带安全约束的反应式综合规范学习

期刊标题 XX(X)：1–19©作者 2025。转载及许可事宜：
sagepub.co.uk/journalsPermissions.nav DOI：
10.1177/ToBeAssigned
www.sagepub.com/SAGE

Kandai Watanabe[1],Nicholas Renninger[2], Sriram Sankaranarayanan[3] 和 Morteza Lahijanian[3]

摘要
本文提出了一种新颖的从示范中学习方法，使机器人能够在动态环境中自主执行复杂任务。我们将潜在任务建模为概率形式语言，并提出一个量身定制的反应式综合框架，在机器人代价与用户任务偏好之间进行权衡。我们的方法侧重于安全约束学习，并将形式化任务规范推断为概率确定有限自动机（PDFA）。我们调整了现有的“证据驱动的状态合并”算法，并在整个学习过程中纳入安全性要求，以确保学习得到的 PDFA 始终符合安全约束。此外，我们提出了一种多目标反应式综合算法，可生成确定性策略，保证满足 PDFA 任务，同时优化用户偏好与机器人代价之间的权衡，从而得到最优解的帕累托前沿。我们将交互建模为机器人与环境之间的双人博弈，并考虑环境的动态变化。我们提出了一种计算上可行的值迭代算法，用于生成帕累托前沿及相应的确定性策略。全面的实验结果展示了我们的算法在各种机器人和任务中的有效性：学习得到的 PDFA 从不包含不安全行为，而综合策略始终能够完成任务，同时满足机器人代价和用户偏好方面的要求。

关键词
形式化方法、规范学习、反应式综合

arXiv:2601.05533v1 [cs.RO] 9 Jan 2026

1 引言

技术进步使机器人能够在人类共享领域中以日益增强的自主性运行。应用实例涵盖家用辅助机器人、装配线，以及深海和行星探索。在这些环境中，机器人必须做出决策，以便在复杂任务 和安全性要求的严格约束下，应对多样且动态的情况并实现目标。然而，复杂的任务规范往往不可获得，或者对于非专家而言过于困难，难以提供。相反，人类可以通过操作或过往数据示范任务。随后，机器人必须推断任务目标并自主执行任务。这一过程面临五项挑战：(i) 确定一种能够从示范中高效且精确地学习的形式化方法；(ii) 确保所学习的规范满足安全属性；(iii) 捕捉操作者偏好或隐含代价；(iv) 将规范应用于新的环境；以及 (v) 通过反应性确保任务完成。在本文中，我们旨在借助形式化方法应对这些挑战，开发一种规范学习方案和一种反应式策略综合算法，使二者能够有效互补。

例如，如图所示，设想一台被部署用于深海科学探索的水下机器人1。科学家希望机器人调查海底的一艘沉船，观察鱼群的行为，并避开珊瑚礁，以防造成损害。与其要求机器人专家和领域专家远程操控机器人，或一丝不苟地制定任务目标和执行计划，不如本研究旨在仅通过向机器人提供以往部署中类似任务的数据（示范），使其能够自主执行这项任务。机器人应能从此类数据中推断出稳健的任务表示，并生成必要的策略，即使在动态变化的环境中也能完成任务。我们将这一问题称为从示范中学习规范，可将其视为一种新型的从演示中学习(LfD)Ravichandar 等人(2020)。

现有的大多数从演示中学习和反应式规划方法，都侧重于学习奖励结构或策略 Ravichandar 等人 (2020); Hussein 等人 (2017年)。这些方法通常会学习一个针对训练期间所用环境和机器人模型的特定函数，因此当这些模型发生变化时，方法就会变得脆弱。然而，在许多现实场景中，示范是在不同于执行环境的场景中进行的。此外，这些方法仅适用于马尔可夫任务：在这类任务中，当前状态下的决策只取决于当前情况，而不受过去事件的影响。

[1]Google LLC[2]MITRE Corporation[3]科罗拉多大学博尔德分校

通讯作者：Kandai Watanabe，Google LLC，901 Cherry Avenue，圣布鲁诺，CA 电子邮箱：kandai.watanabe@colorado.edu

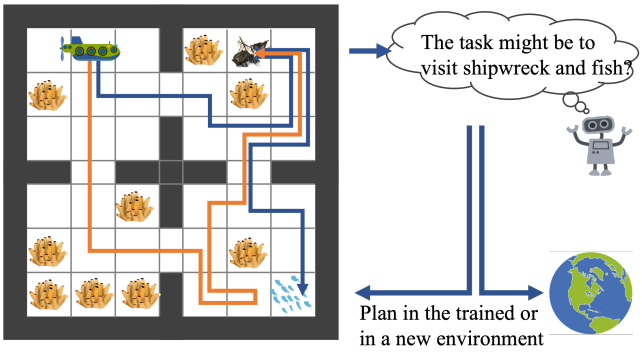


图1。 自主深海科学任务示意图。绿色水下车辆的任务是在避开珊瑚礁（黄色）的同时，访问鱼群（蓝色）和沉船（棕色）。我们的目标是从示范（彩色路径）中推断潜在任务，并在训练/新环境中综合出一个能够完成学习到的任务的控制器。

但是，复杂任务通常需要保留过去事件的历史记录，才能成功完成。例如，图1 中的机器人任务要求按任意顺序前往沉船和鱼群，因此它是非马尔可夫任务。在这类任务中，机器人必须跟踪先前的位置，以决定下一时刻该前往何处。

在这项工作中，我们提出了一种新的从演示中学习方法，将任务视为概率形式语言，并引入一种反应式综合框架，在机器人运行成本与用户对任务完成方式的偏好之间实现最优权衡。我们将形式化任务规范推断为概率自动机，并借鉴了语法推断 （GI）领域De laHiguera （2010）的见解。将任务表示为自动机具有多项优势：（i）可解释性；（ii）符号推理能力；以及（iii）能够使用经过充分研究的算法操作技术。我们专注于安全属性约束学习，即最终学习到的规范必须满足安全属性。与众不同的是，我们提出的学习技术在整个学习过程中纳入安全约束，而不是事后再应用这些约束。具体而言，我们改造了语法推断领域中已有的证据驱动状态合并（EDSM）算法，将任务规范学习为概率确定有限自动机 （PDFA）De la Higuera （2010）。我们将安全属性整合到学习过程中，确保所得PDFA的所有执行轨迹都满足安全性要求。

给定推断出的概率确定有限自动机，我们提出了一种反应式综合算法，用于生成确定性策略，在完成任务的同时，最大化示范者偏好并最小化机器人的运行成本。然而，这些目标往往彼此冲突，因而需要权衡。这会产生多个最优解，统称为帕累托前沿，其中每个解都体现了任务偏好与成本最小化之间的不同平衡Chen 等人(2013a）。我们提出了一种计算效率高的算法，用于计算帕累托前沿，并为每个帕累托点推导策略。该方法将问题建模为机器人（系统）与环境之间的双人博弈，并将环境视为对抗性智能体来应对动态变化。我们在各种机器人和任务上进行测试，实验结果表明，学习得到的概率确定有限自动机始终能够避免不安全行为，而综合出的策略始终满足任务要求。此外，机器人的成本和任务偏好均保持在相应帕累托点所预测的范围内。

本文大幅扩展了会议版本中关于计划综合的部分Watanabe 等人(2021）。具体而言，Watanabe 等人(2021) 假设环境是静态的，并在图上综合一条路径，以优化所学习的概率确定有限自动机的偏好度量。本研究通过考虑动态环境，并综合能够保证完成学习到的任务、同时优化任务偏好度量与机器人动作代价之间权衡的反应式策略，对 Watanabe 等人 (2021) 的工作进行了推广。这将原问题从图搜索显著转变为机器人与环境之间的多目标博弈。新增内容包括针对此类博弈的一种新颖分析方法和综合算法，适用于确定性策略，并附有正确性与完备性的证明以及实验评估。

总体而言，本研究的贡献有四个方面。

- 推导出一种具有安全性保证的概率确定有限自动机学习算法，该算法兼容任何 EDSM 技术，
- 提出一种多目标反应式综合算法，利用所学习的概率确定有限自动机处理动态环境，
- 一种用于计算确定性策略帕累托前沿的值迭代方法，并提供完备性证明；
- 一套全面的实验，展示了所提出的算法在移动机器人和机械臂机器人应用中的有效性。

2 相关工作

规范学习：许多从演示中学习（LfD）研究旨在学习策略或奖励函数。对于策略学习，通常采用强化学习（Sutton 和 Barto（2018年）和动态运动基元 Schaal （2006）；Paraschos 等人 （2013年）等技术，来学习将智能体状态映射为动作的函数。在奖励学习中，例如通过逆强化学习 （IRL）Ng 和 Russell （2000年）；Ziebart 等人 （2008）；Wulfmeier 等人（2015）；Ramachandran 和 Amir（2007）等方法，学习将智能体状态映射为奖励的标量奖励函数，并在智能体上同时训练策略。如上所述，这些方法容易受到环境变化的影响，因为它们学习的是特定于训练期间所用环境模型的函数。

表达任务的另一种方法是使用形式语言，例如 线性时序逻辑(线性时序逻辑)Baier 和 Katoen (2008)，它广泛用于形式化验证，近年来也日益应用于机器人学，例如Kress-Gazit 等人(2018)。这类语言能够对丰富的任务进行形式化表达，包括非马尔可夫任务Vazquez-Chanlatte 等人 (2018)，以及活性 （“最终会有好事发生”）和

安全性 （“坏事永远不会发生”）要求。形式语言的其他重要优势在于其易于理解，并且能够灵活地组合多种规范。这些优势甚至促使形式语言应用于强化学习，例如 Camacho 等人 (2019b); Li 等人 (2017年, 2019年)。然而，编写正确的形式化规范需要领域知识。

近年来，出现了一个新的研究方向，专注于从数据中学习形式化规范 Vazquez-Chanlatte 等（2018年、2017年）；Xu 等人（2018年）；Jha 等人（2017年）；Shah 等人（2018年）。大多数研究致力于学习时序逻辑公式，以便从用户数据中进行分类和预测（即监督学习意义上的分类和预测）Xu 等人（2018年）；Jha 等人（2017年），或用于任务的解释与规划 Shah 等人（2018年）。这些研究将探索问题限制在事先给定的一组公式模板之内。近期研究 Vazquez-Chanlatte 等（2018年）通过遍历公式的所有组合，克服了这一限制。该方法基于最大后验学习，并能处理含噪样本。然而，由于公式的探索空间过大，该方法速度较慢。用于规划的公式学习方法还存在另一个重要问题：通常需要将公式转换为自动机，这可能导致状态爆炸问题 Baier 和 Katoen（2008年）；Kress-Gazit 等人（2018年）。Araki 等人（2019年）的研究通过直接学习确定有限自动机（确定有限自动机）解决了这一问题。不过，他们假设确定有限自动机的结构已知，只学习确定有限自动机状态之间的转移，同时由预言机为每个样本标注确定有限自动机状态。

综合： 利用 LTL 的规划算法已得到广泛研究 Kress-Gazit 等人（2018年），本研究在这些已开发的方法基础上展开。常见方法包括针对离散状态的基于自动机的技术 Kress-Gazit 等人（2018年）、基于采样的运动规划 Bhatia 等人（2010），以及针对连续状态的强化学习 Camacho 等人 （2019a）。这些方法已扩展到在反应式环境中综合解Fainekos 等人 （2005）。在本研究中，我们将动态环境建模为系统玩家与环境玩家之间的博弈，这与He 等人（2019a、2017a）中提出的框架类似； Muvvala 等人 （2022年）； Muvvala 和 Lahijanian（2023年）。然而，我们的方法必须考虑多个定量目标，这些目标源于所学习 PDFA 中的概率和机器人的运行成本，同时还要满足任务完成的可达性要求。这自然会导向多目标可达性博弈，其目标是综合出一种策略，在满足可达性要求的同时，对各项定量目标进行最优权衡。

在Chen 等人（2013b）； Basset 等人（2015）； Chen 等人（2013a）探讨了多目标随机博弈，研究了停止博弈和非停止博弈。对于停止博弈，他们证明了确定性策略要么需要无限记忆，要么必须采用随机化。相比之下，Chatterjee 等人（2012）提出了一种用于多目标（多能量、平均收益和奇偶）非停止博弈的综合算法，证明在多维能量奇偶博弈中，指数级记忆足以满足要求，并引入了一种符号化算法来计算有限记忆获胜策略。然而，我们的方法通过值迭代，为多目标停止博弈综合确定性策略，这与 Chen 等人（2013b）的工作更为接近。此外，Sastry 等人（2005）研究了类似问题，但重点是通过将多个目标转换为单一目标，并将问题转化为最短路径表述，来寻找一组帕累托最优解。相比之下，我们的方法计算出确定性策略所能实现的整个帕累托前沿。

3 预备知识

在本研究中，我们关注如何将机器人部署在动态环境中，以收集用于任务学习和策略综合的示范。为定义这一问题，我们首先介绍动态环境建模、示范、任务规范和策略所需的背景知识。定义所有术语后，我们将在第 4 节正式介绍这一问题。

3.1 双人博弈：机器人—环境交互模型

我们考虑一个需要与动态环境交互以完成任务的机器人。例如，图 1 中的机器人为了实现目标，需要与能够自由游动的鱼群互动。这种互动可以建模为机器人与环境（鱼群）之间的博弈，其中每个玩家都有各自的目标和动作集合。实际上，这种博弈发生在连续域中，而且可能是并发的，但我们可以通过抽象将其表示为离散双人博弈。这类抽象常用于移动机器人学和机械臂领域的形式化方法中Kress-Gazit 等人 (2018,2007);Lahijanian 等人 (2009)以及机械臂领域He 等人 (2015, 2019a); Muvvala 等人(2024年)。

定义 1. 双人博弈。 双人博弈是一个元组 $G = (S, A, s_0, \delta, \Pi, L, W)$，其中

- $S = S_R \cup S_E$ 是一个有限状态集，其中 S_R 和S_E 分别是机器人状态集和环境状态集，并且 $S_R \cap S_E = \emptyset$,
- $A = A_R \cup A_E$ 是一个有限控制动作或动作集，其中 A_R 和 A_E 分别是机器人动作集和环境动作集，
- $s_0 \in S$ 是初始状态，
- δ： $S \times A \to S$ 是转移函数，
- Π 是一个有限原子命题（谓词）集，
- L： $S \to 2^{\Pi}$ 是一个标记函数，它将每个状态映射到该状态下为真的谓词集合，并且
- W： $S \times A \to \mathbb{R}^m_{\geq 0}$ 是一个权重函数，它为每个 $(s, a) \in S \times A$ 分配一个 m 维的非负权重向量 $W(s, a)$。

博弈 G也被称为多目标 双人博弈，因为它允许通过 W 为每条边编码多个权重，即 m 权重，从而表示 m 目标。

例如，这些权重可以表示能量和距离成本。

博弈的演化过程如下。在状态 $s \in S_i$，玩家 $i \in \{\mathrm{R}, \mathrm{E}\}$ 选择动作 $a \in A_i$，并获得权重 $W(s, a)$。随后，博弈状态转移至 $s' = \delta(s, a) \in S_j$，$j \in \{\mathrm{R}, \mathrm{E}\} \setminus \{i\}$。接下来轮到玩家 j采取动作，之后重复这一过程。

