无代码人工智能：根据文档自动生成功能块图，并采用相关启发式方法训练上下文感知机器学习算法

Oluwatosin Ogundare、Gustavo Quiros Araya、Yassine Qamsane 西门子技术中心，美国新泽西州普林斯顿
gustavo.quiros@siemens.com

摘要—工业流程工程和 PLC 程序开发传统上更倾向于使用功能块图（FBD）编程，而非面向对象和函数式编程范式等经典命令式编程风格。如今，统计学习**理论或所谓机器学习的主流成功，与被归类为“无代码”或“低代码”的理念日益兴起并得到尝试**，正在重新定义我们构建程序、供数字机器执行的方式。**“无代码”的一个主要关注点是直接从一组需求文档中推导出可执行程序**，也可以从定义消费者或客户期望的任何其他文档中推导。本研究提出一种方法：利用受限选择算法，从相关推荐系统的首选项中进行选择，根据一组需求文档生成可由目标系统执行的功能块图（FBD）程序，作为中间产物或最终产物。所呈现的结果表明，这类无代码生成模型是工业流程设计的一种可行选择。

关键词——无代码；机器学习训练启发式方法；统计学习；面向工业工程的人工智能

I. 引言

在工业控制领域，功能块图（FBD）广泛用于设计在可编程逻辑控制器（PLC）上实现的控制系统软件。FBD 编程是 IEC 61131-3 标准 [1] 及其扩展标准 IEC 61499 [2]所定义的标准 PLC 编程语言之一。目前，从需求文档生成代码的方法和启发式方法正迅速增多。其中许多解决方案利用 FBD 的简洁性，支持从文档生成代码在工业应用中切实可行这一观点。即使最终生成的是带框架的面向对象代码或函数式代码，有时也会将 FBD 作为中间产物或辅助产物生成。

当前工业控制实践中的代码生成通常基于启发式方法或形式化方法。启发式方法根据从业者的创造力，将文档中的需求直观地转换为控制代码 [3][4]。对于小型系统，这种将需求直接转化为控制代码的方式可能会产生实用的解决方案；然而，随着系统规模扩大，这种方法会变得不够充分且容易出错。为协助控制从业者完成任务，可以利用形式化方法自动生成控制代码。

一种基于线性时序逻辑、用于控制代码生成的规约语言在业界得到了广泛采用 [5]。Event-B形式化方法也被广泛用于设计PLC 控制代码，并要求在建模和细化过程中，通过形式化证明确保所设计的代码满足完备性、一致性、精确性和正确性 [6]。其他方法则将多种形式化语言与PLC 标准语言相结合，例如自动机与 GRAFCET/SFC[7]，Petri 网与梯形图 [8][9]，以及 Max Plus 代数与 Petri 网 [10][11]。

形式化方法优于传统启发式方法，因为它们采用设计精良的算法，能够以更少的精力和时间自动生成控制代码。然而，从实际角度来看，形式化方法要求控制从业者深入掌握高级数学概念，以便对文档中的需求、待控制的系统/过程，或两者进行建模。工业领域的大多数控制从业者并不具备运用形式化方法所需的知识。因此，需要采用额外的方法，将需求转换为形式化方法算法用于生成控制代码的数学规格。近年来，机器学习（ML）方法已被用于简化建模接口并降低设计复杂度。例如，有研究表明，一种基于深度学习、用于将非正式需求转换为正式信号时序逻辑（STL）的工具和技术，其表现可与形式化方法相媲美 [12]。此外，主动学习已成功用于为未知被控对象模型综合控制器 [13]。文献 [14] 中引入了主动自动机学习，以自动学习形式自动机模型。它已应用于许多领域，包括安全分析、测试、验证和综合。将形式化算法方法与学习算法相结合，可以用更少的从业者精力推断出表达能力更强的模型。本文探讨了如何在工业和以过程为中心的领域中，遵循无代码工程范式，从文档生成可执行的 FBD 程序：首先根据参考文档生成受约束的解空间，然后选择使条件概率最大化的功能块，这些条件概率由同一效用函数计算得出，而该效用函数也可独立用作推荐系统。

