arXiv:2304.14996v3 [cs.FL] 4 May 2023

最大化带随机时钟的矩形自动机中的可达概率

JoannaDelicaris[1][0000−0001−9455−4052]、Stefan Schupp[2][0000−0002−2055−7581]、Erika ´Abrah´am[3][0000−0002−5647−6134]和 Anne Remke[1][0000−0002−5912−4767]

[1] 威斯特法伦威廉大学，48149 明斯特，德国{joanna.delicaris,anne.remke}@uni-muenster.de[2] 维也纳工业大学，奥地利维也纳 stefan.schupp@tuwien.ac.at[3] 亚琛工业大学，德国亚琛 abraham@informatik.rwth-aachen.de

摘要：本文提出一种算法，通过历史依赖的预知调度器，最大化带随机时钟的矩形自动机的可达概率。该模型类别包含由时间引起的离散行为非确定性，以及动态行为中的非确定性。通过前向流管构造计算可达状态集后，我们采用后向细化来计算最大可达概率。通过一个可扩展模型展示了所提方法的可行性。

1 引言

混合自动机 [2] 是一种用于描述此类系统的建模形式：系统的演化将连续动力学与离散步交替结合。本研究考虑矩形自动机 [13], 的一个子类，并通过随机延迟为其引入随机性。在混合自动机中，随机延迟的持续时间可以通过两种方式测量：（i）通过语义隐式测量；或（ii）通过秒表和句法约束显式测量。前者对用户友好且直观，而后者则明确规定了建模形式的限制。

我们采用句法建模变体，并明确规定相应的建模限制。与 [7,20], 类似，我们使用秒表时钟对跳变上的随机延迟进行建模。我们提出一种算法，用于优化带随机时钟的矩形自动机（RAR）中的可达概率。

非确定性会自然产生，例如出现在并发系统中；在随机模型中，它通常通过概率方式消解 [17,1,4], ，而在非随机混合系统中，通常不会显式消解。近期，人们提出了依赖历史的调度器，用于在混合 Petri 网 [19] 以及带有随机时钟和紧急转移的奇异自动机中消解离散非确定性，方式可以是预知式的，也可以是非预知式的 [20]。预知调度器知道随机变量的到期时间，因此被认为比不知道这些时间的非预知调度器更强大。预知调度器模拟最坏／最佳情形，并给出概率的最大界。

将随机时钟加入矩形自动机时，难点在于正确处理连续非确定性，从而计算出正确的概率。我们提出一种测度驱动的反向计算，根据调度器到达目标集的能力，对无限多个调度器进行划分。因此，在通过预先计算的可达性树执行反向分析时，预知调度会对调度器进行符号化细化。最大化可达概率需要对所有可达且能够到达目标的状态取并集，这些状态的转移延迟已通过反向分析细化。为计算最优可达概率，需要将该并集投影到相应概率密度的积分域上，然后对联合概率密度进行多维积分。

对于有限次跳变，非初始化矩形自动机 [2,9] 的可达性是可判定的。因此，针对这一模型类别，流管构造能够计算出精确的可达状态集，例如使用多面体状态表示 [9]。随后，采用后向细化来消解固有的连续非确定性，从而最大化可达概率。因此，对于所考虑的模型形式，本文提出的分析方法除数值积分外是精确的。据我们所知，唯一一种无需通过概率方式消解非确定性、便能计算这一模型类别可达性的方法，是借助工具 ProHVer [12] 计算安全过近似。通过一个可扩展模型展示了我们方法的可行性，并通过 ProHVer 中计算的结果进行了验证。

相关工作。CEGAR 式抽象 [26] 使得模型检测方法得以应用于概率混合自动机。通过在跳变上引入离散概率分布（[24],[25]），对混合自动机的可判定子类进行扩展，同时保留了可达性的可判定性。文献 [16], 提出了一种概率定时自动机的扩展，其中采用连续分布重置，并通过随机化调度器对离散化状态空间上的离散非确定性进行消解。针对随机定时自动机（网络）的其他方法（[3]）则保留了以概率方式消解非确定性的做法。针对更一般类别的方法，要么依赖随机近似（也用于处理非确定性）（ [21]），要么结合离散化与随机性（ [15]）。已有若干方法提出将模型抽象为有限马尔可夫决策过程：文献 [23] 针对随机混合自动机，提出了对不可数状态离散时间随机过程进行抽象的方法，其中所有非确定性均以概率方式消解。文献 [5] 提出了一种抽象为区间马尔可夫决策过程的方法。这两种方法都具有随机核，因此无法与我们的工作进行比较。

[12] 中提出的分析通过预知方式解决离散非确定性，并通过安全过近似解决连续非确定性（参见 [26]）。 [11]中也针对随机定时自动机规定了相同的方法。

在 [6], 中，随机自动机的离散非确定性通过（非）预知方式进行消解，其中所有连续变量都是随机时钟。类似地， [19] 为混合 Petri 网引入了离散非确定性的调度，而 [20], 则为带随机时钟的奇异自动机引入了这种调度；在后一种情况下，无需细化的前向分析就足以计算最大可达概率。

贡献。我们提出：（i）带随机时钟的矩形自动机这一建模形式，它将离散和连续非确定性与随机延迟相结合；以及（ii）一种解析方法，用于计算预知调度器所诱导的最大可达概率。我们还提供了一项可行性研究，表明我们的计算具有很高的准确性。

论文结构。第 2 节介绍所考虑的模型类别。第 3 节说明最大概率的计算方法。第 4 节展示可行性研究，第 5 节对论文作总结。

2 带随机时钟的矩形自动机

令 $\mathbb{I}$ 表示所有闭区间的集合 $[a,b]$, $[a,\infty)$, $(-\infty,b]$, $(-\infty,\infty) \subseteq \mathbb{R}$，其端点为无穷或有理数 $a,b \in \mathbb{Q}$，并采用标准集合语义。

对于 distr : $\mathbb{R}_{\geq 0} \to [0,1] \subseteq \mathbb{R}$，令 supp(distr) $= \{v \in \mathbb{R}_{\geq 0} \mid \text{distr}(v) > 0\}$。若 distr 绝对连续且 $\int_0^\infty \text{distr}(v)dv = 1$，则称 distr 为连续分布。若 supp(distr) 可数且 $\sum_{v \in supp(distr)} f(v) = 1$，则称 distr 为离散分布。我们用 $\mathbb{F}_c$ 和 $\mathbb{F}_d$ 分别表示所有连续分布和离散分布构成的集合，并令 $\mathbb{F} = \mathbb{F}_c \cup \mathbb{F}_d$ 包含所有分布。

混合自动机 [2] 是一种用于描述系统的建模形式，其演化由连续动力学（流）与其间发生的离散步（跳变）共同构成。在本工作中，我们将范围限定在矩形自动机 [13]这一子类。

定义 1. 矩形自动机（RA）是一个元组 $R = (\text{Loc}, \text{Var}_C, \text{Inv}, \text{Init}, \text{Flow}_C, \text{Edge}_C)$，其中

– 一个有限的位置集合 Loc；– 一个有限的有序变量集合 $\text{Var}_C = \{x_1, \ldots, x_{d_C}\}$；对于 $\nu = (\nu_1, \ldots, \nu_{d_C}) \in \mathbb{R}^{d_C}$，我们定义 $\nu_{x_i} = \nu_i$；对于 $I = I_1 \times \ldots \times I_{d_C} \in \mathbb{I}^{d_C}$，我们设 $I_{x_i} = I_i$；– 函数 Inv、Init 和 Flow_C，其类型均为 Loc $\to \mathbb{I}^{d_C}$，分别为每个位置指定一个不变式、初始状态和流；我们称 $\text{Flow}_C(\ell)_x$ 为位置 $\ell \in$ Loc 中 $x \in \text{Var}_C$ 的速率；– 一个有限的非随机跳转集合 $\text{Edge}_C \subseteq$ Loc $\times \mathbb{I}^{d_C} \times \mathbb{I}^{d_C} \times 2^{Var_C} \times$ Loc，其中每个非随机跳转 $e = (\ell, \text{pre}, \text{post}, \text{reset}\ \ell') \in \text{Edge}_C$ 的源（目标）位置为 ℓ（ℓ'），且对于所有 $x \in \text{Var}_C \setminus \text{reset}$，$\text{pre}_x = \text{post}_x$，并且 post $\subseteq \text{Inv}(\ell')$；

R 是非阻塞的，当且仅当对每个位置 $\ell \in$ Loc 和每个变量 $x \in \text{Var}_C$：

– 如果 $\text{Inv}(\ell)_x$ 的下界为 $a \in \mathbb{Q}$（即其形式为 $[a,b]$ 或 $[a,\infty)$）且 $\text{Flow}_C(\ell)_x \cap \mathbb{R}_{<0} \neq \emptyset$，则存在一个非随机跳转 $e = (\ell, \text{pre}, \text{post}, \text{reset}\ \ell') \in$ 边$_C$，使得 $\{\nu \in \text{Inv}(\ell) \mid \nu_x = a\} \subseteq$ pre；– 如果 $\text{Inv}(\ell)_x$ 的上界为 $b \in \mathbb{Q}$（即其形式为 $[a,b]$ 或 $(-\infty,b]$），且 $\text{Flow}_C(\ell)_x \cap \mathbb{R}_{>0} \neq \emptyset$，则存在一个非随机跳转 $e = (\ell, \text{pre}, \text{post}, \text{reset}\ \ell') \in$ 边$_C$，使得 $\{\nu \in \text{Inv}(\ell) \mid \nu_x = b\} \subseteq$ pre。

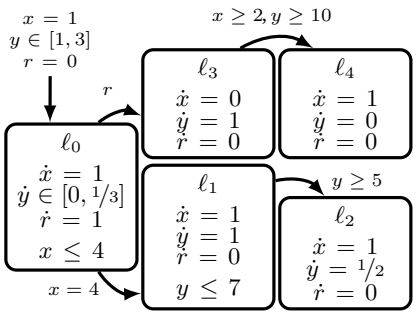


图1：带有 $\mathsf{Var}_R = \{r\}$ 的 RAR。

我们为非阻塞 RA 配备由随机时钟建模的随机延迟，随机时钟的行为类似秒表时钟，即其导数只能为 0（暂停）或 1（运行）。每个随机时钟 r 都关联一个连续分布，用于抽取随机变量 $S_{r,\ n}$，这些随机变量决定在随机时钟达到哪个值（到期时间）时应执行特定跳变。到期后，随机时钟会重置为 0，并使用一个新的随机变量来确定该随机时钟的到期时间下一个事件，从而得到一列随机变量 $S_{r,\ 0}, S_{r,\ 1}, S_{r,\ 2}, \ldots$ 对每个随机时钟 r 都如此。图1展示了一个示例。

定义 2. 带随机时钟的矩形自动机（RAR）是一个元组$\mathcal{A} =$(Loc, Var_C, Var_R, Distr, Inv , Init, Flow_C, Flow_R, Edge_C, Edge_R)，其中 (Loc, Var_C, Inv, Init , Flow_C, Edge_C) 是一个非阻塞 RA，且：

– 一个有限的有序随机时钟集合 $\mathsf{Var}_R = \{r_1, \ldots, r_{d_R}\}$；我们对 $\mu \in \mathbb{R}^{d_R}$ 和 $I = I_1 \times \ldots \times I_{d_R} \in \mathbb{I}^{d_R}$ 分别采用类似的记法 $\mu_{r_i} = \mu_i$ 和 $I_{r_i} = I_i$；– 一个函数 $\mathsf{Distr} : \mathsf{Var}_R \to \mathbb{F}_c$；– 一个函数 Flow_R: Loc $\to \{0, 1\}^{|Var_R|}$；– 一个有限集合 $\mathsf{Edge}_R \subseteq$ Loc $\times \mathsf{Var}_R \times$ Loc，包含随机跳转 $e =$(ℓ, r, ℓ′)，其源位置（目标位置）为 ℓ （ℓ'），随机时钟为 r，其中 (i) 任意两个随机跳转 $e, e' \in \mathsf{Edge}_R$, $e \neq e'$ 若具有相同的源位置 ℓ，则它们具有不同的随机时钟；(ii) 对所有位置 $\ell \in$ Loc 和所有随机时钟$r \in \mathsf{Var}_R$，若 $\mathsf{Flow}_R(\ell)_r = 1$，则存在某个 $\ell' \in$Loc，使得 (ℓ, r, ℓ′) $\in \mathsf{Edge}_R$；(iii) 对每个随机跳转 $(\ell, r, \ell') \in \mathsf{Edge}_R$，均有 $\mathsf{Inv}(\ell) \subseteq \mathsf{Inv}(\ell')$。