示例 *1.* 双人博弈。图*1*所示的博弈抽象定义如下。状态是由车辆位置 l_R、鱼的位置 l_E和玩家的回合 $i \in \{R, E\}$组成的元组，即 $s = (l_R, l_E, i)$，其中 $l = (x, y)$ 是每个智能体的坐标。从初始状态$s_0 = ((2, 1), (7, 7), R)$开始，车辆可以采取动作 *N*（北）、*S*（南）、*E*（东）或 *W*（西），消耗 *2* 的能耗成本，转移到距离为 *1* 的相邻单元格，即对于所有 $a \in \{N, S, E, W\}$，$W(s, a) = (1, 2)$。鱼也可以采取类似的动作。在此环境中可观测到的所有可能原子谓词集合为 Π ={沉船、鱼、珊瑚礁}。当车辆和鱼都位于 $l_R = l_E = (7,7)$ 时，观测结果为 $L((l_R, l_E, i)) = \{fish\}$，其中 $i \in \{R, E\}$。

玩家轮流采取动作*，这一演化过程会产生一系列状态，称为一局；一局会生成一系列观测，称为轨迹（也称为词）。

定义 2. 对局与轨迹。对局 $\mathcal{S} = s_0 s_1 \ldots s_n$ 是一个从初始状态 s_0开始的状态序列；对于所有步数 $0 \le k < n$，都存在一个动作 $a_k \in A$，使状态转移到下一时刻的状态 $s_{k+1} = \delta(s_k, a_k)$。$G$ 中所有对局的集合记为 Play。$\mathcal{S}$的输出轨迹是状态标签序列 $\omega = L(\mathcal{S}) = L(s_0)L(s_1) \ldots L(s_n)$。

对局 $\mathcal{S}$ 在位置 $k \le n$ 处的前缀，是从 s_0 到第 k 个状态的有限状态序列 $\mathcal{S}(k) = s_0 s_1 \ldots s_k$。如果 $s_k \in S_{\mathrm{R}}$，我们称前缀 $\mathcal{S}(k)$ 属于机器人玩家；否则，它属于环境玩家。

示例 2. 对局与观测轨迹。从$s_0 = ((2,1), (7, 7))$开始，如果机器人依次执行动作 *S* 和 *W*，鱼执行动作 *S*，则产生的对局为 $\mathcal{S} = ((2, 1),(7, 7), R)\,((2, 2), (7, 7), E)\,((2, 2), (7, 7), R)((1, 2),(7, 7), E)$。这会产生观测轨迹 $\omega =$ ∅∅∅{珊瑚礁}。

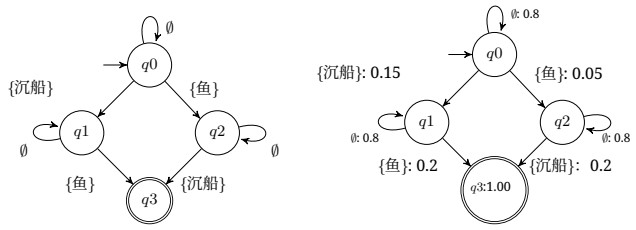
每个玩家在一场对局中获得的总成本称为总收益。

定义 3. 多目标总收益。对局 $\mathcal{S} = s_0 \ldots s_n$ 的多目标总收益是沿 $\mathcal{S}$各权重向量之和，即

$$\mathrm{TP}(\mathcal{S}) = \sum_{k=0}^{n-1} W(s_k, s_{k+1}).$$

在博弈 G中，每个玩家根据一种策略选择动作。动作选择通常可以是确定性的，也可以是随机性的。本文关注确定性策略，因为我们感兴趣的是机器人在一次部署中的行为，而非多次部署中的预期行为。

定义 4. 策略。令 $S^* S_i$ 表示所有以 $S_i \subseteq S$结尾的有限对局的集合，其中 $*$ 表示克莱尼星号，并且



(a) 示例确定有限自动机 **(b)** 概率确定有限自动机示例

图2。 自主深海科学任务的确定有限自动机和概率确定性有限自动机表示。

$i \in \{R,E\}$。玩家 $i \in \{R, E\}$ 的（确定性）策略是一个函数 $\tau_i : S^* S_i \to A_i$，它根据以 S_i 中的某个状态结尾的（有限）对局来选择下一时刻的动作。

在机器人和环境策略 τ_{R} 以及 τ_{E}下，博弈产生一个由Play($\tau_{\mathrm{R}}, \tau_{\mathrm{E}}$)表示的单一对局。然而，$\tau_{\mathrm{E}}$ 通常是未知的；因此，我们的目标是选择一个 τ_{R}，使其针对所有可能的环境策略实现机器人的目标，即在τ_{R}下由Play(τ_{R}, ·)表示的所有可能对局。

3.2 任务规范

我们假设机器人任务可以表示为一个具有字母表 2^{Π}的确定有限自动机（DFA）。

定义 5。 确定有限自动机。确定有限自动机（*DFA*）是一个元组 $\mathcal{A} = (Q, \Sigma, q_0, \delta_{\mathcal{A}}, F)$，其中

- Q是一个有限状态集，
- $\Sigma = 2^{\Pi}$ 是一个有限输入符号集，其中每个符号都是 Π的子集，
- $q_0 \in Q$是初始状态，
- $\delta_{\mathcal{A}}$： $Q \times \Sigma \to Q$ 是转移函数，并且
- $F \subseteq Q$ 是最终状态或接受状态的集合。

转移函数 $\delta_{\mathcal{A}}$ 也可以视为一个关系 $\delta_{\mathcal{A}} \subseteq Q \times \Sigma \times Q$，其中每个转移都是一个元组 (q, σ, q')，当且仅当 $q' = \delta_{\mathcal{A}}(q, \sigma) \in \delta_{\mathcal{A}}$，其中 $\sigma \in \Sigma$。

轨迹 $\omega = \omega_1 \omega_2 \ldots \omega_n$，其中对于所有 $1 \le i \le n$都有 $\omega_i \in 2^{\Pi}$，在确定有限自动机 $\mathcal{A}$ 上诱导出一个运行 $z = z_0 z_1 \ldots z_n$，其中 $z_0 = q_0$，并且对于 $i = 1$、……、n有 $z_i = \delta(z_{i-1}, \omega_i)$。如果 $z_n \in F$，则称运行 z为接受运行。如果轨迹 ω 诱导出一个接受运行，则称其被 $\mathcal{A}$ 接受。确定有限自动机 $\mathcal{A}$ 所接受的所有轨迹的集合称为 $\mathcal{A}$ 的语言，记作 $\mathcal{L}(\mathcal{A})$。在博弈 G中，如果其输出轨迹 $\omega \in \mathcal{L}(\mathcal{A})$，我们称对局 $\mathcal{S}$ 满足由 $\mathcal{A}$表示的任务。

示例3。 图*2a* 展示了一个确定有限自动机的示例，该自动机表示图*1*中的机器人任务。接受状态集合为 $F = \{q_3\}$。轨迹 $\omega = \emptyset${沉船}$\emptyset${鱼}在该确定有限自动机上诱导出接受运行 $q_0 q_0 q_1 q_1 q_3$。

*请注意，轮流行动仅发生在（离散）抽象中。现实中，智能体可能具有连续动力学，并且能够并发执行动作。抽象过程会将这些并发的连续交互映射为离散的轮流行动博弈，详见 He 等人（2017年b）；Muvvala 等人（2024年）。

确定有限自动机的概率扩展称为概率确定有限自动机，它为确定有限自动机的边De la Higuera（2010）赋予概率。因此，这会在确定有限自动机语言中的轨迹上诱导出一个概率测度。我们使用该测度作为接受轨迹的偏好度量。

定义 6. 概率确定有限自动机。 概率确定有限自动机（*PDFA*）是一个元组 $\mathcal{A}^{\mathbb{P}} = (\mathcal{A}, \delta_{\mathbb{P}}, F_{\mathbb{P}})$，其中 $\mathcal{A}$ 是一个确定有限自动机，且 $\delta_{\mathbb{P}}\colon Q \times \Sigma \times Q \to [0,1]$ 为 $\mathcal{A}$ 中的每个转移分配一个概率，使得对于每个 $q \in Q$，$\sum_{\sigma \in \Sigma} \delta_{\mathbb{P}}(q, \sigma, \delta_{\mathcal{A}}(q, \sigma)) = 1$，并且 $F_{\mathbb{P}}\colon Q \to [0,1]$ 为每个状态分配一个终止概率，使得若 $q \notin F$，则 $F_{\mathbb{P}}(q) = 0$。

考虑轨迹 $\omega = \omega_1 \omega_2 \ldots \omega_n$ 及其在概率确定有限自动机 $\mathcal{A}^{\mathbb{P}}$ 上的诱导运行 $z = z_0 z_1 \ldots z_n$。ω 的概率为

$$P(\omega) = \prod_{i=1}^{n} \delta_{\mathbb{P}}(z_{i-1}, \omega_i, z_i) \cdot F_{\mathbb{P}}(z_n).$$

我们称 $\mathcal{A}^{\mathbb{P}}$ 接受 ω，当且仅当 $P(\omega) > 0$。$\mathcal{A}^{\mathbb{P}}$ 的语言是概率非零的轨迹集合，即，

$$\mathcal{L}(\mathcal{A}^{\mathbb{P}}) = \{\omega \in (2^{\Pi})^* \mid P(\omega) > 0\}.$$

示例4。 概率确定有限自动机。 图*2b*展示了示例*3*中确定有限自动机的概率确定有限自动机扩展，其中在 q_3 处的终止概率为*1*，其他所有位置均为*0*。轨迹的概率

$$\omega = \emptyset\{\mathit{shipwreck}\}\emptyset\{\mathit{fish}\}$$

是 $P(\omega) = 0.8 \times 0.15 \times 0.8 \times 0.2 \times 1.0 = 0.0192$。

3.3 安全规范

为了表达机器人面临的安全约束，我们使用定义于原子命题集合Kupferman和Vardi(2001年提出的安全线性时序逻辑 Π。

定义7。 安全线性时序逻辑语法。 定义在 Π 上的语法安全线性时序逻辑公式递归定义如下

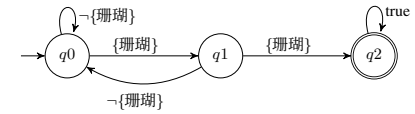
$$\varphi := \pi \mid \neg\pi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \mathcal{X}\varphi \mid \mathcal{G}\varphi$$

其中，$\pi \in \Pi$、$\neg$（否定）、$\vee$（析取）和 $\wedge$（合取）是布尔运算符；$\mathcal{X}$（“下一时刻”）和$\mathcal{G}$（“全局”）是时序算子。

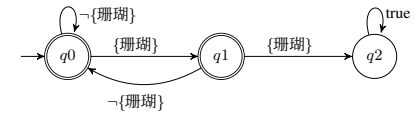
请注意，常用的蕴含运算符（$\rightarrow$）可以由 $\neg$ 和 $\vee$ 推导得出，即 $\varphi_1 \rightarrow \varphi_2 \equiv \neg\varphi_1 \vee \varphi_2$。

定义 8. 安全线性时序逻辑语义。 语法上安全的线性时序逻辑公式定义在 2^{Π} 上的无限轨迹上。设 $\omega = \omega_1 \omega_2 \ldots$ 是一条无限轨迹 $\omega \in (2^{\Pi})^{\omega}$，其符号为 $\omega_i \in 2^{\Pi}$，并定义 $\omega^i = \omega_i \omega_{i+1} \ldots$ 为 ω 的一个后缀。记号 $\omega \models \varphi$ 表示 ω 满足公式 φ，其归纳定义如下：

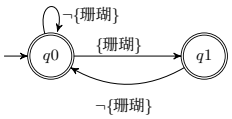
- $\omega \models \pi$ *iff* $\pi \in \omega_0$;
- $\omega \models \neg\pi$ *iff* $\pi \notin \omega_0$;
- $\omega \models \varphi_1 \vee \varphi_2$ *iff* $\omega \models \varphi_1$ *or* $\omega \models \varphi_2$;
- $\omega \models \varphi_1 \wedge \varphi_2$ *iff* $\omega \models \varphi_1$ *and* $\omega \models \varphi_2$;
- $\omega \models \mathcal{X}\varphi$ *iff* $\omega^1 \models \varphi$;
- $\omega \models \mathcal{G}\varphi$ *if* $\forall k \geq 0, \omega^k \models \varphi$.



(a) 安全性违规确定有限自动机 $\mathcal{A}_{\neg\varphi_{\text{safe}}}$



(b) 安全性确定有限自动机 $\neg\mathcal{A}_{\neg\varphi_{\text{safe}}}$



(c) 最小化安全确定有限自动机 $\mathcal{A}_{\varphi_{\text{safe}}}$

图3。 安全性违规确定有限自动机、安全性确定有限自动机和最小化安全确定有限自动机

安全线性时序逻辑公式针对无限轨迹进行推理，但有限轨迹已足以使其被违反 Kupferman 和 Vardi(2001年)。因此，只要有限轨迹没有违反安全线性时序逻辑，就满足安全约束。我们将违反安全公式 φ_{safe} 的$\mathcal{L}(\neg\varphi_{\text{safe}})$有限轨迹集合记为 φ_{safe}。根据安全线性时序逻辑公式 $\mathcal{A}_{\neg\varphi_{\text{safe}}}$，我们可以构造一个确定有限自动机 $\mathcal{L}(\neg\varphi_{\text{safe}})$，它接受所有违反公式的轨迹。通过反转接受条件并对该确定有限自动机进行最小化，我们可以构造一个安全性确定有限自动机 $\mathcal{A}_{\varphi_{\text{safe}}}$，其中每个状态都是接受状态（更多细节请参见Lahijanian 等人 (2016年)。

示例5。安全线性时序逻辑。 我们可以用一个安全线性时序逻辑公式来表达图 *1* 中机器人满足的安全性属性“始终在 *1* 步内逃离珊瑚礁”：$\varphi = \mathcal{G}$(珊瑚 $\rightarrow \mathcal{X}\neg$珊瑚)。我们首先构造图 $\mathcal{A}_{\varphi_{safe}}$ 中的安全性违规确定有限自动机 $\mathcal{A}_{\neg\varphi_{safe}}$（见图 *3a*），然后翻转其接受条件（图 *3b*），并将其最小化（图 *3c*），从而构造出相应的安全性确定有限自动机。

4 问题表述

在这项工作中，我们关注的是如何在动态环境中部署机器人，使其执行的任务并未明确指定，而是通过示范来呈现，例如过往的部署数据。我们的目标是让机器人推断任务，并根据用户偏好执行任务，同时确保任务完成并提供安全保证，即使环境发生变化也不例外。除了需要应对任务推断、偏好推断和动态环境带来的挑战外，我们也不要求示范必须在机器人将要部署的同一环境中采集。这种方式具有高度灵活性，允许独立于环境和机器人动力学来收集示范数据。下文首先介绍示范、偏好、安全约束和任务完成的概念，然后对问题进行正式表述。

我们假设演示者心中有一个任务 φ_{task}，该任务可以表示为字母表为 2^{Π} 的确定有限自动机。令 $\mathcal{A}_{\text{task}}$ 表示该确定有限自动机。基于 φ_{task}，演示者向机器人提供示范。我们将示范定义为一个能够完成 φ_{task} 的符号序列。