II. 使用 FBD 编程进行上下文感知工业过程设计的公理化定义

公理 1. $\forall g \in G$ 存在一个相关联的效用函数 U，用于描述 g 的工业功能；另有一个函数 π，它分配一系列集合 $\{S_j\}$，使得 S_j 由 G 中的设计元素构成，但允许重复，并满足以下条件

$$\exists g_1 \in G : (U_{g_1}(g_1) \wedge \forall t_1 \in T_j : U_{g_1}(t_1) \rightarrow g_1 = t_1)$$

从理论上讲，$j \rightarrow \infty$，因为符号可以任意重复；但 FBD 程序的结构存在实际的计算限制。通俗地说，π 可以看作是这样一个函数：它根据 G 分配创建 FBD 程序所需的符号，有时同一个符号会被使用多次。
FBD 程序所包含的功能块总数，或更一般地说，不考虑唯一性时所包含的符号总数，由 S_j描述。从工艺（设计）工程师的角度来看，在获得 S_j后，他们会确定 S_j中符号或功能块的排列方式。S_j的直观类比是将其视为一组需要拼装的拼图块。按照这一类比，G将是 S_j中的独特拼图块。在工厂自动化领域，π 几乎总是由需求文档定义的。例如，布尔门/块的数量可能与工艺与仪表流程图（P&ID）中的阀门数量相关。

FBD设计函数，非正式地称为拼图排列函数，或更正式地称为设计工程师对一组FBD符号/模块所执行的操作，由函数 ϕ 建模，并在下文中定义。

公理2. ϕ_i: $S_j \rightarrow \boldsymbol{f}_i$ 使得$\boldsymbol{f}_i$ 是S_j的所有 i 排列的集合。
$\boldsymbol{F}$ 是 S_j的所有已知排列的集合。

$$\boldsymbol{f}_i = \{ f_{i,k} \}$$
Where $k = 1, \dots, |S_j|!$
$i = \{1, .., |S_j|\}$

T_j 服从正态分布，即 $T_j \sim N(\mu, \sigma^2)$，这是一项实际考量，因为 FBD 工具箱中的设计元素并非具有相同的使用概率。某些功能块的使用频率高于其他功能块，因此更有可能出现在 FBD 程序实例中。
ψ_i 定义在$\boldsymbol{f}_i$ 上，是 T 的所有i 排列集合的概率分布。

$$\psi_i : f_{i,k} \rightarrow (0, 1)$$

$f_{i,k}$ 可非正式地称为 FBD程序实例或一组符号。例如，最简单的 FBD 程序位于 $\boldsymbol{f}_1$ （包含单个符号的所有 FBD 程序的集合）中。同样，下一个复杂程度允许包含2个符号/功能块的 FBD 程序，其中$\boldsymbol{f}_2$。沿着这一思路，$f_{2,1}$ 是 $\boldsymbol{f}_2$ 中的一个有效 2符号 FBD 程序，而 $f_{2,2}$则表示 $\boldsymbol{f}_2$ 中另一个有效的2符号FBD 程序或一组符号/功能块。ψ_i 受概率公理约束，并且 $\boldsymbol{f}_i's$ 是穷尽的，即如果最后一个选择位于 $\boldsymbol{f}_{i-1}$，则下一个选择必定是$\boldsymbol{f}_i$，再下一个位于 $\boldsymbol{f}_{i+1}$。简而言之，鉴于 $\boldsymbol{f}_{i-1}$ 已实现

$$\Pr(\boldsymbol{f}_i) = \begin{cases} 1, & If\ the\ engineer\ makes\ another\ selection \\ 0 & otherwise \end{cases}$$

