， $B(p)$ 是参数为 p 的伯努利分布，而 $N_T(a, b, c, d)$ 表示均值为 c、标准差为 d 且截断于 $[a, b]$ 的正态密度； $N_T(a, b, c, d)(z)$ 是该密度在 z 处的值。



最后，我们报告了图2中各图的放大版本。

附录 D. 概率程序伪代码

我们考虑以下概率模型，例如文献 [34]中描述的模型。为方便起见，这些程序采用基于顺序组合的 [34], 语言编写，但很容易将它们转换为我们的语言。每个程序末尾的 return 语句描述了在估计 $[\![S]\!]f$时考虑的函数 f；例如，在 DMM 示例中，return r 表示 f是函数 $(d, r, x, y, S) \mapsto r$的提升。

[11]一个例外 是由 CorePPL 针对 NIID 示例返回的结果表示的。结果另外三　　种工具彼此一致，也与精确值 $\frac{24}{7} = 3.428\cdots$ 一致。　　—

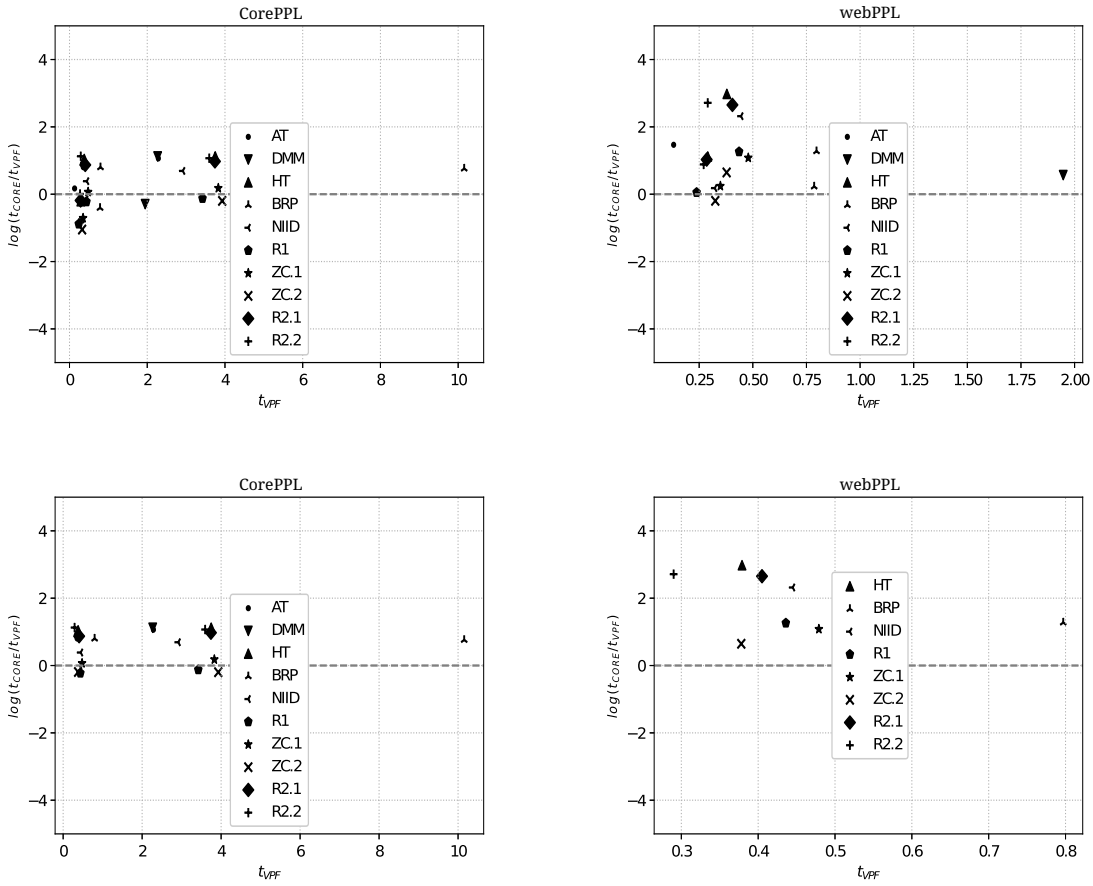


图4。图2中图表的放大版。

```
1: p=uniform(0,1); 2: start=1; 3: curprobe,established=0; 4:
while(curprobe < 100 && established <=0 && start <= 1){ 5:
if(start = 1){ 6: flag=伯努利分布(p) 7: if (flag==false){ 8: established=1;
9: } 10: start=0; 11: }else{ 12: flag=伯努利分布(lambda) 13: if (flag==
true){ 14: curprobe :=curprobe + 1; 15: }else{ 16:
observe(curprobe>=20); 17: start=1; 18: curprobe=0; 19: } 20: } 21: }
22: return p;
```