定义 9。示范。任务 φ_{task}的示范是一条轨迹 $\omega = \omega_1\omega_2\ldots\omega_n$，其中 $\omega_i \in 2^{\Pi}$ 对所有 $1 \le i \le n$ 都成立，且 $\omega \in \mathcal{L}(\mathcal{A}_{task})$。

给定一个有限示范集 $\Omega = \{\omega^1, \ldots, \omega^{n_\Omega}\}$，我们的目标是学习潜在任务，以及演示者对于如何完成该任务的偏好。例如，演示者可能偏好向左转向以避开障碍物，从而避免碰撞，因为这样可能使车辆处于能够以较小代价避开未来碰撞的位置。我们假设偏好的行为在 Ω 中出现得更频繁（重复次数更多）。我们的目标是学习任务~和以概率确定有限自动机（PDFA） $\mathcal{A}^{\mathbb{P}}_{\varphi_{\text{task}}}$形式表示的偏好，其中轨迹上的概率反映了偏好。

此外，我们指定一项安全属性 φ_{safe}，机器人不得违反该属性。对于我们的学习器而言，该安全属性刻画了一个可能无限的负示范集合，这些示范违反了安全属性。通过考虑安全约束，我们也可以避免问题的平凡解，例如接受所有可能行为的平凡规范。

现在，将机器人与环境的交互模型视为双人博弈 G，并为机器人指定动作成本。一旦学习到任务，我们便希望为机器人合成一种确定性策略，使机器人能够保证在一次运行中完成学习到的任务 $\mathcal{A}^{\mathbb{P}}$，同时最大限度地降低总成本（收益）并最大限度地提高偏好。问题的形式化表述如下。

问题1。给定一个由机器人与环境模型构成的双人博弈 G 以及一组示范$\Omega = \{\omega^i\}_{i=1}^{n_\Omega}$，这些示范根据某种偏好（概率分布）给出了一个（潜在）任务 φ_{task}，并给定一个安全线性时序逻辑公式 φ_{safe}，

1. 学习 φ_{task} 以及示范者偏好，将其表示为一个~概率确定有限自动机 $\mathcal{A}^{\mathbb{P}}_{\varphi_{task}}$，使其不接受任何违反安全性的轨迹~，即 $\mathcal{L}(\mathcal{A}^{\mathbb{P}}_{\varphi_{task}}) \cap \mathcal{L}(\neg\varphi_{safe}) = \emptyset$，并且

2. 计算一个（确定性）策略 τ_R^*，使得在该策略 τ_R^* 下的每一局都能完成 φ_{task} 且绝不违反φ_{safe}，同时使最坏情况下的总成本最小化并使最低偏好最大化，即对于 $\mathcal{S} = \text{Play}(\tau_R, \tau_E)$，

$$\tau_R^* = \arg\min_{\tau_R} \max_{\tau_E} \big(\text{TP}(\mathcal{S}), -P(L(\mathcal{S}))\big) \quad (1)$$

满足

$$L\big(\text{Play}(\tau_R, \tau_E)\big) \in \mathcal{L}(\tilde{\mathcal{A}}^{\mathbb{P}}_{\varphi_{task}}) \quad \forall\tau_E \quad (2)$$

$$L\big(\text{Play}(\tau_R, \tau_E)\big) \models \varphi_{safe} \quad \forall\tau_E \quad (3)$$

其中 $P(\cdot)$ 是概率确定有限自动机$\mathcal{A}^{\mathbb{P}}_{\varphi_{task}}$上的概率测度。

请注意，式（1）中的优化是多目标的，即$m+1$目标： m 维度属于 $\text{TP}(\mathcal{S})$，另一个属于$-P(L(\mathcal{S}))$。此外，这些目标相互竞争，也就是说，优化其中一个会使另一个变为次优。在这种情况下，我们关注的是这些目标之间的权衡。因此，我们的目标是优化目标之间的权衡，这称为帕累托最优。由于可能存在多个最优权衡，我们旨在计算确定性策略下所有可能的帕累托最优点。

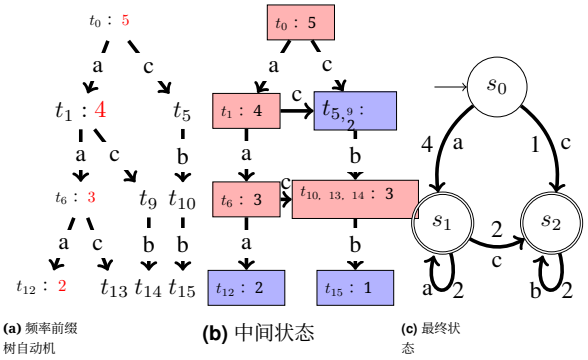


图4. 证据驱动状态合并（EDSM）算法的示意图。(a) 根据给定的示范构建频率前缀树接受器（FPTA），频率大于 1 的部分以红色显示；(b) 按照不同算法各自的准则合并状态，形成中间自动机，并在每个节点标出频率；(c) 学得的最终频率确定有限自动机（FDFA）以红色显示。

然后，选定一个方案后，我们综合出相应的策略。最后，约束（2）–（3）确保计算出的策略能够安全地完成学习到的任务，而不受环境策略的影响。

对于问题1.1，我们采用语法推断DelaHiguera（2010），并在学习过程中纳入安全属性，如第5节所述。我们使用该概率确定有限自动机（PDFA）解决问题1.2，详情见第6节。

5 安全保证PDFA学习

本节将说明如何从示范中学习概率确定有限自动机（PDFA），并介绍如何在学习过程中嵌入安全规范。我们首先展示一种通用的PDFA学习算法，然后介绍我们将安全性纳入其中的算法。

5.1 语法推断：概率确定有限自动机学习

概率确定有限自动机学习已作为语法推断（GI）的一部分得到广泛研究。现有算法如 ALERGIA、DSAI 和 MDI，能够从未标注的示范中学习概率确定有限自动机 Dela Higuera（2010）。这些算法都基于一种称为基于证据的状态合并（EDSM）的原理。从总体上看，EDSM 方法会为自动机找到合适的结构$\mathcal{A}^{\mathbb{P}}$，同时估计概率分布参数 $F_{\mathbb{P}}$ 和 $\delta_{\mathbb{P}}$，给定一组样本轨迹Ω。具体做法是，先根据样本构造一棵大型（前缀）树，再反复合并树的状态，形成尽可能紧凑的自动机，同时确保接受Ω中的示范轨迹。不同算法（例如 ALERGIA 和 MDI）的区别在于所采用的状态合并方法。

图 4 展示了一种基于 EDSM 的 PDFA 学习算法通用方案，算法1 给出了该算法的伪代码。初始步骤是根据 轨迹中的频率前缀树接受器 (FPTA) 构建 Ω(图 4a，以及算法 1 的第 1 行）。下一步是根据兼容性标准，每次合并两个 FPTA 状态，并逐步进行；具体采用何种标准取决于实际算法（第 兼容性 6和7 行）。合并两个状态时，

算法1： EDSM算法

输入： 一组示范轨迹Ω，以及合并操作
一致性参数 α
输出： 一个概率确定有限自动机

$\mathcal{A}' \leftarrow$BUILDFPTA(Ω)
$Q_{\text{red}} \leftarrow \{q_0\}$
$Q_{\text{blue}} \leftarrow$子节点$(q_0)$
当 $|Q_{blue}| > 0$ **时执行**
  $q_b \leftarrow$选择(Q_{blue})
  如果 $\exists q_r \in Q_{red}$ & COMPATIBLE$(\mathcal{A}', q_r, q_b, \alpha)$ **则**
    $\mathcal{A}' \leftarrow$ STOCHASTICMERGE$(\mathcal{A}', q_r, q_b)$
  else
    $Q_{\text{red}} \leftarrow Q_{\text{red}} \cup \{q_b\}$
  $Q_{\text{blue}} \leftarrow \cup_{q \in Q_{\text{red}}}$子节点$(q) \setminus Q_{\text{red}}$
返回FDFA2PDFA$(\mathcal{A}')$

它们在 FPTA 中对应的子树也会合并（图 4b）。中间自动机的节点会按照着色方案标记为红色或蓝色，以跟踪状态的合并选择。此外，算法还会根据到达特定节点的轨迹数量，在节点旁记录频率。在状态合并过程中，这些频率也会合并。最终结果是一个频率确定有限自动机 (FDFA)，其边上的频率表示示范中经过该边的次数（图 4c）。将所有出边的频率归一化，即可得到一个分布（第1行）。

各种概率确定有限自动机学习算法（如 ALERGIA 或 MDI）在实现兼容性检查时采用不同的方法，以判断给定的两个节点能否合并。例如，ALERGIA算法会基于频率进行统计检验，以比较两个状态是否兼容；而 MDI 方法则先临时合并两个状态及其子树，然后检查合并后自动机上计算出的度量是否小于合并前的度量。如果是，则接受合并；否则拒绝合并。我们的目标是在构建现有概率确定有限自动机学习算法的基础上，从 Ω 中学习一个概率确定有限自动机，同时满足安全属性 φ_{safe}，具体如下所述。

5.2 带安全规范的学习

我们现在考虑两种使用安全规范进行学习的不同方法。第一种方法是一种后处理技术：先对给定的示范轨迹运行概率确定有限自动机学习算法，然后再将所得概率确定有限自动机与安全属性对应的自动机求交。第二种方法则通过修改 EDSM 算法，在学习过程中纳入安全规范。具体而言，合并的定义确保结果继续满足安全规范。

5.2.1 后处理算法 从 φ_{safe}出发，我们首先构造一个确定有限自动机 $\mathcal{A}_{\neg\varphi_{\text{safe}}}$，它恰好接受所有违反安全属性的轨迹（Kupferman 和 Vardi（2001）；参见第3.3节）。然后，通过对 $\mathcal{A}_{\neg\varphi_{\text{safe}}}$取补，我们得到 $\mathcal{A}_{\text{safe}} = (Q^s, \Sigma, q_0^s, \delta^s, F^s)$，它接受所有未违反 φ_{safe}的轨迹。设 $\mathcal{A}^{\mathbb{P}} = (Q, \Sigma, q_0, \delta_A, F, \delta_{\mathbb{P}}, F_{\mathbb{P}})$为根据给定的示范轨迹学习得到的概率确定有限自动机不考虑安全属性。通过构造乘积自动机$\mathcal{A}^{\mathbb{P}}_{\text{safe}}$，我们对 $\mathcal{A}^{\mathbb{P}}$ 和 $\mathcal{A}_{\text{safe}}$ 的语言求交集。

定义10。乘积自动机。乘积自动机是一个元组 $\mathcal{A}^{\mathbb{P}}_{safe} = \mathcal{A}^{\mathbb{P}} \otimes \mathcal{A}_{safe} = (Q_{safe}, \Sigma, q_{0,\ safe}, \delta_{safe}, F_{safe}, \delta_{\mathbb{P},\ safe}, F_{\mathbb{P},\ safe})$，其中，

- $Q_{safe} = Q \times Q^s$,
- $q_{0,safe} = (q_0, q_0^s)$,
- $F_{safe} = F \times F^s$,
- $\delta_{safe}((q, q^s), \sigma) = (q', q^{s\prime}) \quad if \quad q' = \delta_{\mathcal{A}}(q, \sigma) \wedge q^{s\prime} = \delta^s(q^s, \sigma)$,
- $F_{\mathbb{P},safe}((q, q^s)) = F_{\mathbb{P}}(q)$,
- $\delta_{\mathbb{P},safe}((q, q^s), \sigma, (q', q^{s\prime})) =$

$$\begin{cases} \frac{\delta_{\mathbb{P}}(q,\sigma,q')}{N(q,q^s)} & if\ (q', q^{s\prime}) = \delta_{safe}((q, q^s), \sigma) \\ 0 & otherwise \end{cases} \tag{4}$$

其中，$N(q, q^s)$ 是归一化函数，使得$\sum_{(\sigma,\ q'_{safe}) \in (\Sigma \times Q_{safe})} \delta_{\mathbb{P},\ safe}((q, q^s)\ \sigma, q'_{safe}) = 1$。

由于语言交集，所得的PDFA能够保证安全。然而，这种将安全性作为后处理步骤施加的剪枝方法会改变下一状态转移上的概率分布，因为我们会移除违反安全性的转移，并在每个状态处对概率分布重新归一化，如（4）所示。这会以不可预测的方式覆盖原始PDFA学习算法构建的概率分布。因此，尽管这种施加安全性的方法总能成功，但其概率分布可能无法准确反映示范中所蕴含的偏好。

5.2.2 使用“预处理”的安全性融入学习算法与后处理 方法在学习 PDFA 后强制实施安全性不同，预处理方法保证中间结果始终保持安全性，从而避免因不安全性而改变概率分布。其核心思想是构建一个能够泛化示范轨迹的 PDFA，同时携带有关该泛化如何满足安全属性的信息 φ_{safe}，并以与 $\mathcal{A}_{\text{safe}}$ 的模拟关系形式呈现。

定义11。模拟关系。两个自动机 $\mathcal{A}$ 和 $\mathcal{B}$ 之间的模拟关系R 是其状态之间的一种关系，$R \subseteq Q_{\mathcal{A}} \times Q_{\mathcal{B}}$

(a) $\mathcal{A}$的初始状态与 $\mathcal{B}$的初始状态相关；*(b)* 如果对偶 $(s, t) \in R$，其中 $s \in Q_{\mathcal{A}}$ 且 $t \in Q_{\mathcal{B}}$，并且自动机 $\mathcal{A}$ 可以在符号 σ上从 s 转移到 $s' \in Q_{\mathcal{A}}$，那么必须存在一个状态 $t' \in Q_{\mathcal{B}}$，使自动机 $\mathcal{B}$ 在相同符号σ 上从 t 转移到 t'，并且 $(s', t') \in R$；*(c)* 对于每个 $(s, t) \in R$，如果 s 在 $\mathcal{A}$ 中是最终状态，则 t 必须在 $\mathcal{B}$中是最终状态。

定理1。设 R 是自动机 $\mathcal{A}$ 和 $\mathcal{B}$ 之间的模拟关系。由此可得 $\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{B})$。

证明。证明通过对字符串 $\omega \in \Sigma^*$进行归纳法，证明如果 $s \xrightarrow{\omega} s'$ 且（s，t）$\in R$，那么存在 t' 使得$t \xrightarrow{\omega} t'$ 且（s'，t'）$\in R$。根据模拟关系的定义，（s_0，t_0）$\in R$。对于 $\omega = \epsilon$（空字符串），$s = s_0$ 且 $t = t_0$。由于（s，t）$\in R$，显然（s'，t'）$\in R$。现在，假设该命题对字符串 ω成立。考虑字符串 ωa，其中 $a \in \Sigma$。设转移为 $s \xrightarrow{\omega} s' \xrightarrow{a} s''$ 和 $t \xrightarrow{\omega} t' \xrightarrow{a} t''$。根据归纳假设，（$s'$，$t'$）$\in R$。根据模拟关系的定义，（$s''$，$t''$）$\in R$。

如果 $\omega \in \mathcal{L}(A)$，则存在 $s' \in F_A$ 使得 $s \xrightarrow{\omega} s'$。根据模拟关系，存在$t' \in F_B$ 使得 $t \xrightarrow{\omega} t'$。因此，$\omega \in \mathcal{L}(B)$。所以，$\mathcal{L}(A) \subseteq \mathcal{L}(B)$。