设 $\mathcal{A} =$(Loc, Var_C, Var_R, distr , Inv, Init , Flow_C, Edge_C, Edge_R)，其中 $\mathsf{Var}_C = \{x_1, \ldots, x_{d_C}\}$和 $\mathsf{Var}_R = \{r_1, \ldots, r_{d_R}\}$，为一个 RAR。状态 $\sigma =$(ℓ, ν, μ, s) $\in \mathcal{S} =$ Loc $\times \mathbb{R}^{d_C} \times \mathbb{R}^{d_R} \times \mathbb{R}^{d_R}$ of $\mathcal{A}$ 表示当前的位置 ℓ、连续变量的值 ν，以及随机时钟的值 μ 和到期时间 s。操作语义 [14] 通过让时间流逝或执行离散跳变（非随机或随机），来规定 $\mathcal{A}$中状态的演化：

$$\text{Flow}\ \frac{t \in \mathbb{R}_{\geq 0} \quad rate \in Flow_C(\ell) \quad \nu' = \nu + t \cdot rate \quad \nu' \in Inv(\ell) \quad \mu' = \mu + t \cdot Flow_R(\ell) \quad \forall r \in Var_R.\, \mu'_r \leq s_r}{(\ell, \nu, \mu, s) \xrightarrow{t, rate} (\ell, \nu', \mu', s)}$$

$$\text{Jump}_C\ \frac{e = (\ell, pre, post, reset, \ell') \in Edge_C \quad \nu \in pre \quad \nu' \in post \quad \forall x \in Var_C \setminus reset.\, \nu'_x = \nu_x \quad \nu' \in Inv(\ell')}{(\ell, \nu, \mu, s) \xrightarrow{e} (\ell', \nu', \mu, s)}$$

$$\text{Jump}_R\ \frac{e = (\ell, r, \ell') \in Edge_R \quad \mu_r = s_r \quad \mu'_r = 0 \quad s'_r \in supp(Distr(r)) \quad \forall r' \in Var_R \setminus \{r\}.\mu'_{r'} = \mu_{r'} \wedge s'_{r'} = s_{r'}}{(\ell, \nu, \mu, s) \xrightarrow{e} (\ell', \nu, \mu', s')}$$

对于 $\sigma \in \mathcal{S}$，令 EnabledJumps(σ) $= \{e \in \text{Edge}_C \cup \text{Edge}_R \mid \exists \sigma' \in \mathcal{S}.\, \sigma \xrightarrow{e} \sigma'\}$为在 $\sigma \in \mathcal{S}$中启用的跳变集合。我们还将使用 EnabledTime(σ) :=$\{(t,$ 速率) $\in \mathbb{R}_{\geq 0} \times \mathbb{R}^{d_C} \mid \exists \sigma' \in \mathcal{S}.\, \sigma \xrightarrow{t,\ rate} \sigma'\}$。

我们定义 $\rightarrow = (\bigcup_{t \in \mathbb{R}_{\geq 0}, rate \in \mathbb{R}^{d_C}} \xrightarrow{t,\ rate}) \cup (\bigcup_{e \in Edge_C \cup Edge_R} \xrightarrow{e})$。 $\mathcal{A}$的一条路径是由状态组成的有限序列 $\pi = \sigma_0 \xrightarrow{a_0} \sigma_1 \xrightarrow{a_1} \ldots$，满足 $\sigma_0 = (\ell_0, \nu_0, \mu_0, s_0)$、$\nu_0 \in$ 不变式(ℓ_0)，且对于所有 $i \in \mathbb{N}$， $\sigma_i \xrightarrow{a_i} \sigma_{i+1}$ ；如果 $\nu_0 \in$ 初始(ℓ_0)、$\mu_0 = 0 \in \mathbb{R}^{d_R}$，并且对于所有 $r \in \text{Var}_R$， $s_{0_r} \in$ supp(Distr (r))，则称其为初始路径。若存在一条初始路径通向某个状态，则该状态可达。用 Paths 表示所有路径的集合。

对于每个可达状态，都存在一条由延迟和跳变交替组成的路径，因为延迟的持续时间可以为零，且连续的延迟可以合并。即使连续延迟的速率不同，这一点仍然成立，因为流集合是凸的[4]:

引理 1。设 $\sigma_0 \xrightarrow{t_1,\ rate_1} \sigma_1 \xrightarrow{t_2,\ rate_2} \sigma_2$ 对于 $\sigma_0, \sigma_1, \sigma_2 \in \mathcal{S}$，其位置为 ℓ，$t_1, t_2 \in \mathbb{R}_{\geq 0}$，速率$_1$, 速率$_2 \in$流 $_C(\ell)$。那么存在速率 $\in$流 $_C(\ell)$，使得 $\sigma_0 \xrightarrow{t_1+t_2,\ rate} \sigma_2$。

路径的持续时间定义为其各步骤持续时间之和，其中跳变视为瞬时发生：dur (σ) $= 0$，dur ($\pi \xrightarrow{e} \sigma$) $=$dur (π)，且 dur($\pi \xrightarrow{t,\ rate} \sigma'$) $=$dur (π) $+ t$。路径的跳转深度定义为其中跳变的数量。当且仅当对于所有 $t \in \mathbb{R}_{\geq 0}$，都存在一个 $k \in \mathbb{N}$，使得所有跳转深度至少为 k 的初始路径的持续时间至少为 t 时，我们称 RAR 为无芝诺。我们采用了不同于标准定义的定义；标准定义要求在有限时间内只能发生有限次跳变。我们的定义保证了对于每个时间界限，每条路径上的跳变次数都有一个具体上界。下文中，我们假设所有讨论的模型均为无芝诺。

RAR 允许存在：(i) 在初始状态选择上的初始非确定性；(ii) 时间非确定性，即时间可以流逝，但在整个流动过程中也始终有跳变可用；以及 (iii) 速率非确定性，即连续变量可以按不同速率演化。除了这些连续类型的非确定性外，我们还考虑 (iv) 离散非确定性，即多个不同跳变同时可用。我们使用预知调度器来消解非确定性。预知调度器不仅能完整获知历史，还能获知所有随机时钟未来的到期时间，正如 [20] 中所介绍的那样。预知调度虽然看起来不切实际，但非常适合进行最坏情况分析，尤其是在将不可控不确定性建模为非确定性的情况下。

定义 3.（预知的、依赖历史的）调度器是一个函数 $\mathfrak{s}$: Paths$\rightarrow \mathbb{F}$，它为每条路径 $\pi = \sigma_0 \xrightarrow{a_1} \ldots \xrightarrow{a_n} \sigma_n \in$Paths 指定一个分布 distr $= \mathfrak{s}(\pi)$。如果 $n \geq 1$ 且 $a_n \in$ Edge$_C \cup$Edge$_R$ 是一条边，则 distr 是连续分布，且 supp(distr) $\subseteq$EnabledTime (σ_n)；否则，distr 是离散分布，且 supp(distr) $\subseteq$ EnabledJumps(σ_n)。调度器集合记作 $\mathfrak{S}$。

我们最大化有时间界限的可达概率，即沿初始路径 π 到达目标状态集合 $\mathcal{S}^{goal} \subseteq \mathcal{S}$，且 dur($\pi$) $\leq t_{max}$ 的概率。

[4] 引理 1 的证明见附录 A。

通过调度器集合 $\mathfrak{S}$ 消解 RAR $\mathcal{A}$ 中的非确定性，会得到一个可达概率区间 $[p^{\mathfrak{S}}_{min}(\mathcal{S}^{goal}, t_{max}), p^{\mathfrak{S}}_{max}(\mathcal{S}^{goal}, t_{max})]$，其上下界分别称为最小值和最大值。我们将 $\mathfrak{S}_{goal} \subseteq \mathfrak{S}$ 定义为这样一个调度器集合：其中的调度器沿初始路径 π 到达 $\mathcal{S}^{goal}$，且 $\mathrm{dur}(\pi) \leq t_{max}$，并产生 $p^{\mathfrak{S}}_{max}$。令 $\mathcal{V}^{\mathfrak{s}} \subseteq \mathbb{R}^{d_R}$ 表示所有允许调度器 $\mathfrak{s}$ 到达 $\mathcal{S}^{goal}$ 的随机变量的样本值。由此得到以下定义：

定义 4。预言式最大可达概率为：

$$p^{\mathfrak{S}}_{max}(\mathcal{S}^{goal}, t_{max}) = \int_{\bigcup_{\mathfrak{s} \in \mathfrak{S}_{goal}} \mathcal{V}^{\mathfrak{s}}} G(\mathbf{s}) \; d\mathbf{s}, \tag{1}$$

其中，$G(\mathbf{s}) = \prod_{r_n} \mathrm{Distr}(r)$ 是所有随机延迟 r_n 和随机时钟 r 的联合概率密度函数。

请注意，由于随机变量相互独立，$G(\mathbf{s})$ 等于概率密度函数 $\mathrm{Distr}(r)$ 的乘积。

3 最大可达概率的计算

RAR 中固有的连续非确定性导致了不可数多个选择，进而产生不可数多个调度器。我们提出一种由测度驱动的状态空间构造，该构造根据无限集合中连续调度器到达目标状态集合的能力对其进行划分。我们指出，我们的模型类别允许重置连续变量，但我们的方法目前尚不支持这一点。

第 3.1 节介绍前向流管构造，第 3.2 节介绍反向细化。第 3.3 节提取样本域，第 3.4 节计算最大预言式可达概率。

3.1 前向流管构造

为计算矩形自动机的可达性，[2], 提出了流管构造方法，该方法可对预定义时间界限内所有可达状态进行几何表示。将此方法应用于 RAR 时，我们通过从操作语义中移除对样本值 s 的所有约束，忽略随机性。所得自动机是一个常规矩形自动机，其中，原自动机中由随机时钟 r 引发的第 n 个随机延迟被视为一个连续变量 r_n。将每个随机时钟r 替换为每种可能延迟对应的一个连续变量，相当于展开该自动机（参见图 2）。因此，展开后的矩形自动机的变量集 Var_R 包含原自动机中每个随机时钟r 对应的连续变量 r_0、r_1、.....，且 d_R 为所有随机延迟的数量。

下文中，我们省略状态 $\sigma = (\ell, \nu, \mu, s)$ 中采样得到的值 s。为简化起见，我们还将连续变量 ν 和随机时钟 μ 的赋值合并起来，将 $(\ell, \mathcal{V})$ 称为状态集，其中 $\mathcal{V}$ 是一个赋值集合，而 ν_x 指变量 $x \in \mathrm{Var}_C \cup \mathrm{Var}_R$ 的赋值。尽管从形式上来说，我们

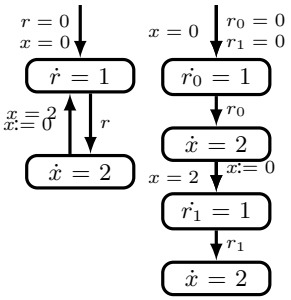
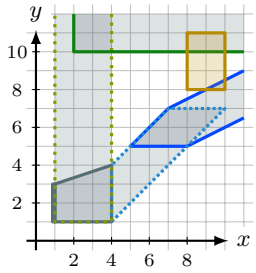
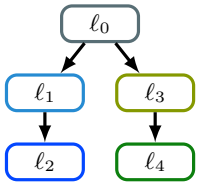


图2：RAR 在 $\mathrm{Var}_R = \{r\}$，$t_{max} = 3$ 下的展开；未注明时，速率均为零。



(a) 流管与目标。



(b) 可达树。

图3：图1所示 RAR 的流管（二维）、目标集及可达树示意图。

处理的是状态集 $(\ell, \mathcal{V})$，但为便于阅读，我们的记号仅限于赋值集合 $\mathcal{V}$。我们将赋值集合称为线段，并将状态集的相应位置称为对应位置。