然而，更具体地说，我们只关注在某个特定的 $i-1$ 符号已实现的情况下，所有有效的 i-符号 FBD 程序的概率分布。我们将条件概率估计如下

$$\Pr(\boldsymbol{f_i} = f_{i,k} \mid \boldsymbol{f_{i-1}} = f_{i-1,b}) = \frac{\boldsymbol{Pr}(\boldsymbol{f_{i-1}} = f_{i,b} | \boldsymbol{f_i} = f_{i,k}) \psi_i(f_{i,k})}{\psi_{i-1}(f_{i-1,b})}$$

因此，建议的下一步是使条件概率最大化的一步。

$$\max_k \{ \Pr(\boldsymbol{f_i} = f_{i,k} \mid \boldsymbol{f_{i-1}} = f_{i-1,b}) \}$$

Where $k, b = 1, \dots, i!$
$i > 1$

初始推荐遵循 T_j 的分布，随后使用条件概率分布，因此 $i > 1$

直观而言，$\boldsymbol{f_i} - \{f_{i,k}\}$ 表示所有其他配置的集合，其中从 T_j 中选择 i 个设计元素，并采用预设的 $i-1$ 配置。这些其他配置遵循条件概率分布，可用于多选项推荐系统，并根据条件概率对备选项进行排序。

由于生成一组功能块符号的所有排列，其数量等于 FBD 编程符号总数的阶乘，而且通常被认为在计算上是 NP难的，也就是说，计算复杂度会随着集合大小呈指数增长；在此情形下，集合大小即 FBD 程序符号集的长度，尤其是因为连接器与功能块本身之间没有区别。从图建模的角度来看，我们的方法抽象掉了

边与顶点之间的区分。构建 FBD 程序所用的所有元素统称为功能块，或更广义地称为符号。我们设计了一种启发式方法，可将置换集中的大块部分赋予零概率，从而将上下文学习的概率分布从

在所有可能性集合上均匀分布，转变为一种几何随机的分布。

III. FIONA 启发式函数

正向排列是双射，但在工业 FBD 程序设计中，并非所有符号/功能块的换位在结构上都有效。

随后，我们介绍函数组织归一化排列（FIONA）置换函数 on 作为一种用于生成定义上下文的启发式函数数据集。

考虑包含两个符号的集合，可以通过单个换位函数 σ 生成所有排列。然而，对于包含 3 个符号的集合，则需要 3 个换位函数 σ_1、 σ_2、 σ_3 来生成该集合的所有排列。我们将 σ_1 记作固定符号 1、交换符号 2 和 3 的换位。同样，σ_2 固定符号 2 并交换其余符号，而 σ_3 固定符号 3 并交换其余符号。严格来说，FIONA 启发式函数 σ_{fiona} 是 n 个符号的置换群的一个子群，并配备一个像和一个核。

在每次迭代 $i \in \{1, 2, 3, \dots, k\}$中，我们定义一个排除集合 $E = \{e_1, e_2, e_3, \dots, e_k\}$

We select the set $S_i = \{ s_{i,j} \in {}^{n}p_2 \mid s_{i,j} \cap e_i = \emptyset \}$

我们假设 S_i 的累积分布函数 (CDF) 在 (0, 1) 上服从均匀分布。我们根据各种 FBD 程序中功能块的历史分布推导出概率密度函数。直观而言，某些功能块更有可能出现在更多 FBD 程序中，因此对于每个 $s_{i,j}$，都有一个与之关联的条件概率 Pr ($s_{i,j}$)。因此，符号与其概率配对如下 ($s_{i,j}$, Pr ($s_{i,j}$))