证明只需通过归纳法说明：对于从初始状态到最终状态、对应于自动机$\mathcal{A}$ 中输入轨迹 ω的任意接受运行，在 $\mathcal{B}$ 中都存在一条针对同一轨迹 ω 、从其初始状态到最终状态的接受运行。关系 R 使我们能够构造这样的运行。

这种预处理 方法背后的关键思想，是在所有中间状态处维持 FDFA 与安全自动机 $\mathcal{A}_{\text{safe}}$ 之间的模拟关系。关键在于限制状态的合并方式，从而确保合并前原始自动机与 $\mathcal{A}_{\text{safe}}$之间的模拟关系可以经过调整，得到合并后合并后的自动机与 $\mathcal{A}_{\text{safe}}$ 之间的模拟关系。

形式化地说，我们通过扩充初始 FPTA 来构建安全 FPTA，使得每个状态现在都是形如 (t_j，s_k) 的元组，其中 t_j 是原始 FPTA 中的一个节点，而 s_k 是在将通向状态 t_j 的前缀输入 $\mathcal{A}_{\text{safe}}$ 后到达的状态。因此，我们确保每个分支不仅对应一个示范，也对应 $\mathcal{A}_{\text{safe}}$ 中的一条有效轨迹。

令 R 为 FPTA 与 $\mathcal{A}_{\text{safe}}$ 的状态之间的关系，其中包含安全 FPTA 中的所有节点 (t_j，s_k)。

引理 1。假设没有示范轨迹违反安全属性 φ_{safe}，那么 R 是初始 *FPTA* 与自动机 $\mathcal{A}_{safe}$ 之间的模拟关系。

证明。根据定义，正轨迹（位于初始 FPTA 中）必须能被自动机 $\mathcal{A}_{\text{safe}}$ 模拟。因此，FPTA 与 $\mathcal{A}_{\text{safe}}$ 之间存在模拟关系。

我们可以将任意中间 FDFA 状态表示为形如 (T，s) 的形式，其中 T 是初始 FPTA 中的一组状态，而 s 是 $\mathcal{A}_{\text{safe}}$ 中的状态。接下来，我们修改 EDSM 方法，使得仅当 $s_k = s_l$ 时，才允许合并两个状态 (T_i，s_k) 和 (T_j，s_l)。合并后得到状态 ($T_i \cup T_j$，s_k)。

引理2. 设 $\mathcal{A}_1$ 和 $\mathcal{A}_2$ 分别为 *EDSM* 合并前后的自动机，该合并关于 $\mathcal{A}_{safe}$状态是兼容的。设 R_1 为 $\mathcal{A}_1$ 的状态与 $\mathcal{A}_{safe}$ 的状态之间的模拟关系。我们可以构造 $\mathcal{A}_2$ 和 $\mathcal{A}_{safe}$ 的状态之间的模拟关系 R_2 。

证明。假设 $\mathcal{A}_1$ 和 $\mathcal{A}_{\text{safe}}$ 存在关系 R_1。根据定理1，$\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_{\text{safe}})$。$\mathcal{A}_1$中的任何转移$((s_{k-1} \to s_k)$、$(s_k \to s_k)$和$(s_k \to s_{k+1}))$ 都可以在 $\mathcal{A}_{\text{safe}}$中模拟。合并具有相同安全状态的两个节点时，始终会保留这些映射。因此，$\mathcal{A}_2$ 中的任何转移都可以在 $\mathcal{A}_{\text{safe}}$中模拟。因此，$R_2$ 是一个模拟关系。

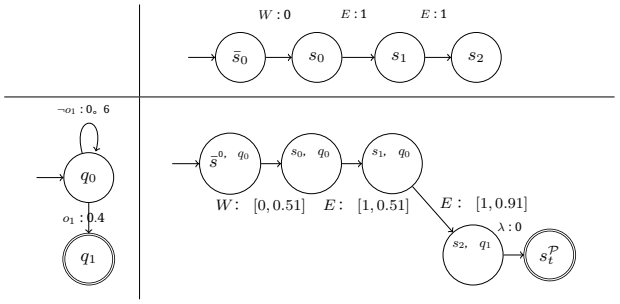


图5. 博弈乘积构造示意图。左侧的~自动机表示一个概率确定有限自动机 $\mathcal{A}^{\mathbb{P}}$，顶部的图表示一个增广博弈图 $\bar{G}$。中间的博弈乘积通过对两者进行乘积构造而得到。E 表示动作“东”，λ 表示随机动作，而 o_1 是在$s_2^{\mathcal{P}}$处满足 $\tilde{\mathcal{A}}^{\mathbb{P}}$中 q_0 与 q_1 之间守卫条件的观测。

结合引理1 和2，根据合并步骤的数量进行归纳，我们可以得出结论：最终得到的概率确定有限自动机必定与安全自动机存在模拟关系$\mathcal{A}_{\text{safe}}$。由于存在模拟关系，我们可以得出结论：最终得到的 FDFA 和概率确定有限自动机的语言都包含于 $\mathcal{A}_{\text{safe}}$ 的语言中，即所得概率确定有限自动机不会接受违反 φ_{safe}的轨迹。

6 基于概率确定有限自动机的反应式策略综合

~一旦将任务学习为概率确定有限自动机 $\mathcal{A}^{\mathbb{P}}$，我们便希望综合出一种策略，以便在动态环境中完成学习到的任务。首先，我们将反应式合成问题归约为可达性博弈问题。为此，~我们对博弈 G和概率确定有限自动机 $\mathcal{A}^{\mathbb{P}}$ 进行乘积构造，得到一个乘积博弈，其中涵盖了所有能够在 G中完成该任务的可能对局。随后，策略合成问题便转化为一个定量（多目标）可达性博弈，以确保所有对局都能到达接受状态，并获得（帕累托最优）收益。

6.1 乘积构造

首先，我们用一个新的初始状态$\bar{s}_0$ 和一条指向原始初始状态 s_0的转移来增广博弈图 $\bar{G}$。形式化地，$\bar{G} = (S, A, \bar{s}_0, \delta, \Pi, L)$，其中 $S = S \cup \{\bar{s}_0\}$，并且 $\bar{\delta}(s, a) = s_0$ 当且仅当 $s = \bar{s}_0$，否则 $\bar{\delta}(s, a) = \delta(s, a)$ $\forall a \in A$。该增广确保在与 $\mathcal{A}^{\mathbb{P}}$ 进行乘积构造时，能够正确~观测到 s_0 的标签。$\bar{G}$ 的一个示例如~所示，位于图5的顶行以及左列 $\mathcal{A}^{\mathbb{P}}$。给定这两者，我们构造一个多权重博弈图，

定义12。乘积博弈。给定增广~博弈 $\bar{G} = (S, A, \bar{s}_0, \delta, \Pi, L)$ 和 *PDFA* $\mathcal{A}^{\mathbb{P}} = (Q, \Sigma, q_0, \delta_{\mathcal{A}}, F, \delta_{\mathbb{P}}, F_{\mathbb{P}})$，乘积博弈是一个元组~$\mathcal{P}^G = \bar{G} \times \mathcal{A}^{\mathbb{P}} = (S^{\mathcal{P}}, A, s_0^{\mathcal{P}}, s_t^{\mathcal{P}}, E^{\mathcal{P}}, W^{\mathcal{P}})$，其中

- $S^{\mathcal{P}} = (\bar{S} \times Q) \cup \{s_t^{\mathcal{P}}\}$ *is a set of states (nodes),*
- A *is a finite set of controls or actions,*
- $s_0^{\mathcal{P}} = (\bar{s}_0, q_0) \in S^{\mathcal{P}}$ *is the initial state,*
- $s_t^{\mathcal{P}} \in S^{\mathcal{P}}$ *is the terminal state,*
- $E^{\mathcal{P}} \subseteq S^{\mathcal{P}} \times A \times S^{\mathcal{P}}$ *is a set of edges, and*

- $W^{\mathcal{P}}$： $E^{\mathcal{P}} \to \mathbb{R}^{m+1}_{\geq 0}$ 是定义在边上的权重函数。

$E^{\mathcal{P}}$ 和 $W^{\mathcal{P}}$ 的构造如下。

- 边 $e = ((s,q)\ a,(s',q')) \in E^{\mathcal{P}}$ 当且仅当 $q' = \delta_{\mathcal{A}}(q, L(s'))$ 且 $s' = \delta(s,a)$。然后，多目标边权重为 $W^{\mathcal{P}}((s,q)\ a,(s',q')) = \big(W(s,a)\ -\log(\delta_{\mathbb{P}}(q, L(s')\ q'))\big)$.

- 边 $e = ((s,q)\ a, s_t^{\mathcal{P}})) \in E^{\mathcal{P}}$ 当且仅当 $F_{\mathbb{P}}(q) > 0$。然后，其权重为 $W^{\mathcal{P}}((q,s)\ a, s_t^{\mathcal{P}}) = (\vec{0}, -\log(F_{\mathbb{P}}(q)))$，其中 0是长度为 m的零向量。

乘积博弈 $\mathcal{P}^G$ 同时刻画了机器人和任务的约束、机器人/环境成本以及示范者偏好。设 $\Lambda^{\mathcal{P}} = s_0^{\mathcal{P}} s_1^{\mathcal{P}} \dots s_n^{\mathcal{P}} s_t^{\mathcal{P}} = (\bar{s}_0, q_0)(s_0, q_1) \dots (s_{n-1}, q_n) s_t^{\mathcal{P}}$是 $\mathcal{P}^G$上的一条路径。该路径在删除 $s_t^{\mathcal{P}}$后投影到 $\mathcal{A}^{\mathbb{P}}$，得到一个接受运行 $q_0 q_1 \dots q_n$。将 $\Lambda^{\mathcal{P}}$投影到 G 上得到对局$\mathcal{S} = s_0 s_1 \dots s_{n-1}$，该对局生成接受观测轨迹 $\omega = \rho_\gamma = L(s_0)L(s_1) \dots L(s_{n-1})$，并由此导出运行$q_0 q_1 \dots q_n$。此外，该路径 $\Lambda^{\mathcal{P}}$的总收益为

$$
\begin{aligned}
&\mathrm{TP}(\Lambda^{\mathcal{P}}) \\
&= \sum_{i=0}^{n-1} W^{\mathcal{P}}(s_i^{\mathcal{P}}, a_i, s_{i+1}^{\mathcal{P}}) + W^{\mathcal{P}}(s_n^{\mathcal{P}}, a_n, s_t^{\mathcal{P}}) \\
&= \Big(\sum_{i=0}^{n-1} W(s_i, a_i),\ -\sum_{i=0}^{n-1} \log\big(\delta_{\mathbb{P}}(q_i, L(s_i), q_{i+1})\big) - \\
&\qquad\qquad \log(F_{\mathbb{P}}(q_n)) \Big) \\
&= \Big(\mathrm{TP}(\mathcal{S}),\ -\log\big(\prod_{i=0}^{n-1} \delta_{\mathbb{P}}(q_i, L(s_i), q_{i+1}) \cdot F_{\mathbb{P}}(q_n) \big) \Big) \\
&= \Big(\mathrm{TP}(\mathcal{S}),\ -\log\big(P(L(\mathcal{S}))\big) \Big). \qquad (5)
\end{aligned}
$$

因此，要计算一个能够在 $\mathcal{L}(\mathcal{A}^{\mathbb{P}}) \cap \mathcal{L}(G)$中生成接受轨迹的机器人策略，我们需要在 $\mathcal{P}^G$上找到一个策略，使得对于任意环境策略，博弈中的每一局都会到达终止状态 $s_t^{\mathcal{P}}$。这样的机器人策略称为获胜策略。具体而言，在所有获胜策略中，我们要求选出产生帕累托最优成本的策略；因此，需要解决问题1.2。

6.2 帕累托前沿计算

在此，我们着重生成所有（帕累托）最优值的集合（帕累托前沿）Chen 等人（2013年a）。首先，我们使用向量上的支配关系这一概念，正式定义帕累托前沿。

定义 13。支配关系。给定两个向量 v、$v' \in \mathbb{R}^{m+1}_{\geq 0}$，如果对于每个 $0 \leq i \leq m+1$，都有 $v_i \leq v'_i$，其中 v_i 是 v的第 i个元素，则称 v 支配 v'，记为 $v \succeq v'$。如果对于所有 $0 \leq i \leq m+1$，都有 $v_i < v'_i$，则称向量 v 严格支配 v'，记为 $v \succ v'$。

定义 14。帕累托前沿。给定一个机器人策略 τ_R，将环境能够强制实现的最大总收益记为 $v^*_{\tau_R}$，即，

$$v^*_{\tau_R} = \max_{\tau_E} \mathrm{TP}(\mathrm{Play}(\tau_R, \tau_E)).$$

如果不存在另一个机器人策略 τ'_R，使其最大总收益$v^*_{\tau'_R}$严格支配 $v^*_{\tau_R}$，则称 τ_R 为 帕累托最优策略，即，

$$\nexists \tau'_R \ \ s.t. \ \ v^*_{\tau'_R} \succ v^*_{\tau_R}.$$

那么，$v^*_{\tau_R}$ 称为一个帕累托点。所有帕累托点组成的集合称为帕累托前沿 $\mathcal{P}$。

根据此定义以及式（5）中的总收益等价关系，问题1.2可简化为：在乘积博弈 $\mathcal{P}^G$上生成帕累托前沿及相应的最优策略。我们首先介绍一种生成帕累托前沿的算法。然后，给定一个帕累托点，我们可以计算相应的策略，具体讨论见第6.3节。

现在，我们介绍一种多项式算法，使用值迭代方法计算 $\mathcal{P}^G$ 上所有获胜策略对应的帕累托点。伪代码见算法 2。该算法使用下文定义的帕累托最大不动点算子 $F_{\mathcal{P}}$。

给定集合 V，令 $\mathcal{P}(V) = \{v \in V \mid \nexists v' \in V$使得 $v' \succ v\}$为 V的帕累托前沿，即 V中所有帕累托点构成的集合。此外，令 $U(s^{\mathcal{P}}) \subset \mathbb{R}^{m+1}$表示从状态 $s^{\mathcal{P}}$开始的对局的总收益向量集合，并将向量 $v \in \mathbb{R}^{m+1} \cup \{\vec{\infty}\}$ 的上集 定义为被 v支配的向量集合，即：

$$\mathrm{upset}(v) = \{v' \in \mathbb{R}^{m+1}_{\geq 0} \cup \{\vec{\infty}\} \mid v \succeq v'\}.$$

接着，我们定义 $F_{\mathcal{P}}\big(U(s^{\mathcal{P}})\big) = \mathcal{P}\big(F\big(U(s^{\mathcal{P}})\big)\big)$，其中

$$
F\big(U(s^{\mathcal{P}})\big) = \begin{cases} \bigcup\limits_{\substack{(s^{\mathcal{P}}, a, s'^{\mathcal{P}}) \in E^{\mathcal{P}} \\ u \in U(s'^{\mathcal{P}})}} \mathrm{upset}(W^{\mathcal{P}}(s^{\mathcal{P}}, a, s'^{\mathcal{P}}) + u) & \text{if } s^{\mathcal{P}} \in S_{\mathrm{R}} \\ \bigcap\limits_{\substack{(s^{\mathcal{P}}, a, s'^{\mathcal{P}}) \in E^{\mathcal{P}}, \\ u \in U(s'^{\mathcal{P}})}} \mathrm{upset}(W^{\mathcal{P}}(s^{\mathcal{P}}, a, s'^{\mathcal{P}}) + u) & \text{if } s^{\mathcal{P}} \in S_{\mathrm{E}} \end{cases}
$$