我们执行前向流管构造，即从初始状态出发，交替计算前向时间闭包和跳转后继，直到到达预定义的 t_{max}。ℓ 中 $\mathcal{V}$ 的前向时间闭包 $\mathrm{T}^+_\ell(\mathcal{V})$（参见 [2]）表示从入口状态集可达的状态集合，即计算在位置 ℓ经过一段时间延迟后可达的状态。跳变由一步关系 $\mathrm{D}^+_e(\mathcal{V})$表示，它定义了通过选择跃迁 $e \in \mathrm{Edge}_C \cup \mathrm{Edge}_R$，从赋值 $\mathcal{V}$ [2]可达的赋值集合。

我们的计算依赖于通过凸多面体表示状态，通常采用 H-表示，将其定义为多个半空间的交集。某些计算需要使用 V-表示，它被定义为一组顶点的凸包（对于无界多面体，则是凸包与凸锥的组合） [27]。

随后，前向流管构造会计算所有可达线段，并根据其出现顺序，将这些线段及其对应的位置一并存储在可达树 $\mathcal{R}$ ac- cording 中。我们定义可达树如下：

定义 5. 对于矩形自动机 R 和时间界限 t_{max}，可达树是一个元组 $\mathcal{R} = (N, E)$，其节点为符号状态集合 $(\ell, \mathcal{V}) \in N$，边$e \in E = N \times$(边$_C \cup$ 边$_R$) $\times N$ 标注有跳变。对于每个状态$\sigma = (\ell, \nu) \in \mathcal{S}$，当且仅当 σ 在时间 t_{max} 之前于 R 中可达时，存在一个节点 $n = (\ell, \mathcal{V}) \in N$，使得 $\sigma \in \mathcal{V}$。

在纯混合情形下进行定性分析时，只要计算得到的线段与目标状态集的交集非空，流管构造便会立即结束。进行定量可达性分析时，我们必须将流管构造完成至时间界限 t_{max}，并收集所有通向目标状态的轨迹，以便充分解析系统中存在的非确定性。将流管计算至 t_{max} 后，通过将得到的线段与目标集求交来确定可达性。我们定义

目标状态为 $\mathcal{S}^{goal} = (L^{goal}, \mathcal{V}^{goal})$，其中 L^{goal} 是一个位置集合，$\mathcal{V}^{goal}$ 是一个定义为凸多面体的赋值集合，其约束仅涉及连续变量 $x \in \text{Var}_C$。因此，$\mathcal{V}^{goal}$ 在随机时钟 $r \in \text{Var}_R$ 所对应的维度上无界。

我们将可达目标状态集合定义为 $\mathcal{S}^{goal}_{reach}$，其中包含所有能够通过可达树中的某条轨迹到达的目标状态集子集：

$$(\ell, \mathcal{V}) \in \mathcal{S}^{goal}_{reach} \Leftrightarrow \ell \in L^{goal} \wedge \exists (\ell', \mathcal{V}') \in \mathcal{R}. \mathcal{V}' \cap \mathcal{V}^{goal} = \mathcal{V} (\neq \emptyset) \wedge \ell' = \ell.$$

我们使用 $\mathcal{V}^i$ 和 $\mathcal{S}^i = (\ell, \mathcal{V}^i)$ 指代这些目标状态子集，并使用索引 i 指代可达树中通向 $\mathcal{S}^i$ 的轨迹（参见图 4c）。我们将收集所有轨迹索引 i 的集合定义为 $I_{\mathcal{S}}$。随后，所有线段 $\mathcal{V}^i$ 都作为后向细化的输入（参见第 3.2 节）。

运行示例。位置 ℓ_0 中初始集合的前向时间闭包（参见图1）对应于图 3a 中灰色实线所示的线段。从位置 ℓ_0 移动到 ℓ_3 对应于随机时钟 r 到期，它表示自动机中唯一存在的随机延迟。在位置 ℓ_2 中，只有 y 演化（绿色虚线边框）。对于满足 $x \geq 2$ 和 $y \geq 10$ 的状态，可以跃迁到位置 ℓ_4（深绿色实线边框）。此时，y 停止，只有 x 演化。或者，对于 $x = 4$，也可以从 ℓ_0 移动到 ℓ_1（蓝色虚线边框）。只要 $y \leq 7$ 成立，所有满足 $y \geq 5$ 的状态都能到达位置 ℓ_2。这会导致 ℓ_1 和 ℓ_2 对应的线段重叠（深蓝色实线边框）。目标状态（黄色边框）由 $\mathcal{S}^{goal} = \{ (\ell, \nu) \in \mathcal{S} \mid \ell \in Loc \wedge \nu_x \in [8, 10] \wedge \nu_y \in [8, 11] \}$ 定义。图 3b 展示了可达树 $\mathcal{R}$。

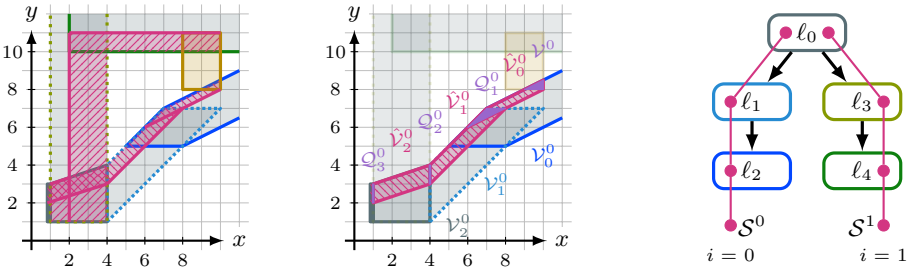
目标既可以通过位置 ℓ_1, ℓ_2 到达，也可以通过位置 ℓ_3, ℓ_4到达。因此，将前向流管段与目标集求交后会得到两条轨迹 $i = 0,1$，并生成状态集 $\mathcal{S}^0 = (\ell_2, \mathcal{V}^0)$ 和 $\mathcal{S}^1 = (\ell_4, \mathcal{V}^1)$，其中：

$$\mathcal{V}^0 = \{\nu \in \mathbb{R}^3 \mid \nu_x \leq 10 \wedge \nu_y \geq 8 \wedge \nu_y \leq 3.5 + 1/2 \cdot \nu_x \wedge \nu_r = 3\}$$
$$\mathcal{V}^1 = \{\nu \in \mathbb{R}^3 \mid \nu_x \in [8, 10] \wedge \nu_y \in [10, 11] \wedge \nu_r \in [0, 3]\}.$$

3.2 后向细化

从每个 $\mathcal{V}^i \subseteq \mathcal{V}^{goal}$ 出发，我们沿可达树执行后向计算，根据预知最大调度器细化状态集，直到到达树的根节点。我们通过为轨迹 i上的所有前向流管段计算细化线段和中间目标段来进行后向细化。后向细化的结果由沿可达树的轨迹 i 中所有细化线段和 $\mathcal{V}^i$ 组成，这些内容恰好包含可达树中允许到达目标的部分。

反向传播依赖于反向时间闭包 $\text{T}^-_\ell(\mathcal{V})$ 和反向单步关系 $\text{D}^-_e(\mathcal{V})$ 的定义（参见 [2]）。与前向时间闭包类似，$\text{T}^-_\ell(\mathcal{V})$ 通过倒推该位置中的时间延迟，计算位置 ℓ 中所有能够到达 $\mathcal{V}$ 中赋值的有效状态。$\text{D}^-_e(\mathcal{V})$ 则逆转跃迁 e 导致 $\mathcal{V}$ 的效果，该跃迁由守卫条件和重置定义。



(a) 向后细化-化（）自 goal (▭) 关于轨迹 $i = 0$，1。

(b) 交替 segments$\mathcal{Q}_k^0$ and $\hat{}\mathcal{V}_k^0$ on 轨迹 $i = 0$。

(c) 带有轨迹的可达树 $i = 0$，1 通向 $\mathcal{S}^i$，经由细化片段 $\hat{}\mathcal{V}_k^i$, $\mathcal{V}^i$。

图 4：后向细化。

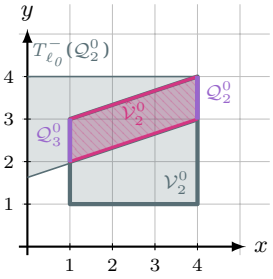
从每个线段 $\mathcal{V}^i$ 开始，根据相应位置 ℓ 的活动性计算反向时间闭包 $\mathrm{T}_\ell^-(\mathcal{V})$。由此得到一个无界锥，其中包含所有可从位置 ℓ 中任意状态到达 $\mathcal{V}^{goal}$ 的状态。我们将相应的预计算前向线段 $\mathcal{V}_k^i$ 定义为可达树中轨迹 i 上的流管线段；沿着 $\mathcal{V}^i$ 的反向方向针对 $\mathcal{V}^i \subset \mathcal{V}_0^i$ 执行到第 k 步时会遇到该线段（参见图 4b）。随后，将它与无界反向时间闭包相交，以将状态集限制为那些确实能够通过最大化调度器在混合自动机中到达的状态（参见图 5a）。由此得到所谓的细化片段 $\hat{}\mathcal{V}_k^i$，其中包含轨迹 i 上从位置 ℓ （第 k 步）能够到达目标集的所有状态：

定义 6。给定线段 $\mathcal{V}_k^i$ 内的中间目标片段 $\mathcal{Q}_k^i$ 及其对应的位置 ℓ，轨迹 i 上的细化片段 $\hat{}\mathcal{V}_k^i$ 为 $\hat{}\mathcal{V}_k^i = \mathrm{T}_\ell^-(\mathcal{Q}_k^i) \cap \mathcal{V}_k^i$。轨迹 i 上的第一个中间目标片段 $\mathcal{Q}_0^i$ 就是 $\mathcal{V}^i$ 本身。

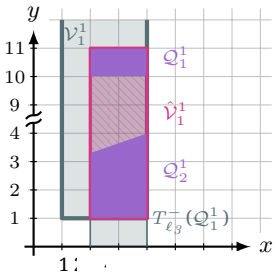
图 5 展示了根据中间目标片段计算出的细化片段，对应于运行示例。给定一个从 ℓ_p 到 ℓ 、连接线段 $\mathcal{V}_{k+1}^i$ 和 $\mathcal{V}_k^i$ 的跃迁 e，我们计算细化片段 $\hat{}\mathcal{V}_k^i$ 的反向单步关系。通过与下一个前向线段 $\mathcal{V}_{k+1}^i$ 求交，确保所得线段 $\mathcal{Q}_{k+1}^i$ 是 $\mathcal{V}_{k+1}^i$ 的子集。这将作为下一计算步骤的中间目标片段。

定义 7。对于细化片段 $\hat{}\mathcal{V}_k^i$ 及其对应的位置 ℓ，轨迹 i 上的下一个中间目标片段 $\mathcal{Q}_{k+1}^i$ 为 $\mathcal{Q}_{k+1}^i = \mathrm{D}_e^-(\hat{}\mathcal{V}_k^i) \cap \mathcal{V}_{k+1}^i$。如果 $\hat{}\mathcal{V}_k^i$ 对应于初始位置 ℓ_0（即 $\mathrm{Init}(\ell_0) \neq \emptyset$），则在求交时使用 ℓ_0 的初始集合而非 $\mathcal{V}_{k+1}^i$，即 $\mathcal{Q}_{k+1}^i = \mathrm{D}_e^-(\hat{}\mathcal{V}_k^i) \cap \mathrm{Init}(\ell_0)$。

运行示例。运行示例的后向细化如图 4a 和图 4b 所示。图 4c 展示了两条轨迹 $i = 0$、1。从线段 $\mathcal{V}^0$ 和 $\mathcal{V}^1$ 开始，可以迭代计算出细化线段 $\hat{}\mathcal{V}_k^0$ 和 $\hat{}\mathcal{V}_k^1$。图 5 展示了每条轨迹上的一个细化线段。细化线段 $\hat{}\mathcal{V}_2^0$ 和下一个中间目标片段 $\mathcal{Q}_3^0$ （参见图 5a ）可以由 $\mathcal{Q}_2^0$ 计算得出。除了图 5a 中可见的约束外，$\hat{}\mathcal{V}_2^0$



(a) 细化线段 $\hat{\mathcal{V}}^0_2$（）和中间的目标线段 $\mathcal{Q}^0_2, \mathcal{Q}^0_3$（）上轨迹 $i = 0$.