IV. FIONA 上下文训练数据的长序列构建

我们引入 γ_i ，使其在第 i 次迭代时随机选择一个累积分布函数（CDF）值，精度为 S_i，使得

$$\gamma_i = \int_0^1 f(k)$$

$f(k)$ 是符号 k-index (value) 的似然密度函数

$j, k = \{1, \dots n \ (total\ number\ of\ symbols)\}$

这将得到以 j 为未知量的含参数的代数，且解将依据第一性原理得出。

序列生成函数，τ_{fiona}，使得在每次迭代中，i

$\tau_{fiona}: (i, j) \rightarrow \{s_{i-1,k} \cup s_{i,j}\}$

V. FIONA 训练启发式的结果

通俗地说，每当 π 在置换集 s_i中进行一次选择时，我们就会创建一个子问题，以确定在 s_{i+1}中有哪些可能性

形式上，我们学习 $\pi_j(s_i)$ 在 s_{i+1}中的图像。π_j 是能够生成唯一 FBD 程序（FBD_j）的具体选择策略。我们还学习了 s_{i+1} 中实际发生的情况，即 $\pi_j(s_{i+1})$；该方法论为 $\pi_j(s_{i+1})$赋予更高的权重，从而将这类学习与基于先前选择 $\pi_j(s_i)$在 s_{i+1} 中的其他可能性区分开来

为简化学习问题，我们为所选机器学习模型的不同实例分配任务；在本例中，循环神经网络（RNN）的任务是预测 $\pi_j(s_i)$，即选择策略 π_j 在 s_i中的操作。通过长短期记忆单元累积在每个 RNN 中的先前选择和图像数量，以及计算复杂度，构成了一个实际的设计问题。

下图展示了在大量 FBD 程序上进行训练时的误差曲面，这些程序按设计步骤划分。也就是说，我们先选取 n 个 FBD 程序中最前面的 2 个功能块，再从同一批 n 个 FBD 程序中选取最前面的 3 个功能块，然后选取最前面的 4 个功能块，依此类推。这样做的主要目的是观察机器学习模型在使用更通用的上下文数据训练后，在训练集上将误差降至最低的表现如何。

我们首先训练一个包含 50 个长短期记忆单元的循环神经网络，使其学习从 1 个符号到 2 个符号的转换关系；随后，我们使用相同的配置，训练另一个循环神经网络实例，使其学习从 2 个符号到 3 个符号的转换关系，以此类推。最后，我们使用相同的数据集，对配置相同但已在专注于相同转换的上下文数据上预训练的循环神经网络进行同样的训练，并在图 1–6 中可视化误差曲面。