直观地说，$F_{\mathcal{P}}$ 对后继状态的总收益集合进行反向传播，并仅保留帕累托最优点。帕累托点的上集表示机器人能够实现的所有总收益的超集。首先，F 对这些集合执行交集/并集操作，以考虑环境节点/机器人节点上可能的情况。具体而言，在环境节点处，F 对上集取交集，以考虑环境做出的最坏（最大）选择。这保证机器人始终能够将总成本维持在低于这些值的范围内。同样，在机器人节点处，F 对集合取并集，以涵盖机器人的所有选择。然后，$\mathcal{P}$提取所得集合的帕累托前沿，完成 $F_{\mathcal{P}}$ 的一次迭代。

上述操作的可视化结果如图 6 所示。橙色和绿色区域表示后继节点处帕累托点的上集。通过添加边权重对这些区域进行扩展。左图展示并集操作，右图展示交集操作。所得集合以灰色显示。请注意，并集操作保留尽可能大的集合，而交集操作则缩小区域。所得帕累托点即为灰色区域的顶点。

该算法将终止状态 $s^{\mathcal{P}} = s_t^{\mathcal{P}}$ 对应的 $U(s^{\mathcal{P}})$ 初始化为全零向量，并将所有其他

算法 2： 帕累托点计算

输入： 一个博弈乘积图 $\mathcal{P}^G = (S^{\mathcal{P}}, A, s_0^{\mathcal{P}}, s_t^{\mathcal{P}}, E^{\mathcal{P}}, W^{\mathcal{P}})$，收敛裕度 ϵ

输出： $s_0^{\mathcal{P}}$ 处的帕累托点

$U(s^{\mathcal{P}}) \leftarrow \{\vec{\infty}\}\ \ \forall s^{\mathcal{P}} \in S^{\mathcal{P}} \backslash \{s_t^{\mathcal{P}}\}$;

$U(s_t^{\mathcal{P}}) \leftarrow \{\vec{0}\}$;

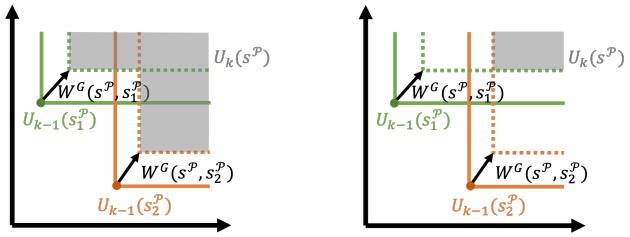
当 U 未收敛 **执行**

  对于 $s^{\mathcal{P}} \in S^{\mathcal{P}}$ **执行**

    $U'(s^{\mathcal{P}}) \leftarrow F_{\mathcal{P}}(U(s^{\mathcal{P}}))$;

  $U \leftarrow U'$

返回 $U(s_0^{\mathcal{P}})$



（a）两个集合的并集　　（b）两个集合的交集

图6。 节点 s 在迭代 k 时的两种不同集合运算示意图，其中 s_1 和 s_2 是其子节点。

状态 $s^{\mathcal{P}} \in S^{\mathcal{P}} \backslash \{s_t^{\mathcal{P}}\}$ 初始化为无穷大。然后，它递归应用算子 $F_{\mathcal{P}}$，通过反向传播帕累托点，直到解收敛。请注意，应用 F_p 会将代价从终止状态反向传播至初始状态，探索所有对局并识别所有可能的总收益。当 $U(s^{\mathcal{P}})$ 收敛时，算法终止。

通过这种方法，所有具有有限值的状态，即 $\max U(s^{\mathcal{P}}) < \infty$，都位于获胜区域中；也就是说，存在一种获胜策略，使得所有从 $s^{\mathcal{P}}$ 开始的对局都能到达终止状态 $s_t^{\mathcal{P}}$。对于其余状态，不存在获胜策略。因此，所得的 $U(s_0^{\mathcal{P}})$若为有限值，则是在仅考虑获胜策略时初始状态的帕累托前沿，从而解决了问题1.2。

示例*6*。图*7*展示了一个乘积博弈的示例。方括号中的值表示边权重，而没有标注边权重的边则假定其权重为零。初始时，每个节点的帕累托点均设为无穷大，而在终止状态 $s_t^{\mathcal{P}}$处，帕累托点设为零。帕累托点的计算首先在 $s_t^{\mathcal{P}}$上应用 F_p，然后递归地将帕累托前沿反向传播到初始状态（$\bar{s}_0$，q_0）。在第一次迭代中，（s_9，q_1）处的总收益从无穷大更新为零（$s_t^{\mathcal{P}}$处的边权重和代价均为零）。接下来，对于 $i \in \{5, \ldots, 8\}$ 的节点（s_i，q_0），其帕累托点也更新为零。注意，在（s_4，q_0）处，环境玩家可以强制执行一个自环，从而确保战胜系统。这导致值为 U（（s_4，q_0））$=\{\vec{\infty}\}$。

在(s_2, q_0)处，系统可以选择(s_4, q_0)、(s_5, q_0)或(s_6, q_0)。通过对后继状态帕累托点的上集取并集，我们得到(5,5)的帕累托点。请注意，取并集后，无穷大值被(5,5)这一点的上集所涵盖。换言之，(5,5) 支配无穷大值，因此无穷大的代价

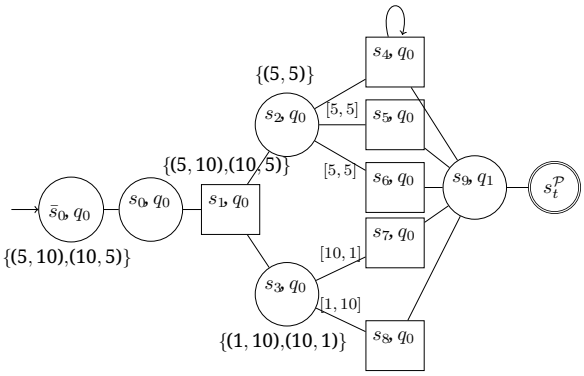


图7. 帕累托点计算示意图。圆形节点和方形节点分别表示系统状态和环境状态。边权重显示在方括号中，节点处的帕累托点用花括号括起。未显示权重的边假定权重为零。

不再纳入考虑。同样，(s_3, q_0)处的帕累托点为 *(1,10)* 和 *(10,1)*。在下一时刻的迭代中，在环境状态 (s_1, q_0) 处，我们将(5, 5) 的上集与(1,10) 和(10, 1)的上集并集取交集，得到帕累托点(5,10) 和(10, 5)。经过两次迭代，这些值被传播到$(\bar{s}_0, q)$，并达到收敛。

下面，我们证明该算法的完备性和运行时复杂度。具体而言，我们证明 $F_{\mathcal{P}}$ 算子是压缩的，且其最大不动点是真实帕累托前沿，并能在有限时间内达到。为简便起见，我们扩展支配算子 $\succeq$ 的定义，使其作用于总收益集合，即若对所有 $u' \in U'$ 和所有$u \in U$均有 $u' \succeq u$，则记为 $U' \succeq U$。

引理 3. 给定博弈乘积 $\mathcal{P}^G$ 和算子 $F_{\mathcal{P}}$，令$U_i(s^{\mathcal{P}})$ 表示状态 $s^{\mathcal{P}} \in S^{\mathcal{P}}$在算法 i第 *2* 次迭代中获得的总收益点集合。那么，$U_{i+1}(s^{\mathcal{P}}) = F_{\mathcal{P}}(U_i(s^{\mathcal{P}}))$ 中的总收益点支配 $U_i(s^{\mathcal{P}})$ 中的总收益点，对每个 $s^{\mathcal{P}} \in S^{\mathcal{P}}$ 都成立，即

$$U_{i+1}(s^{\mathcal{P}}) \succeq U_i(s^{\mathcal{P}}) \qquad \forall s^{\mathcal{P}} \in S^{\mathcal{P}}.$$

证明见附录A。引理3 表明，算子 $F_{\mathcal{P}}$ 具有单调性，即新计算出的总收益集合支配先前的集合。因此，通过反复应用 $F_{\mathcal{P}}$，我们可以剪除被支配的值。以下命题表明，经过有限次 $F_{\mathcal{P}}$迭代，总收益集合即可传播到所有状态。

命题 1。 总收益集合可以在 $\mathcal{O}(|S^{\mathcal{P}}|(|S^{\mathcal{P}}| + |E^{\mathcal{P}}|))$ 次迭代后传播到每个状态。

证明见附录 B。利用该命题的结果，以下引理表明 $F_{\mathcal{P}}$ 可在多项式时间内达到不动点。

引理 4。 经过命题*1* 中规定的最大迭代次数后，U_i 不会发生变化。

证明。 这可以通过矛盾来证明。回想一下，$F_{\mathcal{P}}(U_i(s^{\mathcal{P}}))$所获得的收益集合只有在

找到一条通往终止状态的更短路径时才会发生变化。然而，根据命题 1，所有具有优势收益的路径至多经过 $\mathcal{O}(|S^{\mathcal{P}}|(|S^{\mathcal{P}}|+|E^{\mathcal{P}}|)$次迭代即可探索完毕。经过$\mathcal{O}(|S^{\mathcal{P}}|(|S^{\mathcal{P}}|+|E^{\mathcal{P}}|)$次迭代后，如果存在一条更短的路径，那么它必定包含一个尚未访问的后继节点 $s^{\mathcal{P}\prime}$。设当前状态为 $s^{\mathcal{P}}$，在步骤 i从 $s_t^{\mathcal{P}}$到状态 $s^{\mathcal{P}}$和 $s^{\mathcal{P}\prime}$的总收益分别为 $U_i(s^{\mathcal{P}\prime})$和$U_i(s^{\mathcal{P}\prime})$。如果 $s^{\mathcal{P}\prime}$ 位于最短路径上，那么$U_i(s^{\mathcal{P}})\preceq U_i(s^{\mathcal{P}\prime})\oplus\{W(s^{\mathcal{P}\prime},a,s^{\mathcal{P}})\}$。然而，$U_i(s^{\mathcal{P}})$必须优于 $U_i(s^{\mathcal{P}\prime})\oplus\{W(s^{\mathcal{P}\prime},a,s^{\mathcal{P}})\}$，否则这些总收益就会被包含在上集中，即对于 $u\in U_i(s^{\mathcal{P}\prime})$，$U_i(s^{\mathcal{P}})\supseteq\text{upset}(W(s^{\mathcal{P}\prime},a,s^{\mathcal{P}})+u)$。因此，$U_i$经过 $\mathcal{O}(|S^{\mathcal{P}}|(|S^{\mathcal{P}}|+|E^{\mathcal{P}}|)$次迭代后收敛。

上述引理和命题表明，帕累托前沿是 $F_{\mathcal{P}}$ 的最大不动点，可在有限时间内计算得出。因此，算法2具有完备性和效率，具体如下所述。

定理 2。 给定一个博弈 $\mathcal{P}^G$，算法 2可在 $\mathcal{P}^G$ 中以多项式时间计算所有状态的帕累托前沿。

证明。 将帕累托集初始化为无穷大的向量后，根据引理3，算法 $i\geq 02$ 能够在每次迭代中正确计算帕累托点。根据命题1，算法经过 $\mathcal{O}(|S^{\mathcal{P}}|(|S^{\mathcal{P}}|+|E^{\mathcal{P}}|))$ 次迭代后传播所有边权重，从而根据引理 4 使总收益集合收敛。

注释1。 虽然定理2 表明算法2 保证在有限次迭代内终止，但每次迭代都需要执行交集和并集的集合运算。在我们的实现中，我们使用多边形裁剪算法，其运行时间为 $\mathcal{O}(n\cdot m)$，其中n 和 m 分别是两个多边形的顶点数*Puri 和 Prasad (2013年)*。已知该算法的运行时间为$\mathcal{O}((k+n)\log n)$，其中 k 是交点的数量。

6.3 帕累托最优策略综合

这里我们将说明，针对选定的帕累托点，可以在与乘积博弈中的节点数成线性关系的时间内计算出一个策略。该算法见算法3。从整体上看，该算法会在每个状态选择一个动作，以找到总收益小于或等于帕累托点的路径。请注意，帕累托点表示可能的最差总收益。从初始状态出发，任何通向终止状态的路径，其代价都必须小于或等于帕累托点。因此，只要帕累托点与当前节点处累计路径成本之间的差为正，所选动作就能确保总收益不超出帕累托点。该算法会跟踪从初始状态开始的剩余总收益。

在第 3 行，我们首先将初始状态和所选的帕累托点 $p\in\mathcal{P}$ 加入队列。在第 3 行，我们选择一个后继节点 $s^{\mathcal{P}}$，使其帕累托点 p' 仍处于当前节点的成本集内，$p-W^{\mathcal{P}}(s^{\mathcal{P}},a,s_i^{\mathcal{P}})$。这确保了剩余总收益为正，$p-W^{\mathcal{P}}(s^{\mathcal{P}},a,s_i^{\mathcal{P}})-p'\geq 0$。获得策略后，算法会检查是否存在可能的循环。为此，我们可以仅通过保留乘积博弈中的策略动作来构建策略图，

算法 3： 针对帕累托点的策略综合 t p

输入： 博弈乘积图
$\mathcal{P}^G=(S^{\mathcal{P}},\ A,\ s_0^{\mathcal{P}},\ s_t^{\mathcal{P}},\ E^{\mathcal{P}},\ W^{\mathcal{P}})$，一个帕累托指向初始状态 p，以及.....的集合
总收益 U

输出： 一个确定性策略 τ

```
1  Q = Queue((s_0^P, p));
2  Visited = Set(s_0^P);
3  E = List();
4  当 Q不为空 则
5      s^P, p ← Q.pop();
6      对于 (s^P, a, s'^P) ∈ E^P执行
7          如果 p' ∈ U(s'^P) 且 p − W^P(s^P, a, s'^P) ≥ p'
           then
8              τ(s^P) = a;
9              如果 s'^P 不在 Visited 则
10                 Q.add((s'^P, p'));
11                 Visited.add(s'^P);
12                 E.append((s^P, a, s'^P));
13 返回 τ
```

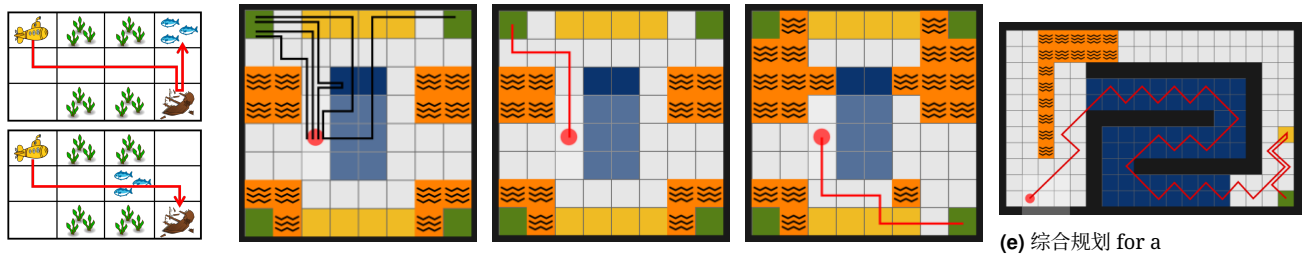
对策略图执行反向可达性分析，并且只保留从接受状态可达的状态。这样可以避免在策略图中陷入循环。