(b) 细化线段 $\hat{\mathcal{V}}^1_1$（）及中间目标线段 $\mathcal{Q}^1_1$，$\mathcal{Q}^1_2$（）在轨迹 $i = 1$。

图 5：详细的后向细化步骤。

随后还包含约束 $\nu_r = \nu_x - 1$，该约束源自 $\nu_x = 1$ 的初始赋值。注意，$\mathcal{Q}^0_3$ 是通过与 Init (ℓ_0) 求交计算得出的。

细化线段 $\hat{\mathcal{V}}^1_1$ 的中间目标片段 $\mathcal{Q}^1_2$（参见图5b）由 $\mathcal{Q}^1_1$ 计算得出。同样，$\hat{\mathcal{V}}^1_1$ 和 $\mathcal{Q}^1_2$ 包含约束 $\nu_r = \nu_x - 1$。请注意，由于 $\nu_x \in [2, 4]$ 也成立，ν_r 也隐式地受到限制，必须包含在 $[1, 3]$ 中。这些计算步骤也可见于附录 B。

3.3 提取样本域

后向细化会产生细化片段 $\hat{\mathcal{V}}^i_k$，据此可按如下方式提取所有随机变量的样本域。细化片段包含关于所有变量之间相互依赖关系的信息，以及在 $\mathcal{V}^i$ 中通向目标的赋值信息。每个随机变量 $S_{r,\,n}$ 的样本域，取自与随机时钟 r 第 n 次到期相对应的线段。

多胞体 $\mathcal{P}^i \subset \mathbb{R}^{d_R}$ 汇集了沿轨迹 i 通向目标的样本值。按正向遍历轨迹时，我们针对每条线段，根据允许沿轨迹 i（在可达树中）迈出下一步的随机时钟赋值约束，推导出有关样本域的信息。因此，我们收集轨迹 i 上每个随机延迟的所有有效取值。

对于每条线段，样本的约束推导如下。首先，将线段投影到随机维度上。其次，收集所有随机延迟的信息，这些延迟可能已经到期、将在轨迹的下一步到期，或者尚未到期。(i) 已经到期的随机延迟无法提供任何新信息，因此无需再次考虑。(ii) 轨迹中对应随机延迟到期的步骤会对样本域产生（上界）约束。可达树中的每条边都恰好映射到自动机中的一次跳转。若为随机跳转，该步骤会产生恰好一个随机时钟的上界。(iii) 为了纳入有关未来到期的信息，必须提升所有其他随机延迟的上界。

定义 8。我们将多胞体 $P \subseteq \mathbb{R}^{d_R}$ 中变量 r 的提升定义为$\Lambda_r(P) := \{(s_1, \ldots, s_r + c, \ldots, s_{d_R}) \in \mathbb{R}^{d_R} \mid (s_1, \ldots, s_r, \ldots, s_{d_R}) \in P \wedge c \in [0, \infty)\}$。

在样本域上迭代收集这些约束，得到一个多胞体 $\mathcal{P}^i$，其中包含所有允许沿轨迹 i运行的样本值。要计算最大可达概率，必须将所有到达目标的可能情况都纳入积分。这相当于对所有 $i \in I_{\mathcal{S}}$取 $\mathcal{P}^i$的并集。

Summarizing, this leads to a polytope $\mathcal{P}_{max} = \bigcup_{i \in I_{\mathcal{S}}} \mathcal{P}^i$.

运行示例。随机延迟 r 在轨迹 0 上不会到期，因此会针对 s_r 迭代收集样本域上的下界约束，从而得到 $\mathcal{V}^0$中最强的下界约束。将 $\mathcal{V}^0$ 投影到 $\mathbb{R}^{d_R}$ 上，并将所得多胞体提升到 r维，即可推导出约束 $s_r \geq 3$ 。

在轨迹 1 上，随机时钟的到期对应于从 ℓ_0 到 ℓ_3的跃迁，即 r 即将在$\hat{} \mathcal{V}_2^1$线段中到期。因此，可从$\hat{} \mathcal{V}_2^1$推导出约束 $s_r \geq 1$ 和 $s_r \leq 3$，即 $\mathcal{P}^1 = [1, 3]$。由此得到 $\mathcal{P}_{max} = [3, \infty) \cup [1, 3] = [1, \infty)$，其中包含所有使目标可达的随机变量 S_r 的取值：对于 $s_r \geq 3$，可通过轨迹 0 到达；对于 $s_r \in [1, 3]$，可通过轨迹 1 到达。

3.4 最大预知可达概率

为了以预知方式最大化连续非确定性，我们根据潜在的无限多个预知调度器 $\mathfrak{S}_{goal}$ 到达目标的能力，对其进行划分。后向细化返回可达树中通向目标状态 $\mathcal{S}^{goal}$（具体而言是状态集 $\mathcal{S}^i \in \mathcal{S}^{goal}_{reach}$）的片段，从而可以提取样本域。此过程纳入了关于随机变量未来到期时间的信息，进而产生预知调度器。

可达性根据状态集 $\mathcal{S}^i$以及通过轨迹 i可达的状态，为调度器集合 $\mathfrak{S}_{goal}$定义了一种等价关系。由此得到等价类 $[\mathfrak{s}]_i$，其中包含所有能够到达 $\mathcal{S}^i$的调度器 $\mathfrak{s}$ 。因此，$\bigcup_{i \in I_{\mathcal{S}}} [\mathfrak{s}]_i = \mathfrak{S}_{goal}$。借助这种等价关系，我们能够处理不同类型的（连续）非确定性。下面将针对转移上的非确定性时间延迟、矩形流集合和冲突迁移对此进行说明。

初始与时间非确定性。在不同时间点执行跃迁（从 ℓ_p 到 ℓ）会得到集合中不同的状态，而这些状态会进入目标位置 ℓ。因此，集合 $\mathcal{Q}^i_{k+1} \subset \hat{} \mathcal{V}^i_k$ 包含了所有能够使目标 $\mathcal{Q}^i_k$可达的时间延迟所对应的状态。回想一下，在轨迹末尾，$\mathcal{Q}^i_0$ 等于$\mathcal{V}^i$。这对应于最大化调度器以正向方式为该跃迁选择一个时间延迟，使得从 $\mathcal{Q}^i_{k+1}$中的每个状态进入位置 ℓ_p后，都可以到达中间目标片段 $\mathcal{Q}^i_k$。类似地，对于相应的初始位置，$\mathcal{Q}^i_{k+1}$ 会限制初始集合。随后，调度器选择一个初始状态，使得 $\mathcal{Q}^i_k$ 可以通过 $\hat{} \mathcal{V}^i_k$到达。

图5b 展示了由非确定性守卫造成的线段重叠。从 $\mathcal{Q}^1_1$ 在 $\mathcal{V}^1_1$ 线段中的反向时间闭包限制了可能的时间延迟范围：只能从粉色片段，即从满足 $\nu_y \leq 11$ 的状态，到达（中间）目标线段。

速率非确定性。后向细化产生了受限线段，这些线段隐式地定义了调度器的划分。对于 $\mathcal{Q}_{k+1}^{i}$ 中的每个状态，最大化预知调度器至少可以选择一个斜率，从而到达中间目标集 $\mathcal{Q}_{k}^{i}$。相反，对于所有不在 $\mathcal{Q}_{k+1}^{i}$ 中的状态，都不存在这样的斜率。图 5a 表明，只有位于 $\hat{} \mathcal{V}_{2}^{0}$（因此也位于 $\mathcal{Q}_{3}^{0}$）中的初始状态才能到达目标。初始状态的选择限制了可能的速率。在此示例中，对于初始状态 $\nu_x = 1$，$\nu_y = 2$，只有最大的可能速率（$\dot{y} = 1/3$）才能到达中间目标线段 $\mathcal{Q}_{2}^{0}$，并使其能够到达 $\mathcal{V}^{0}$。

离散非确定性。每个离散选择都会在前向流管中产生一条不同的轨迹。反向分析从 $\mathcal{S}_{reach}^{goal}$ 开始，即只考虑通向目标状态的轨迹。因此，对后向细化所得到的所有轨迹 i 的 $\mathcal{P}^{i}$ 取并集，就表示所有能够到达目标的离散选择。由此，离散非确定性达到最大化。

总而言之，所有最大化调度器所诱导的赋值与后向细化计算出的赋值集合相符。

引理 2. 给定随机变量空间中允许调度器 $\mathfrak{s}$ 到达目标集 $\mathcal{S}^{goal}$ 的赋值集合 $\mathcal{V}^{\mathfrak{s}}$，以及定义等价类 $[\mathfrak{s}]_i$ 的等价关系 $\mathfrak{S}_{goal}$，$\bigcup_{\mathfrak{s}\in\mathfrak{S}_{goal}} \mathcal{V}^{\mathfrak{s}}$ 可按如下方式计算：

$$\bigcup_{\mathfrak{s}\in\mathfrak{S}_{goal}} \mathcal{V}^{\mathfrak{s}} = \bigcup_{i\in I_{\mathcal{S}}} \bigcup_{\mathfrak{s}\in[\mathfrak{s}]_i} \mathcal{V}^{\mathfrak{s}} = \bigcup_{i\in I_{\mathcal{S}}} \mathcal{V}^{[\mathfrak{s}]_i} = \bigcup_{i\in I_{\mathcal{S}}} \mathcal{P}^{i} = \mathcal{P}_{max}, \tag{2}$$

其中，$\mathcal{V}^{[\mathfrak{s}]_i}$ 是由调度器 $\mathfrak{s} \in [\mathfrak{s}]_i$ 所诱导的赋值的并集。

引理 2 的证明见附录 A。现在，我们可以通过对 $\mathcal{P}_{max}$ 积分来计算最大可达概率：

$$p_{max}^{\mathfrak{S}}(\mathcal{S}^{goal}, t_{max}) = \int_{\bigcup_{\mathfrak{s}\in\mathfrak{S}_{goal}} \mathcal{V}^{\mathfrak{s}}} G(\mathbf{s})\ d\mathbf{s} = \int_{\mathcal{P}_{max}} G(\mathbf{s}) d\mathbf{s}, \tag{3}$$

其中 $G(\mathbf{s}) = \prod_{r_n\in Var_R} \mathrm{Distr}(r_n)$ 是联合概率密度函数。

最大可达概率来自所有能够到达目标的调度器。因此，对所有样本值 $\mathcal{V}^{\mathfrak{s}}$ reachedby $\mathfrak{s} \in \mathfrak{S}_{goal}$ 取并集，即得到最大可达概率的积分域。根据构造，$\mathfrak{S}_{goal}$ 包含所有使可达概率达到最大的预知调度器：

(i) 所有能够到达目标的调度器 $\mathfrak{s}$ 都会到达一个状态集 $\mathcal{S}^{i}$，因此可由一个等价类 $[\mathfrak{s}]_i$表示，它们共同到达 $\mathcal{S}^{i}$。(ii) 调度器 $\mathfrak{s} \in \mathfrak{S} \setminus \mathfrak{S}_{goal}$ 无法到达目标状态 $\mathcal{S}^{goal}$。如果调度器 $\mathfrak{s} \in \mathfrak{S} \setminus \mathfrak{S}_{goal}$ 能够到达目标状态，那么 $\mathfrak{s}$ 可达的状态将属于某个流管线段，因此也属于一个状态集 $\mathcal{S}^{i}$。(iii) 前向分析与后向细化相结合，得出预知调度器 $\mathfrak{S}_{goal}$。前者确保所有导致目标状态的样本值均已知；后者则利用这些知识，根据不同状态集 $\mathcal{S}^{i}$ 的可达性对无限的调度器集进行划分。