(g) ZeroConf（ZC.1： $\lambda = 0.99$, ZC.2： $\lambda = 0.5$）。

```
1: var=均匀分布(0,7); 2: y=1; 3: prob=0.5;4: i=0;5:while(i <= 100)
{6: oldy=y;7: y=高斯分布(oldy,2*var); 8: flag=伯努利分布
(lambda); 9: if(flag){ 10: observe(|y-oldy|<2)};11: i=i+1;}
12:return y;13:14:15:16:17:18:19:20:21:
```

(h) 随机游走 (RW2.1： $\lambda = 0.5$, RW2.2:$\lambda = 0.9999$)。

```
1: while(time<=8){ 2: float[]radius; 3: float obs-dist; 4: if(ti
me==0){ 5: x  =高斯分布(2,1); 6: y  =高斯分布(-1.5,1); 7: }else{
8: x  =高斯分布(x,2); 9: y  =高斯分布(y,2);} 10: for i in (0,6){
11: d= 计算距离(i,x,y); 12: if(d>radius[i]){ 13: flag=伯努利分布
(0.999); 14: if (flag==true){ 15: obs-dist=radius[i]; 16: }else{
17: obs-dist=radius[i]+0.001* 18: 截断高斯分布(0,radius[i]);
19: } 20: }else{ 21: obs-dist=d+0.1*截断高斯分布(0,radius[i]);
22: } 23: obs-dist1 = 高斯分布(obs-dist,0.01); 24:
observe(obs-dist1==...); //证据编号 25: 省略 26: } 27: time=ti
me+1; 28: } 29:return x
```

(a) 飞机跟踪 (飞机跟踪)。

```
1: d=均匀分布(0,2);
2: r=均匀分布(0,1);
3: x=-1;
4: y=1;
5: 当(|x-y|<1/10){
6:         x=高斯分布(x,d);
7:         y=高斯分布(y,r);
8:         观测(|x-y|<=3);
9: }
10: return r;
11:
12:
13:
14:
15:
16:
17:
18:
19:
20:
21:
22:
23:
24:
25:
26:
27:
28:
29:
```

(b) 醉汉与老鼠（DMM）。

```
1: 初始位置=均匀分布(0,10); 2: 乌龟=初始位置; 3: 野兔=0;4:
n=0; 5: while(野兔<乌龟){ 6: n=n+1; 7: 乌龟=乌龟+18:
flag=伯努利分布 (2/5); 9: if (flag==true){ 10: 野兔=野兔+高斯
分布(4,2); 11: } 12: observe(野兔-乌龟|<=10);13: }14:
observe((n>=20));15:return hare;16:17:18:
```

(c) 兔子与乌龟（HT）。

```
1: 初始位置=均匀分布(0,10); 2: s=100;
3: f=0; 4: t=0; 5: n=0; 6:while(s>=0  &&
f<=4  && t<=280){ 7: t=t+1; 8: flag= 伯
努利分布(0.2); 9: if (flag==1){ 10: f=f+1;
11: n=n+1; 12: observe((s<=80)); 13: }
else{ 14: f=0; 15: s=s-1; 16: } 17: } 18:
return s>0;
```

(d) 有界重传协议（BRP）。

```
1: a0=1;2: b0=1;3: c0=1;4: d0=1;5: n=0;6: while((a
0==1||b0==1)){7:       a1=伯努利分布(0.5); 8: b1=伯努利分
布(0.5); 9: 观测(c0==a1 ||d0==b1);10:       c1=a1;11:
d1=b1;12:       n=n+1; 13: } 14:返回 n
```

(e) 非独立同分布循环（NIID）。

```
1: r=均匀分布(0,1); 2: y=0;3: n=0; 4:当(|y|<1 && (n<=100))
{5:       y=高斯分布(y,2r); 6: n=n+1; 7: } 8: 观测(n>=3); 9:
返回 r; 10: 11: 12: 13: 14:
```

(f) 随机游走（RW1）。

本作品采用知识共享署名许可协议授权。如需查看该许可协议副本，请访问 https://creativecommons.org/licenses/by/4.0/，或致函知识共享，地址：美国加利福尼亚州旧金山市第二街171号300室，邮编94105；或德国柏林10777，艾森纳赫大街2号。