备注 2。我们的算法同样适用于静态环境。静态环境可以视为动态环境，其中环境玩家在每个状态下只能采取一个动作。

6.4 可扩展性讨论

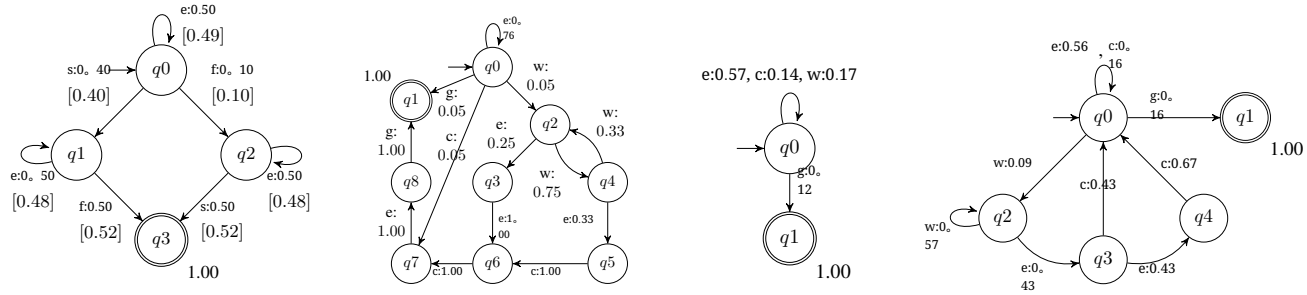
如定理 2所示，综合算法的复杂度关于乘积博弈 $\mathcal{G}^{\mathcal{P}}$的规模呈多项式级，其中乘积博弈是博弈 G 与学习得到的概率确定有限自动机$\tilde{A}^{\mathbb{P}}$的笛卡尔积。对于机器人任务，概率确定有限自动机通常规模较小，因此，我们框架的可扩展性主要受G的规模影响，而该规模取决于将问题特定抽象为双人博弈的过程。尽管这一抽象过程已有充分研究，且与具体领域相关（移动机器人学方面，Kress- Gazit 等人(2018，2007）；Lahijanian 等人(2009；机器人机械臂方面，He 等人(2015，2019a））， 这一过程仍可能带来高昂的计算开销，尤其是在涉及多个环境智能体的场景中。

为缓解这一问题，可以采用二元决策图（BDDs）和代数决策图（ADDs）等符号表示方法，以紧凑地表示和操作大型博弈图，如 He 等人(2019年b) 所示；Muvvala 和 Lahijanian(2023年)。此外，我们强调，与基于线性时序逻辑的规范相比，使用学习得到的概率确定有限自动机来表示任务，可显著缩小自动机规模，从而提高乘积构造的效率。这是因为，由协安全线性时序逻辑或线性时序逻辑任务$_{\mathrm{f}}$ 规范生成的确定有限自动机，其规模相对于公式规模呈双指数增长。线性时序逻辑规范通常不包含物理约束，因此自动机必须表示所有逻辑上可能的执行。相比之下，我们的概率确定有限自动机是从示范中学习得到的，而这些示范天然



(a) 非马尔可夫任务 **(b)**五个示范 **(c)** 综合规划 **(d)** 综合规划 **(e)** 综合规划 for a 对角线移动机器人

图8. 案例研究中考察的各种环境和机器人。(a) 非马尔可夫任务的学习与规划。(b) 来自 Vazquez-Chanlatte 等人 (2018) 的环境和示范。(c)-(e) 基于 (b) 中学习到的任务综合出的规划（以红色显示）。



(a) 图 8a 的概率确定有限自动机 **(b)** 香草味， $\alpha = 0.4$，图 8b。 （c）原味， $\alpha = 4$，图 8b。 **(d)** 预处理， $\alpha = 4$，图 8b

图 9。 图 8a 和 8b中的场景的任务规范和学习到的概率确定有限自动机。每个字母表示一个仅包含单个原子命题 s=$\{ship\}$、f=$\{fish\}$、b=$\{blue\}$、c=$\{carpet\}$、g=$\{green\}$、p=$\{purple\}$和 $e = \emptyset$的符号。 $F_{\mathbb{P}}$ 双边状态的终止概率为 1，其他所有状态的终止概率均为 0。(a) 中方括号内的数值为学习到的概率。

反映了物理约束，因此生成的自动机规模要小得多，也更易于处理。

7 案例研究与评估

在本节中，我们通过五个案例研究评估所提出算法的性能。我们证明，采用我们的方法生成的解满足问题 1中提出的所有要求。这些案例研究旨在回答以下关键问题：

i. 算法能否从示范中学习非马尔可夫任务，同时将示范者偏好建模为概率确定有限自动机?

ii. 无论超参数或示范数量如何，算法能否学习一个确保安全性永不受损的概率确定有限自动机?

iii. 综合策略能否保证在动态环境中完成任务，同时最大化偏好并最小化机器人代价?

iv. 该算法是否适用于现实世界中的机器人系统?

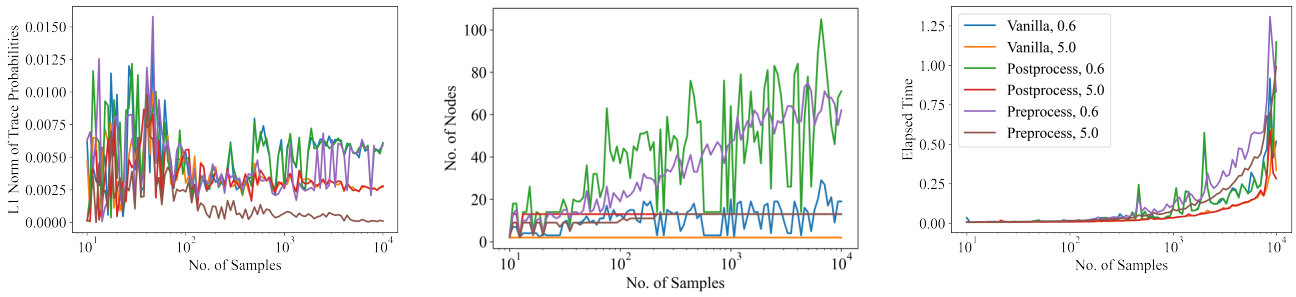
我们基于*flexfringe* 库 Verwer和Hammerschmidt（2017年）中使用的*MDI*方法实现了EDSM算法。我们将基本算法称为原始算法。所有案例研究均在配备2.3 GHz双核Intel Core i5处理器和16 GB内存的MacBook Pro上运行。所有案例研究的视频均可观看 [1]。

7.1 非马尔可夫任务的学习与规划

在本案例研究中，我们考虑图8a所示的机器人场景。任务是以任意顺序访问鱼群学校和沉船，同时始终避开珊瑚礁。偏好是先访问沉船。图9a展示了该规范的概率确定有限自动机表示。

为学习这一任务，我们在图8a所示的网格世界环境中，从该概率确定有限自动机中采样了 1000 条轨迹。根据这些示范，Vanilla算法学得的概率确定有限自动机与真实概率确定有限自动机具有完全相同的结构，且概率与真实值相差不超过 0.02（真实值见图9a方括号内）。

如概率确定有限自动机（PDFA）所示，我们的方法正确地学习到了一个非马尔可夫任务：按任意顺序访问沉船和鱼群，并且更倾向于先前往沉船。利用该概率确定有限自动机，我们的规划器生成了图8a （上）所示的机器人轨迹，它先访问沉船，再访问鱼群，符合要求。接下来，我们通过移动鱼的位置来改变环境，使其位于机器人前往沉船的途中，如图 8a（下）所示。该图还展示了在此环境中使用同一个已学习的概率确定有限自动机生成的综合规划。请注意，由于环境约束，机器人没有先访问沉船，而是先访问鱼群，再访问沉船；这同样是正确的行为。与学习强烈依赖环境的策略相比，学习规范具有这种泛化能力，这正是其优势所在。



(a) L1 范数误差（对数刻度） **(b)**节点数 **(c)** 计算时间 [s]

图 10. 所提出的算法的性能分析。(a)–(b) 中的图使用与 (c) 相同的图例。

7.2 利用少量样本在安全性约束下进行学习

在这个案例研究中，我们考虑图 8b所示的环境和五个示范，取自Vazquez- Chanlatte 等人 (2018)，并以概率确定有限自动机的形式学习规范，以此与Vazquez-Chanlatte 等人(2018)的方法进行比较；后者基于学习规范公式。在这个网格世界中，每种颜色代表一种物体：橙色是熔岩，蓝色是水，黄色是干燥地毯，白色是空地，绿色是充电站。任务是抵达充电站。然而，机器人在身上有水时不应充电。也就是说，一旦机器人被弄湿（进入水中），就必须到干燥地毯上烘干。

7.2.1 少量样本我们首先使用 Vanilla算法处理五个示范，学习到了图中所示的 PDFA9b。请注意，在学习到的 PDFA 中，要到达最终状态，必须始终观测到绿色区域（充电站）。这表明，到达充电站这一任务已被正确学习。接下来，在 PDFA 最右侧的分支上，当机器人被淋湿时，总会观测到地毯。同样，算法成功学习了这样的任务：机器人被淋湿后、到达地毯，再到达充电站。一个有趣的观测结果是，PDFA 还学到了机器人离开地毯后必须在一个步骤内到达地毯。事实上，这是样本中的一种偏差，因为所有包含地毯的示范都具有这一特性。如果这是演示者的意图，那么这种行为就是正确的；如果不是，则可以通过提供更多样本来消除这种偏差。

这种单步偏差在Vazquez-Chanlatte 等人(2018) 的方法中并不明显，因为他们所考虑的语言的语法不允许使用“下一时刻”算子。相比之下，我们的方法是在包含下一算子的正规语言上进行推断。此外，在Vazquez-Chanlatte 等人(2018) 中，仅用 5 个示范学习规范耗时 95 秒，而我们的方法耗时不到 0.01 秒。

7.2.2 超参数选择与安全性 图 9b中的概率确定有限自动机是超参数 α 设为 0.4 时，Vanilla算法的结果。它是一个调节旋钮，用于控制我们允许合并的激进程度。 α的值越高，概率确定有限自动机就越小。如果我们能正确调整超参数，就能得到如上所述的理想结果。但是，如果过度增大 α ，某些合并可能会导致不安全行为。出现不理想的合并，是因为算法只是在尝试 最小化该结构的大小。事实上，如何为 α 选择正确的值仍是一个开放问题。对于 $\alpha = 4$，根据相同示范学习得到的概率确定有限自动机如图 9c所示。该概率确定有限自动机不考虑安全性，只要求到达 充电站。我们可以通过嵌入安全规范来缓解这一问题。我们定义如下安全公式：

$$\varphi_{\text{safe}} = \mathcal{G}\neg\text{lava} \wedge \mathcal{G}(\text{water} \rightarrow \mathcal{X}(\varphi(\neg\text{charge}, \text{carpet}, \text{k})),$$

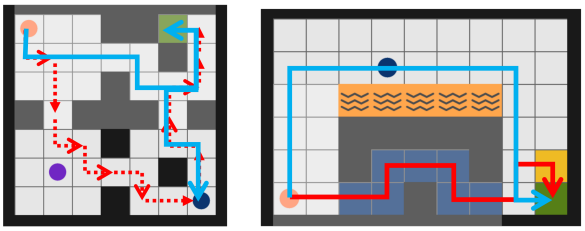
其中， φ 是递归定义的公式： $\varphi(a,\ b,\ k) = a \wedge (b \vee \mathcal{X}(\varphi(a,\ b,\ k-1)))$，以及 $\varphi(a,\ b,\ 0) = a$，其读法是“访问 a ，持续 k 步，除非访问了 b”。该公式要求永远不能前往熔岩，并且如果机器人进入水，则除非访问地毯，或在空地上连续停留 k 步以晾干，否则不能充电。我们在所有实验中设置 $k = 10$ 。

现在，我们使用相同的五个示范，通过后处理算法和预处理算法学习概率确定有限自动机，并使其满足 α =4约束 φ_{safe}。后处理算法生成了一个具有13个节点和36条边的大型概率确定有限自动机，因为安全性确定有限自动机本身就很大（具有12个节点和34条边）。尽管规模较大，它始终保证不会违反 φ_{safe}。预处理算法生成的概率确定有限自动机如图9d所示。它规模较小，并且正确地嵌入了安全性和活性。此外，两个学习得到的概率确定有限自动机都接受所有示范。至于概率，后处理算法得到的概率确定有限自动机的平均 L1 范数误差为 1.65×10^{-3}，而预处理算法得到的概率确定有限自动机的平均 L1 范数误差为 7.42×10^{-5}，这表明预处理算法的性能更好。后处理算法得到的概率确定有限自动机具有更大的概率误差，这是由于其与安全性确定有限自动机组合后，会剪除学习得到的概率确定有限自动机中的不安全轨迹，从而破坏学习得到的概率分布。

此外，我们注意到，将任务学习为概率确定有限自动机（PDFA）能够增强可解释性，为设计者和演示者提供另一种调整超参数的工具 α，如图9所示。接下来，我们通过增加样本数量，对各种学习方法进行全面比较。

7.3 后处理算法与预处理算法

这里的任务与上文相同，但目标是随着样本数量增加，定量分析并比较所提出算法的性能。我们从真实概率确定有限自动机中随机抽取示范，并使用后处理和预处理算法



(a) 鱼群与沉船 **(b)** 充电站

图11. 动态 MiniGrid 环境。实线和点线分别表示 1 步和 2 步动作。

学习概率确定有限自动机，超参数值分别为 $\alpha = 0.6$（合并力度较弱）和 $\alpha = 5.0$（合并力度较强），以展示极端结果。我们根据以下度量评估所得的概率确定有限自动机：轨迹概率误差的 L1 范数、状态数量和计算时间。

请注意，理想的方法旨在降低这三项度量。也就是说，对于概率确定有限自动机学习，只要自动机能够准确表示接受轨迹（语言）上的概率分布，状态数量越少越好。这意味着，语言（任务）的表示是紧凑的。这带来了若干优势，包括更快的策略综合（因为更小的概率确定有限自动机会产生更小的乘积博弈）和更好的可解释性。

结果如图 10 所示（所有图共用同一图例）。结果表明，预处理算法在准确性和大小方面再次优于其他算法（但速度较慢）。根据这些结果，我们可以说，预处理算法在准确性和自动机大小方面表现最佳。此外，在所有试验中，其输出的概率确定有限自动机均未违反安全性（已检查，但未在图中展示）。

7.4 不同环境中各种机器人的规划

7.4.1 静态环境中的规划 从上述学习到的概率确定有限自动机中，我们选取了一个 L1 误差范数较小的模型。然后，使用这个概率确定有限自动机，我们针对不同的机器人和环境进行了规划，这些机器人和环境与示范所处的环境不同（见图 8b）。在所有情况下，计算得到的规划都正确满足了要求和偏好。在图 8d所示的环境中，熔岩 迫使机器人前往右下方的充电站。请注意，机器人通过地毯 避开了水，这正是更受偏好的行为。在图 8e中，我们修改了机器人的动力学，使其只能进行对角移动。算法再次成功生成了满足要求且不违反规范的规划。由于规范独立于任何机器人系统和环境，我们的框架能够适应环境和机器人动力学的变化。