积分误差。实际上，无法对无界多面体进行多维积分。因此，如第 3.3 节所述，我们将积分上限设为预先定义的界 $t_{int} \geq t_{max}$，而非无穷大。这会导致过近似误差：

$$e_{\infty} = 1 - \int_{[0,t_{int}]^{d_R}} G(\mathbf{s}) d\mathbf{s}.$$

当满足以下条件时，此误差是精确的：(i) 在所有轨迹上，抵达目标集时都没有随机时钟曾经到期；或者 (ii) 抵达目标时尚未到期的所有随机时钟的支集都是有限的。显然，增大 t_{int} 会降低 e_{∞}。积分通过蒙特卡洛 Vegas [18], 以统计方式完成，这会引入额外的统计误差 e_{stat}，其大小取决于积分样本的数量。

运行示例。最大化调度器形成两个等价类，其中$[\mathfrak{s}]_0$ 包含所有抵达 $\mathcal{S}^0$ 的调度器，而 $[\mathfrak{s}]_1$ 包含所有抵达$\mathcal{S}^1$的调度器。 $[\mathfrak{s}]_0$ 中的调度器从 ℓ_0中的 $\nu_y \geq 2$ 开始，然后选择一个速率为 y 且至少为$1 - \nu_y/3$ 的 ℓ_1。最后， ℓ_1 中的时间延迟必须确保 $\nu_y \geq 3 + 1/2 \cdot \nu_x$。 $[\mathfrak{s}]_1$ 中的调度器选择离开 ℓ_3 ，使得 $\nu_y \leq 11$。

假设随机延迟服从折叠正态分布 $\mathcal{N}(2, 1)$，通过对 $[1, \infty]$进行积分，计算在 $t_{max} = 10$ 之前到达 $\mathcal{S}^{goal}$ 的最大可达概率： $p^{\mathfrak{S}}_{max}(\mathcal{S}^{goal}, 10) = \int_{[1,\infty]} G(\mathbf{s}) d\mathbf{s}$ $= 1 - 0.2464 = 0.7536.$

计算复杂度。前向可达性分析 fwra 的复杂度随状态空间维度呈指数增长，并取决于自动机、初始状态集合，以及依赖于 t_{max} 的跳变次数 jmp。根据轨迹 $i \in I_{\mathcal{S}}$的最坏情况长度 $\mathcal{O}$(jmp)，将从其前驱线段 $\hat{} \mathcal{V}^i_{k-1}$ 计算一个细化片段 $\hat{} \mathcal{V}^i_k$ 的复杂度记为 bwra。由此可将所提分析的复杂度界定为 $\mathcal{O}$(fwra $+ |I_{\mathcal{S}}| \cdot$ jmp $\cdot$ bwra)，其中用于数值积分的集合数量为 $\mathcal{O}(|I_{\mathcal{S}}| \cdot$ jmp)。

4 可行性研究

图6展示了我们的电动汽车模型，该模型包含不同的充电模式，并以非确定性方式从不同区间中选择充电速率。电池充满后，充电便会停止。行驶距离由随机变量采样得到。参见 [8] ，了解包含所有动态过程的自动机。

位置 charging$_A$、charging$_B$、charging$_C$ 和 full 对应不同的充电速率，充电速率会根据电池的荷电状态而递减。充电时间由随机时钟 c 建模，该时钟服从折叠正态分布，即 $\mathcal{N}$(2, 2)。在 driving 位置，电池电量会减少，随机时钟 d 则对行驶所用时间建模。 d 到期后会转至 arrival 位置，而电池耗尽则会转至 empty 位置。驾驶时间服从折叠正态分布（$\mathcal{N}$(4, 1)）。

该模型具有可扩展性，因为它允许进行 0 次或多次绕行。在行驶位置，直到下一次绕行的随机延迟（r）与行驶结束时间

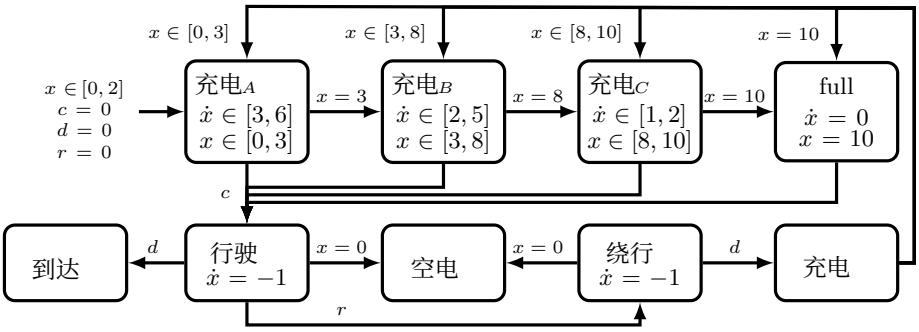


图 6：带绕行的汽车模型。随机时钟 c 在充电位置处激活（$\dot{c} = 1$），在行驶和绕行位置处激活的是 d，而在行驶位置处激活的是 r。除非另有说明，所有位置中的荷电状态 x 均限制为 $[0, 10]$。由于未显示的不变式，充电位置中不耗费时间。

（d）展开竞争。随机时钟 r 服从指数分布，其参数为 $\lambda = 2$。在绕行位置，d 仍处于活动状态，电池电量仍会下降。d 到期表示绕行结束，并执行向充电位置的跃迁，这标志着下一个充电周期的开始。根据电池当前的充电量，立即选择相匹配的充电位置。通过明确限制绕行次数，我们确保模型是无芝诺的。通过进行最坏情况分析，我们计算电池电量耗尽的最大概率。

实现与可复现性。我们采用工具 RealySt[5] 的原型实现，并使用库 HyPro [22] 计算流管段；具体而言，使用 HyPro 计算前向流管、可达树、反向时间闭包以及一步关系，这些正是后向细化中所用的内容。此外，我们使用 GNU Scientific Library [10] 进行多维积分。所有实验均在一台配备 Intel® Core™ i7（6×3.30GHz）处理器和 32 GB RAM 的计算机上运行。

结果。我们考虑三种不同的模型变体，即 ABC、AB 和 A，它们恰好包含模型名称所示的充电位置。如果移除充电位置$_B$ 和/或充电位置$_C$，则最后一个充电位置中的不变式将扩展至 $x \leq 10$，并相应调整该位置的输入和输出转移。为进一步降低模型复杂度，我们构造了一个奇异版本，其中连续变量的速率等于相应矩形区间的下界。通过初始集合保留连续非确定性，因此结果无法与 [20] 进行比较。

表 1列出了图6所示完整模型（标记为 ABC）的最大可达概率及相应的计算时间

[5] 工具和模型：https://zivgitlab.uni-muenster.de/ag-sks/tools/realyst/。

表 1：$\mathcal{S}^{goal} = \{$（空电 ν）$\mid \nu_x = 0\}$，$t_{max} =$（绕行数 $+1$）$\cdot 21$ 和绕行数 $\in \{0,1,2\}$ 的最大可达概率。t_{comp} 的计算时间适用于 RealySt（R），误差为 e_{stat} 和 ProHVer（P）。$K =$（$|\mathsf{Var}_R|$，$|\mathcal{R}|$，$|I_S|$）。

			0 次绕行			1 次绕行			2 次绕行
		var.	A	AB	ABC	A	AB	ABC	A
		K	(2, 8, 2)	(2, 12, 3)	(2, 16, 4)	(5, 38, 10)	(5, 88, 19)	(5, 167, 39)	(8, 128, 34)
rectangular	R	p_{max}	0.050 354	0.064 824	0.066 698	0.407 864	0.426 210	0.429 555	0.402 105
		e_{stat}	$8.763 \cdot 10^{-5}$	$1.192 \cdot 10^{-4}$	$1.286 \cdot 10^{-4}$	$1.715 \cdot 10^{-3}$	$1.677 \cdot 10^{-3}$	$6.131 \cdot 10^{-4}$	$3.327 \cdot 10^{-3}$
		t_{comp}	4.49 秒	10.75 秒	15.31 秒	484.50 秒	1166.49 秒	1800.63 s	35 570.30 s
	P	p_{max}	0.305 592	0.327 600	0.327 600	0.579 922	0.614 576	0.590 174	0.794 977
		t_{comp}	613.48 秒	1202.66 秒	1735.18 秒	7890.29 秒	15 247.25 秒	28 533.80 秒	1881.61 秒
singular	R	p_{max}	0.026 673	0.026 534	0.026 600	0.406 624	0.424 759	0.429 528	0.408 268
		e_{stat}	$3.405 \cdot 10^{-5}$	$9.476 \cdot 10^{-5}$	$1.123 \cdot 10^{-4}$	$7.120 \cdot 10^{-4}$	$6.703 \cdot 10^{-4}$	$1.676 \cdot 10^{-3}$	$3.719 \cdot 10^{-3}$
		t_{comp}	2.47 秒	11.66 秒	19.38 秒	245.21 秒	518.30 秒	1720.16 秒	12 187.80 秒
	P	p_{max}	0.305 592	0.327 600	0.327 600	0.442 959	0.467 916	0.523 958	0.758 034
		t_{comp}	348.09 秒	500.19 秒	546.56 秒	6984.12 秒	10 249.28 秒	14 273.85 秒	1461.85 秒

以及简化模型版本（AB 和 A）的结果。我们对 $t_{max} =$（绕行数 $+1$）$\cdot 21$ 进行缩放，绕行数为 $\in \{0,1,2\}$ 时使用 $1 \cdot 10^5$、$2 \cdot 10^6$、$1 \cdot 10^7$ 个积分样本，分别对应 0 、 1 、 2 次绕行。我们将原型 RealySt 计算出的不同参数设置下的结果与 ProHVer [12] 的结果进行验证；后者通过离散化对可达概率进行安全过近似。

表 1 对于每个所考虑的绕行次数，给出了模型的规模，表示为元组 $K = (|\mathsf{Var}_R|, |\mathcal{R}|, |\mathcal{S}^{goal}_{reach}|)$，其中 $|\mathsf{Var}_R|$ 是随机变量的数量，$|\mathcal{R}|$ 是可达树中的节点数，$|I_S| = |\mathcal{S}^{goal}_{reach}|$ 是通向目标集的轨迹数。通过正向和反向分析构造的多胞体维数为 $|\mathsf{Var}_R| + 3$，积分的维数等于 $|\mathsf{Var}_R|$。

RealySt 计算所有具有 0 次和 1 次绕行的模型变体的最大可达概率，最多需要 30 分钟。对于 2 次绕行，模型的复杂度显著增加。在奇异模型 A 中，计算最多需要 3.5 小时；而对于具有 2 次绕行的矩形变体，计算时间略低于 10 小时。具有 2 次绕行的变体 AB 和 ABC 的维数变得过高，以至于流管构造无法终止，或需要非常长的时间。RealySt 能够在略少于 83 小时内完成 AB 的奇异变体计算，并得到概率 0.431 565，$e_{stat} = 7.064 \cdot 10^{-3}$ 统计误差。

随着绕行次数增加，电池电量耗尽的概率显著上升，因为绕行会给电池的荷电状态带来不确定性。调度器可以利用这一点，最大限度地提高电池电量耗尽的概率。

最大化到达不期望目标的可达概率，可得到最坏情况概率。电池电量耗尽是不期望发生的，因此，计算出的到达目标的概率给出了事事不顺时的上界。结果表明，对充电过程进行更详细的建模，对计算出的概率影响相对较小。这是预期之中的，因为无论如何，充电对电池荷电状态的影响（速率介于 1 和

6）都大于行驶对电池荷电状态的影响（速率 -1）。矩形行为为调度器提供了许多影响模型演化的机会，这可能会提高可达概率。

在 1 次绕行的情况下，矩形模型与奇异模型的结果相近。这是因为奇异模型的速率等于矩形速率区间的下界。调度器旨在降低电池的荷电状态，因此在大多数情况下会选择可能的最低速率。

ProHVer 计算最大可达概率的安全过近似。不过，其精度很大程度上取决于所选离散化区间的数量。 ProHVer 的最新版本可根据给定参数自动生成区间，并进行细化。在 0 次绕行的情况下，ProHVer 计算得到的可达概率过近似程度明显高于 RealySt所得结果。 ProHVer 的计算时间是其 28 至 140 倍。在 1 次绕行的情况下， ProHVer 最多需要 8 小时，但得到的可达概率近似结果更好。在 2 次绕行的情况下， Pro-HVer 无法对其离散化进行细化，因此计算时间较短，但过近似程度明显。使用会增加离散化区间数量的替代参数运行 ProHVer，耗时超过 15 小时仍未完成。有关 ProHVer参数设置的详情，请参见附录 C.2；有关 RealySt计算时间的详情，请参见附录 C.3。