7.4.2 动态环境中的规划 我们为已学习的概率确定有限自动机（PDFA）在动态环境中合成策略。回顾一下同时访问鱼群和沉船的任务。现在，我们假设鱼群可以自由地动态移动。新示例如图 11a 所示。红色车辆必须捕捉蓝色鱼，并且

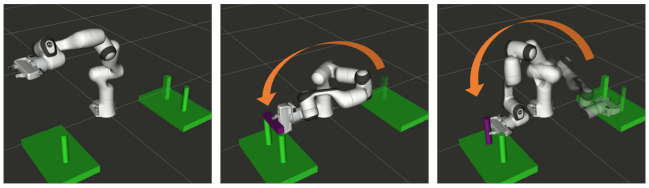


图 12. 机械臂完成学习到的搭建拱门任务。

找到绿色沉船，同时避开只能在左下方区域内移动的紫色车辆。此外，我们为机器人增加了更多动作；它可以选择每次移动一到两步。我们将动作的能耗成本设定为与其步数成正比。每次移动两步可以确保以更少的步数捕捉鱼，但会消耗更多能量。

我们的算法找到了由六个帕累托点（代价和偏好）组成的帕累托前沿，机器人能够保证达到这些点。其中一个点是 (40.0,40.72)，即能耗成本对应的总收益最高为40.0，偏好最高值（–对数 P）为40.72。相应策略下得到的路径以红色显示在图 11a中。另一个帕累托点是 (59.0,20.01)，相应策略下的一局以蓝色显示在图11a中。

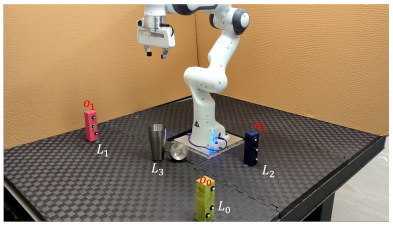
我们通过为环境随机选择策略，在这个博弈中模拟了 1000 次对局。所有模拟得到的对局都成功完成了任务，且其总收益均受帕累托点界定。这些案例研究表明，无论环境如何行动，我们的算法计算出的策略都能保证任务完成，并使总收益保持在所选帕累托点的范围内。

这里，我们展示了算法如何以多种方式避开另一个智能体并完成任务。回顾一下从上方前往充电站的任务。现在我们考虑图11b所示的扩展环境。设想机器人是一辆自动驾驶汽车，而蓝色智能体是一名行人。汽车必须避开行人并抵达充电站，也可以穿过水，然后在地毯上弄干，以避开行人。我们假设行人只能在最上方两行区域内行走，而机器人可以走一步或三步来避免与行人发生冲突。

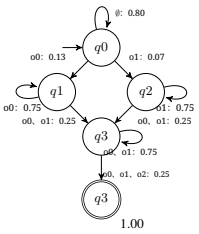
我们的算法为这个示例找到了八个帕累托点。我们通过模拟图11b中的策略，展示了两种不同的对局。产生蓝色路径的策略对应帕累托点(47，5。24)，这意味着以更受偏好的方式完成任务，但机器人代价较高。红色路径策略对应帕累托点(12，15。39)，它能保证较低的机器人代价，但完成任务的方式不那么受偏好。对应其他帕累托点的路径与这两种类似，但其行为会根据机器人每个动作所走的步数而变化。

7.5 操作任务的学习与规划

7.5.1 拱门搭建示例 为说明我们的方法并不局限于移动机器人，我们考虑了图 12 （左）中的一个操作示例。机器人是拥有 7 个自由度的 Franka Emika Panda 机械臂，潜在任务是



(a)实验设置



(b)学习得到的概率确定有限自动机

图13. 鸡尾酒调制实验

用两个圆柱体作为柱子，并在顶部放置一个矩形箱体来搭建拱门。机器人的抽象依据 He 等人 (2019a, 2015); Muvvala 等人(2022年)的方法进行，最终得到约 20,000 个状态。机器人获得了九次示范：五次最受偏好的示范（最快）、三次中等长度的示范（动作数量为其 1.4 倍），以及一次非常糟糕的示范（动作数量为其 3 倍）。我们使用 $\alpha = 1.8$学习了一个概率确定有限自动机。学习得到的概率确定有限自动机有四个状态，规划耗时 0.036 秒。机器人执行规划的过程如图 12（中和右）所示，表明机器人成功学习并执行了完成任务时最受偏好的方法。

7.5.2 鸡尾酒调制示例为了展示我们反应式算法的强大能力，我们在人机协作场景中展示一个鸡尾酒制作示例。设想机器人和人类在酒吧厨房中并肩各自调制鸡尾酒。实验设置如图13a所示。方块代表酒瓶，可通过执行“转移抓取”和“转移释放”动作，在预定义位置之间移动 L_0、 L_1、 L_2和 L_3。人类有两种选择：随时介入并移动一个物体，或等待机器人执行动作。不过，存在一些限制：机器人持有物体时，人类不能介入；同一物体也不能被人类介入两次。如果人类确实介入了某个物体，则必须在机器人执行两步动作之内将该物体归还。边权重根据各位置之间的距离自动分配。

机器人的任务是将酒液 o_0、 o_1和 o_2 倒入摇酒器，而人类可以介入，一次借用一种酒液。底层任务是先按任意顺序倒入 o_0 和 o_1 ，最后再倒入 o_2 。最优先的物体抓取方式是 $o_0 \to o_1 \to o_2$。我们采集了3个示范，并在给定安全要求“绝不在 o_2 之前观察到 o_0 和 o_1”的情况下学习了一个概率确定有限自动机，即$\mathcal{G}(o_2 \to X(\neg o_0 \wedge \neg o_1))$。学习得到的概率确定有限自动机见图13b。

我们的算法计算出了两个不同的帕累托点，分别代表距离成本与偏好成本之间的权衡：(4.41, 12.65) 和 (4.68,10.88)。此外，它还为每个帕累托点分别生成了对应的策略，记为 τ_0 和 τ_1。这些策略产生的对局总收益均受各自的帕累托点约束。例如，策略 τ_0 按照 o_1、 o_0和 o_2的顺序倾倒酒液，总成本为 (3.44,7.83)。相反，策略 τ_1 按照 o_0、 o_1和 o_2的顺序执行，总成本为 (3.45,7.21)。值得注意的是，由于 o_1 距离更近（距离成本更低），但偏好程度不如 o_0，因此总与策略τ_1相比，策略 τ_0 的收益以偏好成本较高为代价，实现了更低的距离成本。此外，由于施加了安全要求，机器人在取回其他物体之前，绝不会尝试拿起 o_2。

在人为干预的情况下，这些策略仍能确保最坏情况下的总收益处于帕累托点的范围内。在图14中，我们展示了策略 τ_0 生成的总收益最差的对局。策略 τ_1 生成的对局与策略 τ_0 生成的对局相似，只是 o_0 和 o_1的顺序颠倒了。

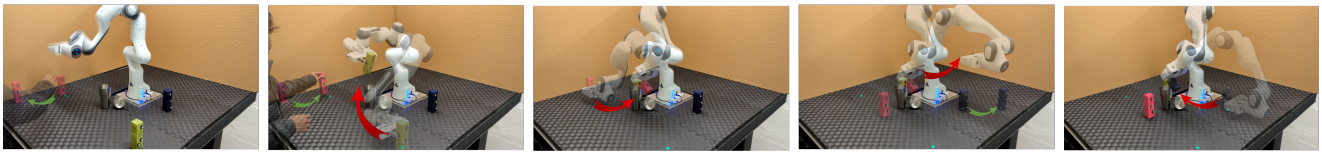
在图14a所示的场景中，对局开始时，机器人移动到位置 L_1 以取回物体o_1。然而，在机器人抓取 o_1之前，人类操作员进行了干预。由于约束规定，在一次干预尚未结束时，人类不能干预另一个物体，机器人推断不会再有其他干预。因此，它继续拿起 o_0 ，并将其中的液体倒入位于 L_3的摇酒器中。人类归还最初受到干预的物体后，机器人返回，将 o_1 倒入摇酒器。随后，当机器人试图拿起 o_2时，人类再次进行了干预。机器人耐心等待，直到 o_2被归还，然后将最后一种酒液倒入摇酒器。在图14b中，机器人的策略以将 o_1 倒入摇酒器开始。有趣的是，随后它转移到 o_2，并将其移到更靠近 L_0的位置。这个动作看起来或许令人意外，但实际上是为了在 o_0发生人为干预时，尽量减少往返于 L_0和 L_2之间的移动代价。果然，当机器人试图拿起 o_0时，人类进行了干预，机器人不得不返回 L_2 ，一直等到 o_0 被归还。最后，机器人依次将 o_0 和 o_2 倒入摇酒器。

这些案例研究表明，基于帕累托前沿计算的方法能够生成多种策略，以满足不同的优先事项和约束。通过提供一系列帕累托最优解，用户可以根据应用的具体要求选择最合适的策略，例如优先考虑距离成本、偏好成本，或在两者之间取得平衡。

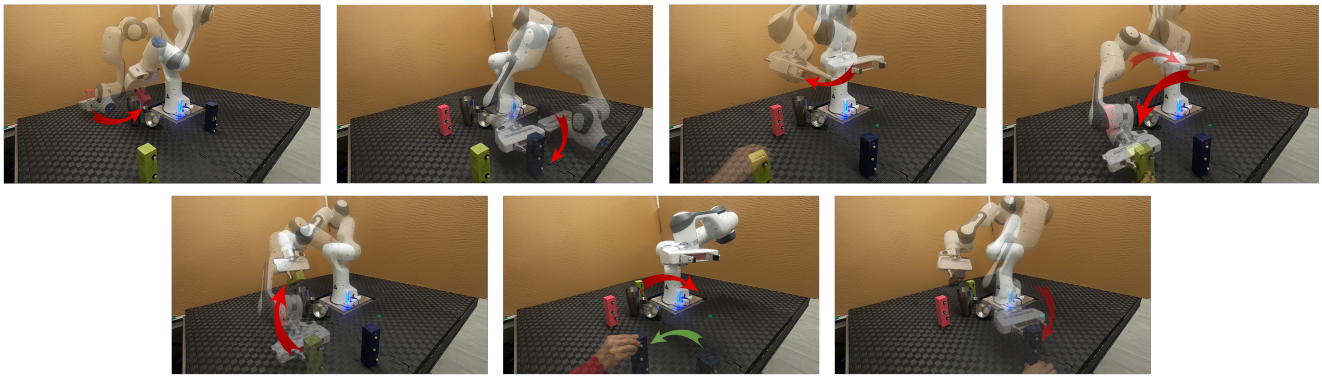
7.6 基准测试

在此，我们通过实证评估所提出框架的计算性能和可扩展性。为了进行全面评估，我们纳入了所有涉及动态环境的案例研究，即第7.4.2 节和7.5.2节中的案例，以及一个额外的 MiniGrid 场景。该场景涉及两个环境智能体，如图15所示。在此场景中，红色机器人需要抓住一个蓝色（环境）智能体，并将其送到绿色区域。每个蓝色智能体都有丰富的动作空间，包括朝四个基本方向移动以及对角移动。相比之下，红色智能体只能朝四个基本方向移动，但每次可以移动三步，而蓝色智能体每次只能移动一步。通过对蓝色智能体取笛卡尔积，我们得到一个双人博弈抽象。我们的综合算法为红色智能体计算出多个帕累托最优获胜策略。

表 1 详细展示了我们的框架在三个具有代表性的 MiniGrid 环境以及机械臂上的计算性能细目



(a) 策略 τ_0的最差距离成本为 [4.41, 7.88] 的一次对局



(b) 策略 τ_0的最差偏好成本为 [4.35, 12.65] 的一次对局

图14. 策略 τ_0的对局。

表1. 四种动态环境下的基准测试结果。该表报告了 (i) 学得的概率确定有限自动机 $\mathcal{A}^{\mathbb{P}}$、(ii) 博弈图 G和 (iii) 乘积博弈 $\mathcal{P}^G$的规模（节点数和边数），(iv) 乘积博弈$\mathcal{P}^G = \mathcal{A}^{\mathbb{P}} \times G$ 构造、(v) 所有帕累托点集合（帕累托前沿） $\mathcal{P}$的综合，以及 (vi) 帕累托最优策略集合$\mathrm{T}^* = \{\tau_p^* \mid p \in \mathcal{P}\}$的综合所需的计算时间（秒），以及帕累托点的数量 $|\mathcal{P}|$。记号 $|E^{\mathcal{A}^{\mathbb{P}}}|$、$|E^G|$和 $|E^{\mathcal{P}}|$分别表示 $\mathcal{A}^{\mathbb{P}}$、G和 $\mathcal{P}^G$的边数。

场景		$\mathcal{A}^{\mathbb{P}}$		G		$\mathcal{P}^G$			综合时间（秒）		$\lvert\mathcal{P}\rvert$
		$\lvert Q\rvert$	$\lvert E^{\mathcal{A}^{\mathbb{P}}}\rvert$	$\lvert S\rvert$	$\lvert E^G\rvert$	$\lvert S^{\mathcal{P}}\rvert$	$\lvert E^{\mathcal{P}}\rvert$	时间（秒）	$\mathcal{P}$	T^*	
鱼与沉船	图 11a	4	7	5328	202464	15626	585721	18.90	1087.13	91.02	1
充电站	图 11b	3	5	1886	61192	3245	100007	3.60	209.79	87.77	5
鸡尾酒调制	图 13	5	9	17662	39102	31765	67298	7.24	498.74	66.86	3
三个智能体	图 15	4	7	13824	497664	39619	1376348	48.16	1626.92	278.50	2

（鸡尾酒调制）。其中报告了学习得到的概率确定有限自动机 $\mathcal{A}^{\mathbb{P}}$、博弈图 G和乘积博弈 $\mathcal{P}^G$的规模（节点数和边数），以及乘积博弈 $\mathcal{P}^G = \mathcal{A}^{\mathbb{P}} \times G$ 构建、合成所有帕累托点（帕累托前沿）集合 $\mathcal{P}$和合成帕累托最优策略集合 $\mathrm{T}^* = \{\tau_p^* \mid p \in \mathcal{P}\}$所需的计算时间（秒）。表中还列出了每项实验的帕累托点数量 $|\mathcal{P}|$ 。

正如预期，乘积博弈的规模随概率确定有限自动机的规模线性增长。值得注意的是，尽管帕累托前沿综合算法是计算量最大的步骤，但在实践中仍然易于处理，这与定理2所证明的多项式时间复杂度一致。事实上，策略提取所需时间甚至更短。

我们注意到，框架的整体可扩展性从根本上受到底层博弈图规模的限制G。在我们的实验中，综合时间最长的场景出现在图 11a 和图 15所示的场景中；由于底层博弈图中的动作数量较多，乘积博弈的边数 $|E^{\mathcal{P}}|$ 也较多 G。这是文献中也已指出的一项挑战。为缓解这一问题，已有研究提出使用二元决策图（BDD）或代数决策图（ADD）等符号表示方法，并可将其纳入框架，以减少内存使用和计算成本 He 等人(2019年b); Muvvala 和 Lahijanian(2023年).