RealySt 表示 10^{-5} 至 10^{-3}之间的误差。根据引理 2，该误差完全源于积分。对于 $t_{int} = 100$ 以及分布 $\mathcal{N}(4, 1)$、$\mathcal{N}(2, 2)$ 和 $Exp(2)$，使用 IEEE 754 双精度计算得到的误差为 $e_{\infty} = 0$。因此， RealySt 计算出的概率加上所示误差 e_{stat} 后，与 ProHVer提供的过近似结果一致。

5 结论与未来工作

我们提出带随机时钟的矩形自动机，将离散行为和连续行为与随机延迟相结合，作为一种新的建模形式。在随机混合系统中，非确定性通常以概率方式得到解决。与此相反，本文提出了首个计算带随机时钟的矩形自动机的最大可达概率的方法，以预知方式完整解决所有类型的离散与连续非确定性。该计算需要结合前向流管构造和后向细化，以划分潜在的无限调度器集合。由此产生的误差完全源于多维积分。可行性研究的结果表明， RealySt 在最多包含五个随机变量的情况下表现良好。与 ProHVer相比，可达概率具有很高的准确性，且获取速度很快。未来的工作旨在通过其他状态集表示方法提高可扩展性，并计算预知的最小可达概率。我们将提供一种等价的 RAR 概念，通过语义隐式施加对随机延迟的限制，以简化建模；两者之间的转换将通过 RealySt保持可分析性。

参考文献

1. Abate, A., Prandini, M., Lygeros, J., Sastry, S.：受控离散时间随机混合系统的概率可达性与安全性。Automatica 44 027 2 11 2724 2734 2008 10 1016 2008 03 ( )，– ( )。https://doi.org/ 。/j.automatica. . . . Alur, R., Courcoubetis, C.A., Halbwachs, N., Henzinger, T.A., Ho, P., Nicollin, X., Olivero, A., Sifakis, J., Yovine, S.：混合系统的算法验证。理论计算机科学 138(1)，3–34（1995年）。https://doi.org/10.1016/0304-3975(94)00202-T 3. Ballarini, P., Bertrand, N., Horv´ath, A., Paolieri, M., Vicario, E.：使用随机状态类对随机定时自动机网络进行瞬态分析。载于：第 10 届国际系统定量评估会议。LNCS，第 8054 卷，第 355 371 2013 10 1007 978 3 642 40196 1_30 – 页。施普林格（）。https://doi.org/ 。/ - - - - 。Bertrand, N., Bouyer, P., Brihaye, T., Menet, Q., Baier, C., Großer, M., Jurdzinski, M.：随机定时自动机。计算机科学逻辑方法 10(4)，1–73 2014 5 2014 10 2168 10 4 6（）。https://doi.org/ 。/LMCS- (：)。Cauchi, N., Abate, A.：StocHy：随机过程的自动验证与综合。载于：TACAS'19 论文集。第 247–264 页。施普林格，查姆（2019年）。https://doi.org/10 3313349 6 1145 3302504 。/ 。 。D'Argenio, P.R., Gerhold, M., Hartmanns, A., Sedwards, S.：随机自动机的调度器类层次结构。载于：第 21 届软件科学与计算结构基础国际会议。LNCS，第 10803 卷，第 384–402 页。施普林格（2018 2 21 7 10 1007 978 3 319 89366）。https://doi.org/ 。/ - - - - 。D'Argenio, P.R., Katoen, J.P.：随机系统理论第一部分：随机自动机。信息与计算 203(1)，1–38（2005年）。https://doi.org/ 10 1016 2005 07 001 。/j.ic. . . . Delicaris, J., Schupp, S., Abraham, E., Remke, A.：带随机时钟的矩形自动机中的最大可达概率（2023年）。https://doi.org/10 14996 9 48550 2304 。/arXiv. . . Frehse, G.：PHAVer：继 HyTech 之后的混合系统算法验证。载于：第 8 届国际混合系统：计算与控制研讨会。LNCS，第 3414 卷，第 258–273 页。施普林格（2005年）。https://doi.org/10.1007/s10009-007-0062-x 10. Galassi, M., Davies, J., Theiler, J., Gough, B.：GNU 科学计算库参考手册——第三版。Network Theory Ltd.（2009年） 11. Hahn, E.M., Hartmanns, A., Hermanns, H.：随机定时自动机的可达性与奖励检查。EASST 电子通讯 70（2014年）。https://doi.org/10 2014 12 2168 10 4 6 。/LMCS- (：)。Hahn, E.M., Hartmanns, A., Hermanns, H., Katoen, J.P.：随机混合系统的组合式建模与分析框架。系统设计形式化方法 43(2)，191–232（2013年）。https://doi.org/10.1007/s10703-012-0167-z 13. Henzinger, T.A.：混合自动机理论。载于：数字与混合系统验证，NATO ASI 系列，第 170 卷，第 265–292 页。施普林格（2000年）。https://doi.org/10 5 13 14 1007 978 3 642 59615 。/ - - - - 。Henzinger, T.A., Kopke, P.W., Puri, A., Varaiya, P.：关于混合自动机，有哪些问题是可判定的？计算机与系统科学杂志 57(1)，94–124（1998年）。https://doi.org/10 1581 15 1006 1998 。/jcss. . . Koutsoukos, X.D., Riley, D.：随机混合系统验证的计算方法。IEEE 系统、人类与控制论汇刊 A 部分：系统与人类 38(2)，385–396（2008年）。https://doi.org/10.1109/TSMCA.2007.914777

16. Kwiatkowska, M.Z., Norman, G., Segala, R., Sproston, J.: 验证连续概率定时自动机的定量性质。载于：第11届并发理论国际会议。LNCS，第1877卷，第123–137页。施普林格（2000年）。https://doi.org/10 4 11 17 1007 3 540 44618 . / - - - . Lygeros, J., Prandini, M.: 随机混合系统：一种适用于复杂大规模应用的强大框架。《欧洲控制杂志》16(6)，583–594（2010 594 18 10 3166 16 583）。https://doi.org/ . /ejc. . - . Peter Lepage, G.: 一种用于自适应多维积分的新算法。《计算物理学杂志》27(2)，192–203（1978年）。https://doi.org/10 9 19 1016 0021 9991 78 90004 . / - ( ) - . Pilch, C., Hartmanns, A., Remke, A.: 针对具有随机触发的混合 Petri 网的经典与非预言式模型检验。载于：第23届ACM混合系统：计算与控制国际会议。第1–11页。ACM（2020年）。https://doi.org/10 3382198 20 1145 3365365 . / . . Pilch, C., Schupp, S., Remke, A.: 通过流管构造优化一类受限随机混合自动机的可达概率。载于：第18届系统定量评估国际会议。第435–456页（2021年）。https://doi.org/10 9 23 21 1007 978 3 030 85172 . / - - - - . Prandini, M., Hu, J.: 一种用于可达性计算的随机逼近法。载于：《随机混合系统：理论与安全关键应用》，LNCS，第 337, 107 139 2006 10 1007 11587392_4 22 ++ 卷，第–页。施普林格（）。https://doi.org/ . / . Schupp, S., Abraham, E., Ben Makhlouf, I., Kowalewski, S.: HyPro：一种用于混合系统可达性分析的状态集表示C语言库。载于：第9届NASA形式化方法研讨会。LNCS，第10227卷，第288–294页。施普林格（2017 8 20 23 10 1007 978 3 319 57288 2）。https://doi.org/ . / - - - - . Soudjani, S.E.Z., Gevaerts, C., Abate, A.: FAUST：不可数状态随机过程的形式化抽象。载于：第21届系统构造与分析工具及算法国际会议。LNCS，第9035卷，第272–286页。施普林格（2015 0 23 24 10 1007 978 3 662 46681）。https://doi.org/ . / - - - - . Sproston, J.: 概率混合自动机的可判定模型检验。载于：第6届实时与容错系统形式化技术国际研讨会。LNCS，第1926卷，第31–45页。施普林格（2000年）。https://doi.org/10 0 5 25 1007 3 540 45352 . / - - - . Sproston, J.: 具有有限互模拟的概率混合自动机验证与控制。J. Log. Algebraic Methods Program. 103，46–61（2019年）。https://doi.org/10 001 26 1016 2018 11 . /j.jlamp. . . . Zhang, L., She, Z., Ratschan, S., Hermanns, H., Hahn, E.M.: 概率混合系统的安全性验证。《欧洲控制杂志》18(6)，572–587（2012年）。https://doi.org/10 587 27 3166 18 572 . /EJC. . - . Ziegler, G.: 多胞体讲义，第152卷。施普林格科学与商业媒体（1995年）。https://doi.org/10.1007/978-1-4613-8431-1

附录 A 证明

引理 1. 假设 $\sigma_0 \xrightarrow{t_1,\ rate_1} \sigma_1 \xrightarrow{t_2,\ rate_2} \sigma_2$ 对于 $\sigma_0, \sigma_1, \sigma_2 \in \mathcal{S}$，其位置为$\ell$，$t_1, t_2 \in \mathbb{R}_{\geq 0}$，速率为$_1$，速率$_2 \in$流 $_C(\ell)$。那么存在速率为 $\in$的流 $_C(\ell)$，使得$\sigma_0 \xrightarrow{t_1+t_2,\ rate} \sigma_2$。

证明（引理 1 的正确性）。令 $\sigma_i = (\ell, \nu_i, \mu_i) \in \Sigma$ 对于 $i = 0$，1，2，并令 t_1，$t_2 \in \mathbb{R}_{\geq 0}$ 且速率为 $_1$，速率 $_2 \in$ 流 $_C(\ell)$，其中 $\sigma_0 \xrightarrow{t_1,\ rate_1} \sigma_1 \xrightarrow{t_2,\ rate_2} \sigma_2$。根据时间延迟的语义，我们知道

– 存在速率 $_1 \in$ 流 $_C(\ell)$，使得 $\nu_1 = \nu_0 + t_1 \cdot$ 速率 $_1$，$\mu_1 = \mu_0 + t_1 \cdot$ 流 $_R(\ell)$，并且 – 存在速率 $_2 \in$ 流 $_C(\ell)$，使得 $\nu_2 = \nu_1 + t_2 \cdot$ 速率 $_2$，$\mu_2 = \mu_1 + t_2 \cdot$ 流 $_R(\ell)$，且对于所有 $r \in$ 变量 $_R$，存在某个 $v \in$ supp(Distr(r))，使得 $\mu_{2r} \leq v$。

情形 $t_1 + t_2 = 0$ 显而易见。对于 $t_1 + t_2 > 0$，我们观察到

$$\begin{aligned}\nu_2 &= \nu_0 + t_1 \cdot rate_1 + t_2 \cdot rate_2 \\ &= \nu_0 + (t_1 + t_2) \cdot \frac{t_1 \cdot rate_1 + t_2 \cdot rate_2}{t_1 + t_2} \\ &= \nu_0 + (t_1 + t_2) \cdot \Big[\frac{t_1}{t_1 + t_2} \cdot rate_1 + \frac{t_2}{t_1 + t_2} \cdot rate_2\Big] .\end{aligned}$$

由于 t_1 和 t_2 均为非负数，因此因子 $\frac{t_1}{t_1+t_2}$ 和 $\frac{t_2}{t_1+t_2}$ 也均为非负数，且其和为 1；也就是说，来自凸集流 $_C(\ell)$ 的速率 rate $= \frac{t_1}{t_1+t_2} \cdot$ 速率 $_1 + \frac{t_2}{t_1+t_2} \cdot$ 速率 $_2$ 的线性组合，其中速率为 rate$_1$ 和 rate$_2$，也包含在流 $_C(\ell)$ 中。由此

$$\begin{aligned}\mu_2 &= \mu_1 + t_2 \cdot Flow_R(\ell) \\ &= \mu_0 + t_1 \cdot Flow_R(\ell) + t_2 \cdot Flow_R(\ell) \\ &= \mu_0 + (t_1 + t_2) \cdot Flow_R(\ell)\end{aligned}$$