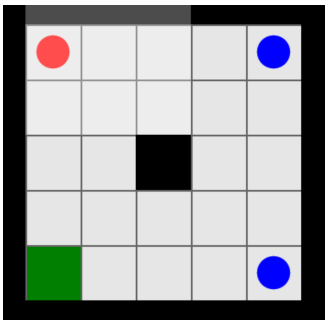


图 15。 红色智能体必须抓住任意一个蓝色智能体，并将其送到绿色区域。每个蓝色智能体的动作空间由四个基本方向和两个对角方向组成。红色智能体可以沿四个基本方向移动，每次最多移动 3 步。

尽管如此，报告的结果为我们提出的框架在中等规模问题上的实际可行性提供了实证证据，尤其是在室内机器人学等结构化环境中。

8 结论

本文提出了一种从示范中学习规范的新方法，其形式为概率确定有限自动机（PDFA）。与现有研究不同，该方法无需先验

知识，运行速度快，并且能够捕捉偏好。我们提出了一种将安全约束纳入学习过程的预处理算法。该算法显著提升了速度。我们还引入了一种用于已学习规范的规划算法，同时优化多重成本。该算法生成一组所有帕累托点，用户可以从中进行选择，并为每个帕累托点生成一个帕累托最优策略。大量评估表明，该框架具有灵活性，并能够将知识稳健地迁移到各种环境和机器人中。

规范学习未来的研究方向包括：推断无限时域上的规范、将先验预测或知识嵌入推断算法，以及利用反例指导推断。对于策略综合，我们关注在无限时域上综合策略、为机器人行为生成解释，以及从未预测到的状态中恢复（例如，故障或未建模的人工干预）。

引理3的证明

证明。 这很容易对系统节点证明。随着算法逐步找到多条可行路径，它会选择占优总收益对应的路径。形式上，通过对总收益取并集和上集，集合中只保留占优的帕累托点。令 $U_i(s^{\mathcal{P}}) = \{u_1, \ldots, u_n\}$，并假设在某次迭代中找到了一条总收益较小的路径$u_1' \succeq u_1$。那么，

$$U_{i+1}(s^{\mathcal{P}}) = \{u_1', u_2, \ldots, u_n\} \succeq \{u_1, u_2, \ldots, u_n\} = U_i(s^{\mathcal{P}}).$$

在环境节点处，如果算法尚未找到从某个状态到终止节点的路径，则该状态下的帕累托点仍为无穷大。只有在其后继节点的帕累托点更新后，这些帕累托点才会更新。令 u_1 为无穷大，则

$$U_{i+1}(s^{\mathcal{P}}) = \{\vec{\infty}\} = U_i(s^{\mathcal{P}}).$$

如果所有后继节点的路径都更短，即对于所有 $i \in \{1, \ldots, n\}$，都有 $u_i' \succeq u_i$，那么

$$U_{i+1}(s^{\mathcal{P}}) = \{u_1', \ldots, u_n'\} \succeq \{u_1, \ldots, u_n\} = U_i(s^{\mathcal{P}}).$$

由于系统节点处的总收益单调递减，环境节点处所有总收益的集合也只能单调递减。因此，我们可以推导出，对于所有 $s^{\mathcal{P}}$，$U_{i+1}(s^{\mathcal{P}}) \succeq U_i(s^{\mathcal{P}})$。下面，我们将证明，当博弈中存在强连通分量（SCCs）时，这一结论同样成立。

假设博弈由强连通分量组成。环境节点会迫使系统进入一个循环，进而导致非获胜区域；但如果存在通往终止节点的路径，系统节点就可以打破循环。设 $s_k^{\mathcal{P}}$ 是一个可以选择退出循环的节点，$s_{k+1}^{\mathcal{P}}$ 是通往终止节点的后继节点，$\mathcal{S}_{\text{loop}} = \{s_k^{\mathcal{P}}, s_{k'+1}^{\mathcal{P}}, \ldots, s_{k'+n}^{\mathcal{P}}\}$ 是循环中的一串节点。在 $s_k^{\mathcal{P}}$ 处的最小总收益通过取通往终止节点的最短路径来更新，即 $U_i(s_k^{\mathcal{P}}) = U_{i-1}(s_{k+1}^{\mathcal{P}}) + W^{\mathcal{P}}(s_k^{\mathcal{P}}, a, s_{k+1}^{\mathcal{P}})$，而节点 $s_{k'+1}^{\mathcal{P}}$ 处的总收益等于 $s_k^{\mathcal{P}}$ 处的总收益与边权重之和，即 $U_i(s_k^{\mathcal{P}}) \oplus \{W^{\mathcal{P}}(s_{k'+n}, a, s_k) + \sum_{i=1}^{n-1} W^{\mathcal{P}}(s_{k'+i}, a, s_{k'+i+1})\}$，其中 $\oplus$ 是闵可夫斯基和。

取其后继节点的总收益的并集，可得，

$$\begin{aligned} U_{i+1}(s_k^{\mathcal{P}}) &= U_i(s_k^{\mathcal{P}}) \cup U_i(s_{k'+1}^{\mathcal{P}}) \\ &= U_i(s_k^{\mathcal{P}}) \cup \big(U_i(s_k^{\mathcal{P}}) \oplus \\ &\quad \{W^{\mathcal{P}}(s_{k'+n}, a, s_k) + \sum_{i=1}^{n-1} W^{\mathcal{P}}(s_{k'+i}, a, s_{k'+i+1})\}\big) \\ &= U_i(s_k^{\mathcal{P}}) \end{aligned}$$

由于权重之和均为正，且总收益严格大于 $U_i(s_k^{\mathcal{P}})$，并集操作会选取占优总收益 $U_i(s_k^{\mathcal{P}})$。因此，即使存在循环，总收益仍会单调递减。

B 命题证明1

证明。 命题 1 的证明依赖于以下引理。

引理 5. 从初始状态 $s_0^{\mathcal{P}}$到接受状态 $s_t^{\mathcal{P}}$的最大步数为 $|S^{\mathcal{P}}| - 1$。

证明。 如果初始状态位于获胜区域中，则始终存在一条从初始状态到接受状态的路径。获胜策略会选择最短路径，且绝不经过循环。因此，每个节点最多会被访问一次。由此，策略可采取的最大步数（边数）受节点数 $|S^{\mathcal{P}}| - 1$限制。

从 $F_{\mathcal{P}}$ 开始，从终止状态到初始状态逐个节点应用该方法，会遍历乘积博弈中的所有节点和边，因此 $|S^{\mathcal{P}}| + |E^{\mathcal{P}}|$ 在每次迭代中。根据引理5，每个状态都必须在距 $|S^{\mathcal{P}}| - 1$ 步之内从终止状态到达。因此，通过重复该过程$|S^{\mathcal{P}}| - 1$ 次，所有路径都会被遍历，并将所有路径的代价纳入考虑。

注释

1. https://youtu.be/TU8MhPBDBBs

参考文献

Araki B、Vodrahalli K、Leech T、Vasile CI、Donahue MD 和 Rus DL（2019）利用逻辑自动机学习规划。载于：机器人学：科学与系统基金会。Baier C 和 Katoen JP（2008）模型检测原理。剑桥，马萨诸塞州：麻省理工学院出版社。Basset N、Kwiatkowska M、Topcu U 和 Wiltsche C（2015）多目标长期随机博弈的策略综合。载于：系统构造与分析工具和算法国际会议。施普林格，第256–271页。Bhatia A、Kavraki LE 和 Vardi MY（2010）面向时序目标的基于采样的运动规划。载于：*2010 年 IEEE 国际机器人与自动化会议*。IEEE，第2689–2696页。Camacho A、Icarte RT、Klassen TQ、Valenzano RA 和 McIlraith SA（2019a）线性时序逻辑及其他：强化学习中奖励函数规范的形式语言。载于：*IJCAI*，第19卷，第6065–6073页。

Camacho A、Toro Icarte R、Klassen TQ、Valenzano R 和 McIlraith SA（2019b）线性时序逻辑及其他：强化学习中奖励函数规范的形式语言。载于：国际人工智能联合会议。第6065–6073页。Chatterjee K、Randour M 和 Raskin JF（2012）多维定量目标的策略综合。载于：国际并发理论会议。施普林格，第115–131页。Chen T、Forejt V、Kwiatkowska M、Simaitis A 和 Wiltsche C（2013a）关于多目标随机博弈。载于：国际计算机科学数学基础研讨会。施普林格，第266–277页。Chen T、Kwiatkowska M、Simaitis A 和 Wiltsche C（2013b）多目标随机博弈的综合：在自主城市驾驶中的应用。载于：国际系统定量评估会议。施普林格，第322–337页。De la Higuera C（2010）语法推断：学习自动机与文法。剑桥大学出版社。Fainekos GE、Kress-Gazit H 和 Pappas GJ（2005）移动机器人时序逻辑运动规划。载于：*2005*年*IEEE*机器人与自动化国际会议论文集。IEEE，第2020–2025页。He K、Lahijanian M、Kavraki E Lydia 和 Vardi Y Moshe（2019a）面向基于代价的反应式综合的操作领域自动抽象。*IEEE*机器人与自动化快报 4(2): 285–292。He K、Lahijanian M、Kavraki LE 和 Vardi MY（2015）迈向具有时序逻辑规范的操作规划。载于：机器人与自动化国际会议。IEEE，第346–352页。He K、Lahijanian M、Kavraki LE 和 Vardi MY（2017a）资源约束下有限任务的反应式综合。载于：*2017*年*IEEE/RSJ*智能机器人与系统国际会议（*IROS*）。IEEE，第5326–5332页。He K、Lahijanian M、Kavraki LE 和 Vardi MY（2017b）资源约束下有限任务的反应式综合。载于：智能机器人与系统国际会议（*IROS*）。加拿大不列颠哥伦比亚省温哥华：IEEE，第5326–5332页。He K、Wells AM、Kavraki LE 和 Vardi MY（2019b）有限时域任务的高效符号反应式综合。载于：*2019*年国际机器人与自动化会议（*ICRA*）。IEEE，第8993–8999页。Hussein A、Gaber MM、Elyan E 和 Jayne C（2017）模仿学习：学习方法综述。*ACM*计算调查（*CSUR*）50(2)：1–35。Jha S、Tiwari A、Seshia SA、Sahai T 和 Shankar N（2017）Telex：仅用正例进行信号时序逻辑被动学习。载于：国际运行时验证会议。施普林格，第208–224页。Kress-Gazit H、Fainekos G 和 Pappas GJ（2007）Where's Waldo? 基于传感器的时序逻辑运动规划。载于：机器人与自动化国际会议。意大利罗马：IEEE，第3116–3121页。Kress-Gazit H、Lahijanian M 和 Raman V（2018）机器人综合：机器人行为的保证与反馈。《控制、机器人学与自主系统年度评论》1：211–236。DOI: 10.1146/annurev-control-060117-104838。Kupferman O 和 Vardi MY（2001）安全属性的模型检测。系统设计形式化方法19：291–314。Lahijanian M、Kloetzer M、Itani S、Belta C 和 Andersson S（2009）在机器人城市类环境（RULE）中自动部署自动驾驶汽车。载于：机器人与自动化国际会议。日本神户：IEEE，第2055–2060页。

Lahijanian M, Maly MR, Fried D, Kavraki LE, Kress-Gazit H 和 Vardi MY（2016年）不确定环境中具有部分满足保证的迭代时序规划。 *IEEE*机器人学汇刊 32(3): 538–599。 DOI:10.1109/TRO. 2016.2544339。Li X, Serlin Z, Yang G 和 Belta C（2019年）一种面向机器人规划中可解释强化学习的形式化方法。 《科学·机器人学》4(37)。Li X, Vasile CI 和 Belta C（2017年）带有时序逻辑奖励的强化学习。载于： *2017*年*IEEE/RSJ*智能机器人与系统国际会议。IEEE，第3834–3839页。Muvvala K, Amorese P 和 Lahijanian M（2022年）让我们协作：面向机器人操作的基于遗憾的反应式综合。载于：*2022*年机器人与自动化国际会议（*ICRA*）。第4340–4346页。Muvvala K 和 Lahijanian M（2023年）用于有限任务的定量反应式综合的高效符号方法。载于： *2023*年*IEEE/RSJ*智能机器人与系统国际会议（*IROS*）。IEEE，第8666–8672页。Muvvala K, Wells A, Lahijanian M, Kavraki L 和 Vardi M（2024年）交互式操作领域中的随机博弈。载于：*2024*年*IEEE*机器人与自动化会议（*ICRA*）。日本横滨：IEEE。DOI:10.1109/ICRA57147.2024. 10611623。网址 https://arxiv.org/abs/2403.04910。Ng AY 和 Russell SJ（2000年）逆强化学习算法。载于：*ICML*，第1卷。第2页。Paraschos A, Daniel C, Peters J 和 Neumann G（2013年）概率运动基元。 *Neurips* 。Puri S 和 Prasad SK（2013年）用于GIS多边形叠加处理的高效并行与分布式算法。载于： *2013*年*IEEE*并行与分布式处理国际研讨会、专题讨论会及博士生论坛。IEEE，第2238–2241页。Ramachandran D 和 Amir E（2007年）贝叶斯逆强化学习。载于：*IJCAI*，第7卷。第2586–2591页。Ravichandar H, Polydoros AS, Chernova S 和 Billard A（2020年）从示范中进行机器人学习的最新进展。 《控制、机器人学与自主系统年度评论》3: 297–330。Sastry V, Janakiraman T 和 Mohideen SI（2005年）用于计算多目标最短路径问题中一组帕累托最优路径的新型多项式时间算法。 《计算机数学国际期刊》 82(3): 289–300。Schaal S（2006年）动态运动基元——人类运动控制与人形机器人学的框架。载于：《动物与机器的自适应运动》。施普林格，第261–280页。Shah AJ, Kamath P, Li S 和 Shah JA（2018年）基于示范的时序任务规范贝叶斯推断。载于： 神经信息处理系统基金会 Sutton RS 和 Barto AG（2018年）《强化学习：导论》。MIT出版社。Vazquez-Chanlatte M, Deshmukh JV, Jin X 和 Seshia SA（2017年）时间序列数据的逻辑聚类与学习。载于：《计算机辅助验证》。施普林格，第305–325页。Vazquez-Chanlatte M, Jha S, Tiwari A, Ho MK 和 Seshia S（2018年）从示范中学习任务规范。载于： *NeurIPS*，第31卷。

Verwer S 和 Hammerschmidt CA（2017年）Flexfringe：被动自动机学习软件包。载于：国际软件维护与演化会议（*ICSME*）。IEEE，第638–642页。Watanabe K, Renninger N, Sankaranarayanan S 和 Lahijanian M（2021年）面向安全约束规划的概率规范学习。载于：*2021*年*IEEE/RSJ*智能机器人与系统国际会议（*IROS*）。IEEE，第6558–6565页。Wulfmeier M, Ondruska P 和 Posner I（2015年）最大熵深度逆强化学习。*arXiv:1507.04888*。Xu Z, Saha S, Hu B, Mishra S 和 Julius AA（2018年）面向半自主机器人的咨询式时序逻辑推断与控制器设计。*IEEE*自动化科学与工程学报16(1): 459–477。Ziebart BD, Maas AL, Bagnell JA 和 Dey AK（2008年）最大熵逆强化学习。载于：*AAAI*，第8卷。美国伊利诺伊州芝加哥，第1433–1438页。