我们已证明，时间步 $\sigma_0 \xrightarrow{t_1+t_2,\ rate} \sigma_2$ 的所有前提条件均已满足。

接下来，我们证明为最大化预知调度器推导出的样本域的正确性，即等式 $\bigcup_{\mathfrak{s} \in \mathfrak{S}_{goal}} \mathcal{V}^{\mathfrak{s}} = \mathcal{P}_{max}$ 成立：

引理 2. 给定随机变量空间上允许调度器 $\mathfrak{s}$ 到达目标集 $\mathcal{S}^{goal}$ 的赋值集合 $\mathcal{V}^{\mathfrak{s}}$，以及定义等价类 $[\mathfrak{s}]_i$ 的等价关系 $\mathfrak{S}_{goal}$，可以按如下方式计算 $\bigcup_{\mathfrak{s} \in \mathfrak{S}_{goal}} \mathcal{V}^{\mathfrak{s}}$：

$$\bigcup_{\mathfrak{s} \in \mathfrak{S}_{goal}} \mathcal{V}^{\mathfrak{s}} = \bigcup_{i \in I_S} \bigcup_{\mathfrak{s} \in [\mathfrak{s}]_i} \mathcal{V}^{\mathfrak{s}} = \bigcup_{i \in I_S} \mathcal{V}^{[\mathfrak{s}]_i} = \bigcup_{i \in I_S} \mathcal{P}^i = \mathcal{P}_{\max}, \tag{2}$$

其中，$\mathcal{V}^{[\mathfrak{s}]_i}$ 是由调度器 $\mathfrak{s} \in [\mathfrak{s}]_i$ 诱导的赋值并集。

回想一下，在样本域的提取过程中（参见第 3.3 节），我们会迭代遍历每条通向目标状态集 $\mathcal{S}^i$ 的轨迹 i，并收集关于样本的约束。在这里，我们区分每条线段中的随机延迟是 i）已到期，ii）将通过在可达树中迈出下一步而到期，还是 iii）尚未到期。因此，在每条轨迹上，每个随机延迟都对应一条线段 $\hat{} \mathcal{V}_k^i$，表示该随机延迟即将

到期时的赋值集。我们将索引 k 映射到相应的随机时钟 $r_n \in \text{Var}_R$，映射关系为 $map_i: I_i \to \text{Var}_R$，其中 I_i 收集了所有此类线段 $\hat{} \mathcal{V}_k^i$ 的索引 k。注意，$|I_i| \leq |\text{Var}_R|$，因为对于在 $\mathcal{S}^i = (l, \mathcal{V}^i)$ 中尚未到期的所有随机时钟，都不存在此类线段。

接下来，我们用 P_k^i、$i \in I_S$ 和 $k \in I_i$ 表示从 $\hat{} \mathcal{V}_k^i$ 推导出约束后 $\mathcal{P}^i$ 的状态。因此，对于每个随机延迟 r_n，多胞体 P_k^i（$map_i(k) = r_n$）包含样本 s_{r_n} 的上界。随后，我们可以将 $\mathcal{P}^i$ 的计算描述为所有 P_k^i 的交集：$\mathcal{P}^i = \left(\bigcap_{k \in I_i} P_k^i\right) \cap P^i$；其中，我们用不带索引 k 的 P^i 表示可直接从 $\mathcal{V}^i$ 本身推导出的约束。我们用 $\Pi_{\{r\}}(P)$ 表示多胞体 P 在维度 r 上的投影。利用这些记号，我们可以证明引理 2。

证明（引理 2 的正确性）。在存在连续非确定性的情况下，最大化调度器 $\mathfrak{S}_{goal}$ 的集合是不可数无限的。我们注意到，导致经由一条轨迹和可达树得到相同状态集 $\mathcal{S}^i \subset \mathcal{S}^{goal}$ 的调度器划分（我们将该轨迹的索引指定为 i）可以通过等价类 $[\mathfrak{s}]_i$ 集体考察。因此，在公式 2 中对所有 $[\mathfrak{s}]_i$ 以及每个等价类中的所有调度器取并集，便完整覆盖了最大化调度器 $\mathfrak{S}_{goal}$ 的集合。因此，公式 2 中的第一个等式由等价关系的性质得出，第二个等式则直接由 $\mathcal{V}^{[\mathfrak{s}]_i}$ 的定义得出。第四个等式由 $\mathcal{P}_{max}$ 的定义得出；$\mathcal{P}_{max}$ 是所提出的反向分析方法的结果，该方法在第 3.2 节和第 3.3 节中概述。

对于第三个等式，我们围绕轨迹索引 i 进行论证。每个集合 $[\mathfrak{s}]_i$ 都对应一条由 $[\mathfrak{s}]_i$ 中的调度器导出的轨迹 i。对于每条轨迹 i，$\mathcal{P}^i$都对所有样本定义约束，使得能够做出决策并导向 $\mathcal{S}^i$ 的调度器存在。因此，第三个等式源于对所有 $i \in I_{\mathcal{S}}$ 都成立的等式 $\mathcal{V}^{[\mathfrak{s}]_i} = \mathcal{P}^i$，我们将在下文证明：

$$s \in \mathcal{P}^i = \left(\bigcap_{k \in I_i} P_k^i\right) \cap P^i \tag{1}$$

$$\Leftrightarrow s \in P^i \wedge \forall k \in I_i : s \in P_k^i \tag{2}$$

$$\Leftrightarrow \forall r_n \in Var_R : \left(s_{r_n} \in \Pi_{\{r_n\}}\left(P^i\right) \wedge \forall k \in I_i : s_{r_n} \in \Pi_{\{r_n\}}\left(P_k^i\right)\right) \tag{3}$$

$$\Leftrightarrow \forall r_n \in Var_R : \forall k \in I_i : \left(s_{r_n} \in \Pi_{\{r_n\}}\left(P^i\right) \wedge s_{r_n} \in \Pi_{\{r_n\}}\left(P_k^i\right)\right) \tag{4}$$

$$\begin{aligned} \Leftrightarrow \forall r_n \in \text{变量}_R : \quad & \forall k \in I_i : \\ & \left(s_{r_n} \in \Pi_{\{r_n\}}\left(P^i\right) \wedge s_{r_n} \in \Pi_{\{r_n\}}\left(P_k^i\right) \wedge map_i(k) = r_n\right) \\ & \vee \left(s_{r_n} \in \Pi_{\{r_n\}}\left(P^i\right) \wedge s_{r_n} \in \Pi_{\{r_n\}}\left(P_k^i\right) \wedge map_i(k) \neq r_n\right) \end{aligned} \tag{5}$$

$$\begin{aligned} \Leftrightarrow \forall r_n \in \text{变量}_R : & \\ & \left(\forall k \in \{k \in I_i \mid map_i(k) = r_n\} : \left(s_{r_n} \in \Pi_{\{r_n\}}\left(P^i\right) \wedge s_{r_n} \in \Pi_{\{r_n\}}\left(P_k^i\right)\right)\right) \\ & \wedge \left(\forall k \in \{k \in I_i \mid map_i(k) \neq r_n\} : \left(s_{r_n} \in \Pi_{\{r_n\}}\left(P^i\right) \wedge s_{r_n} \in \Pi_{\{r_n\}}\left(P_k^i\right)\right)\right) \end{aligned} \tag{6}$$

$$\Leftrightarrow \forall r_n \in \text{变量}_R : \tag{7}$$

$$\Big(\exists k \in I_i : map_i(k) = r_n \Rightarrow \big(s_{r_n} \in \Pi_{\{r_n\}}\left(P^i\right) \wedge s_{r_n} \in \Pi_{\{r_n\}}\left(P^i_k\right)\big)\Big)$$
$$\wedge \Big(\forall k \in \{k \in I_i \mid map_i(k) \neq r_n\} : \big(s_{r_n} \in \Pi_{\{r_n\}}\left(P^i\right) \wedge s_{r_n} \in \Pi_{\{r_n\}}\left(P^i_k\right)\big)\Big)$$
$$\Leftrightarrow \forall r_n \in Var_R : \Big(s_{r_n} \in \Pi_{\{r_n\}}\left(P^i\right) \wedge \tag{8}$$
$$\big(\exists k \in I_i : (map_i(k) = r_n) \Rightarrow s_{r_n} \in \Pi_{\{r_n\}}\left(P^i_k\right)\big)\Big)$$
$$\Leftrightarrow \forall r_n \in Var_R : \Big(s_{r_n} \in \Pi_{\{r_n\}}\left(\Lambda_{\{r_n\}}(\mathcal{V}^i)\right) \tag{9}$$
$$\wedge \big(\exists k \in I_i : (map_i(k) = r_n) \Rightarrow s_{r_n} \in \Pi_{\{r_n\}}\left(\hat{\mathcal{V}}^i_k\right)\big)\Big)$$
$$\Leftrightarrow \exists \mathfrak{s} \in [\mathfrak{s}]_i : \exists \pi \text{ initial } : \pi = \sigma_0 \to \cdots \to \sigma_n \wedge \sigma_n \in \mathcal{S}^i \subset \mathcal{S}^{goal} \wedge \tag{10}$$
$$\forall r_n \in Var_R : \exists b \in \{0, \ldots, n\} :$$
$$\Big(b = n \vee \big(\exists k \in I_i : (map_i(k) = r_n) \Rightarrow b = n - 2 \cdot k\big)\Big)$$
$$\wedge \forall \sigma_a = (\ell_a, \nu_a, \mu_a, s_a) \in \{\sigma_0, \ldots, \sigma_b\} : s_a = s_{r_n}$$
$$\Leftrightarrow s \in \mathcal{V}^{[\mathfrak{s}]_i} \tag{11}$$

前两个等价关系（$1 \Leftrightarrow 2 \Leftrightarrow 3$）由第 3.3 节给出的定义得出。接下来的等价关系（$3 \Leftrightarrow 4 \Leftrightarrow 5 \Leftrightarrow 6$）区分了映射到某个变量 r_n 的索引 k 与不映射到该变量的索引。具体而言，由于对于每个 r_n，至多存在一个 k 满足 $map_i(k) = r_n$，因此等价关系 $6 \Leftrightarrow 7$ 成立。

虽然蕴含关系 $7 \Rightarrow 8 \Rightarrow 9$ 显而易见，但蕴含关系 $9 \Rightarrow 8 \Rightarrow 7$ 成立，因为额外的线段 P^i_k 和 P^i 不会因提升以及随机时钟作为秒表时钟而对赋值施加更严格的限制；也就是说，随着自动机中的进程推进，赋值要么递增，要么保持不变。注意，对于所有满足 $map_i(k) = r_n$ 的变量 r_n，所有 $P^i_{k'}$ 都只包含下界，其中 (i) 对于满足 $k' < k$ 的 $P^i_{k'}$，其限制比 P^i_k 中的下界宽松，因为 r_n 尚未到期；(ii) 对于满足 $k' > k$ 的 $P^i_{k'}$（以及类似的 P^i），由于到期，其下界等于从 P^i_k 得到的下界。对于不会在轨迹 i 上到期的 r_n，(i) 对每个 P^i_k 都成立，因为 $\forall k \in I_i : map_i(k) \neq r_n$。

由于线段 $\hat{} \mathcal{V}^i_k$ 和 $\mathcal{V}^i$ 属于前向流管（以及后向细化），因此可以找到一个遵循该轨迹且赋值为 $s \in \mathcal{P}^i$ ($9 \Leftrightarrow 10$) 的调度器。因此，存在一条初始路径，其中直到第一次（也是唯一一次）到期为止的样本值与 s 相符。这相当于掌握了随机变量未来到期时间的信息。随后，基于这些信息作出调度决策，即采用预知式调度。最后一个等价关系（$10 \Leftrightarrow 11$）由 $\mathcal{V}^{[\mathfrak{s}]_i}$ 的定义得出。

由于可达树是有限的，所提方法必然终止。后向细化在所有线段 $\mathcal{V}^i$ 上进行，每条线段都对应可达树中的一条线段。随后，计算沿着轨迹朝可达树的根节点进行，直到到达根节点。将

每条轨迹上的相关线段合并后，其数量也因轨迹数量有限而有界。

附录 B 运行示例计算

细化片段 $\hat{}\ \mathcal{V}_2^0$ 和下一个中间目标片段 $\mathcal{Q}_3^0$ （参见图5a）可由 $\mathcal{Q}_2^0$ 计算得出，从而得到以下线段：

$$\begin{aligned}
\mathcal{Q}_2^0 &= \{\nu \in \mathbb{R}^3 \mid \nu_x = 4 \wedge \nu_y \leq \nu_x \wedge \nu_y \geq \nu_x - 1 \wedge \nu_r = 3\} \\
\hat{\mathcal{V}}_2^0 &= T_{\ell_0}^-(\mathcal{Q}_2^0) \cap \mathcal{V}_2^0 = \{\nu \in \mathbb{R}^3 \mid \nu_x \leq 4 \wedge \nu_y \geq 5/3 + 1/3 \cdot \nu_x \\
&\quad \wedge \nu_y \leq 8/3 + 1/3 \cdot \nu_x \wedge \nu_r = \nu_x - 1\} \\
\mathcal{Q}_3^0 &= \hat{\mathcal{V}}_2^0 \cap \mathit{Init}(\ell_0) = \{\nu \in \mathbb{R}^3 \mid \nu_x \leq 1 \wedge \nu_y \in [2,3] \wedge \nu_r = 0\}
\end{aligned}$$

细化片段 $\hat{}\ \mathcal{V}_1^1$ 和下一个中间目标片段 $\mathcal{Q}_2^1$ （参见图5b）由 $\mathcal{Q}_1^1$ 计算得出，从而得到以下线段：

$$\begin{aligned}
\mathcal{Q}_1^1 &= \{\nu \in \mathbb{R}^3 \mid \nu_x \in [2,4] \wedge \nu_y \in [10,11] \wedge \nu_r = \nu_x - 1\} \\
\hat{\mathcal{V}}_1^1 &= T_{\ell_3}^-(\mathcal{Q}_1^1) \cap \mathcal{V}_1^1 = \{\nu \in \mathbb{R}^3 \mid \nu_x \in [2,4] \wedge \nu_y \in [1,11] \wedge \nu_r = \nu_x - 1\} \\
\mathcal{Q}_2^1 &= D_e^-(\hat{\mathcal{V}}_2^1) \cap \mathcal{V}_2^1 = \{\nu \in \mathbb{R}^3 \mid \nu_x \in [2,4] \wedge \nu_y \geq 1 \\
&\quad \wedge \nu_y \leq 8/3 + 1/3 \cdot \nu_x \wedge \nu_r = \nu_x - 1\}
\end{aligned}$$

附录 C 可行性研究

附录 C.1 汽车模型

为完整起见，我们提供可行性研究中使用的完整自动机，其中包含图 7 中的所有变量和流。

附录 C.2 使用 ProHVer 进行验证

据我们所知， ProHVer 是唯一有望计算这类模型中预言式最大可达概率的安全过近似的工具。然而，其精度高度依赖于所选的离散化区间数量，以及连续域中的非确定性选择数量。这两者都严重限制了可扩展性， ProHVer计算出的结果便证明了这一点。 ProHVer 中的区间细化需要整数参数（X、Y、Z），这些参数限定离散化范围并决定离散化区间的数量（参见表 2）。对于 0 次绕行，我们选择了 (60,8,2)；对于 1 次绕行，选择了 (10,2,1)。模型复杂度随绕行次数增加。对于 2 次绕行，我们只能得到参数 (5,2,1) 下的结果。对于 0 次绕行，这产生了 66 个区间，并使得到的可达概率相较于 RealySt 大幅过近似。
ProHVer 的计算时间是

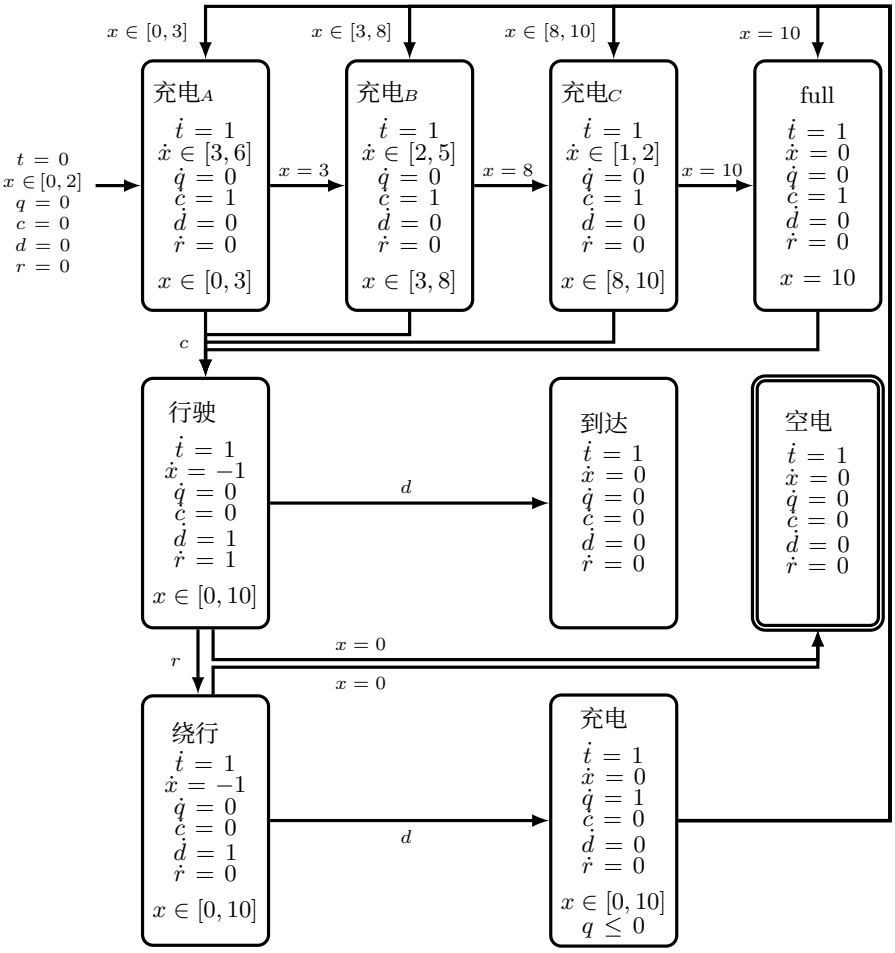


图 7：可行性研究中使用的 RAR 完整模型。

其 28 至 140 倍。对于 1 次绕行， ProHVer 计算了 10 个区间，耗时最长达 8 小时，并得到了更准确的可达概率近似。对于 2 次绕行， ProHVer 无法细化最初的 5 个区间，因此计算时间较短，但过近似幅度较大。对于参数 (6, 2, 1)， ProHVer 在 15 小时内未能终止。

附录 C.3 计算时间分布

如图 8 所示，对于不同的模型版本，计算时间分为流管构造、细化操作、样本域提取和积分。

绕行数为 0 时，计算时间主要消耗在积分上；而绕行数更多时，流管构造则占主导。一方面，更多

表 2： $\mathcal{S}^{goal} = \{$（空电 ν）$\mid \nu_x = 0\}$、 $t_{max} =$（绕行数 $+1$）$\cdot 21$ 以及绕行数 $\in \{0, 1, 2\}$ 的最大可达概率。提供了 REALYST （R）在误差为 e_{stat} 时的计算时间 t_{comp}，以及 P（I 个离散化区间） PROHVER 的计算时间。 $K =$（$|\mathrm{Var}_R|$、 $|\mathcal{R}|$、 $|I_S|$）。

			0 次绕行			1 次绕行			2 次绕行
		var.	A	AB	ABC	A	AB	ABC	A
		K	(2, 8, 2)	(2, 12, 3)	(2, 16, 4)	(5, 38, 10)	(5, 88, 19)	(5, 167, 39)	(8, 128, 34)
rectangular	R	p_{max}	0.050 354	0.064 824	0.066 698	0.407 864	0.426 210	0.429 555	0.402 105
		e_{stat}	$8.763 \cdot 10^{-5}$	$1.192 \cdot 10^{-4}$	$1.286 \cdot 10^{-4}$	$1.715 \cdot 10^{-3}$	$1.677 \cdot 10^{-3}$	$6.131 \cdot 10^{-4}$	$3.327 \cdot 10^{-3}$
		t_{comp}	4.49 秒	10.75 秒	15.31 秒	484.50 秒	1166.49 秒	1800.63 秒	35 570.30 秒
	P	p_{max}	0.305 592	0.327 600	0.327 600	0.579 922	0.614 576	0.590 174	0.794 977
		$\#I$	66	66	66	10	10	10	5
		t_{comp}	613.48 秒	1202.66 秒	1735.18 秒	7890.29 秒	15 247.25 秒	28 533.80 秒	1881.61 秒
singular	R	p_{max}	0.026 673	0.026 534	0.026 600	0.406 624	0.424 759	0.429 528	0.408 268
		e_{stat}	$3.405 \cdot 10^{-5}$	$9.476 \cdot 10^{-5}$	$1.123 \cdot 10^{-4}$	$7.120 \cdot 10^{-4}$	$6.703 \cdot 10^{-4}$	$1.676 \cdot 10^{-3}$	$3.719 \cdot 10^{-3}$
		t_{comp}	2.47 秒	11.66 秒	19.38 秒	245.21 秒	518.30 秒	1720.16 秒	12 187.80 秒
	P	p_{max}	0.305 592	0.327 600	0.327 600	0.442 959	0.467 916	0.523 958	0.758 034
		$\#I$	66	66	66	10	10	10	5
		t_{comp}	348.09 秒	500.19 秒	546.56 秒	6984.12 秒	10 249.28 秒	14 273.85 秒	1461.85 秒

更高维度下需要更多积分样本，因为所得的多维多胞体覆盖了整体积分域中更小的一部分。此外，在更高维度下，评估单个样本（包含性测试）所需的时间也更长。

另一方面，维数增加后，对凸多胞体进行的所有几何运算都需要更多计算时间，这会显著影响流管构造。同一模型版本中，矩形变体与奇异模型变体之间的计算时间差异，源于矩形混合自动机的流管构造更为复杂。这一点也体现在：当绕行为 2 时，模型版本 AB 的矩形变体在流管构造过程中因内存不足而终止。

请注意，对于固定数量的绕行和样本，奇异模型版本的积分时间不一定更短，因为基于重要性抽样的方法会带来统计波动。模型版本 ABC、AB 和 A 的规模略有差异，但在给定绕行数量时维数相同。因此，维数对计算时间的影响大于模型规模的影响。

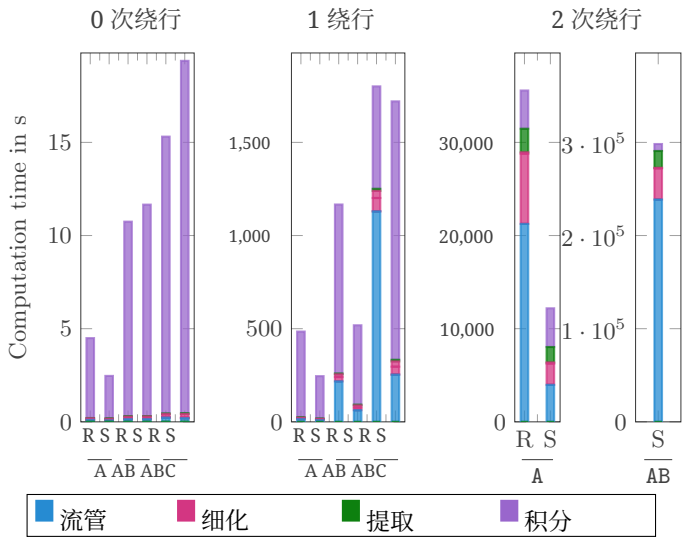


图 8：不同模型和绕行次数下的计算时间，采用 REA-LYST。我们区分流管（第 3.1 节）、细化（第 3.2 节）、提取（第 3.3 节）和积分（第 3.4 节）。