1. 1 个符号到 2 个符号的转换训练

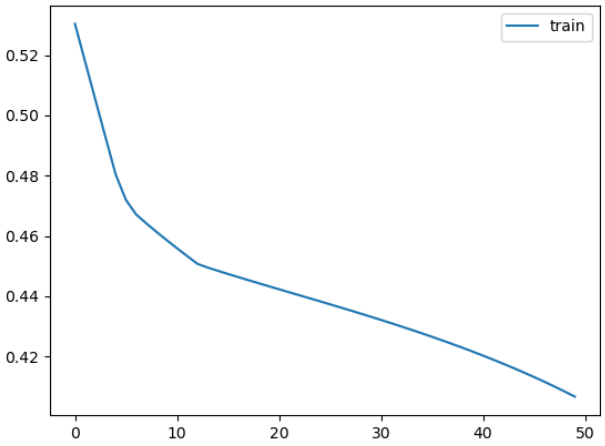

图 1. 展示 1-2 个符号转换在训练 50 轮次后的误差曲面的图。

2. 2 个符号到 3 个符号的转换训练

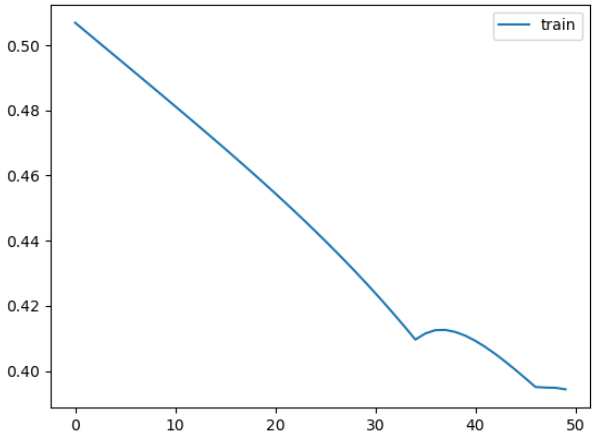

图 2. 展示 2-3 个符号转换在训练 50 轮次后的误差曲面的图。

3. 3 个符号到 4 个符号的转换训练

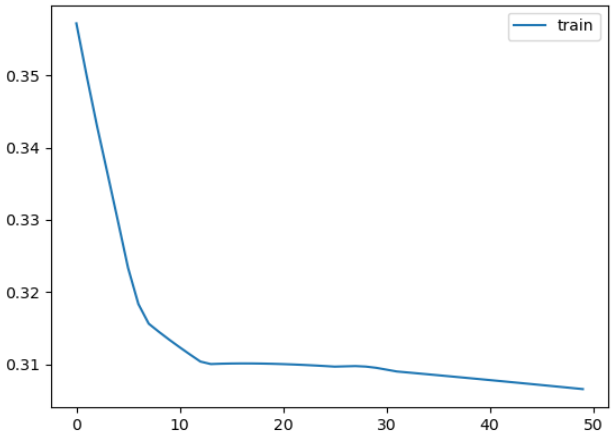

图 3. 展示 3-4 个符号转换在训练 50 轮次后的误差曲面的图。

在下方的图 4–6 中，我们使用在上下文数据上预训练的循环神经网络，可视化同一数据集上的误差曲面。

1. 预训练 RNN 训练：1 个符号到 2 个符号的转换

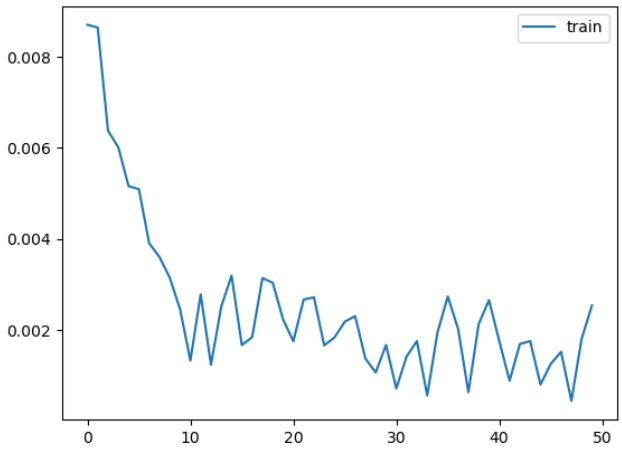

图 4. 显示 50 个训练轮次后 1-2 次转换的误差曲面的图表。

2. 预训练 RNN 训练：2 个符号到 3 个符号的转换

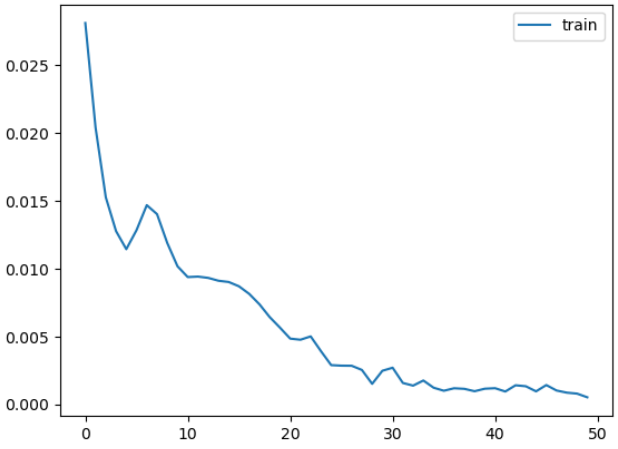

图 5. 展示经过 50 个训练轮次后 2-3 转换误差曲面的图。

3. 预训练 RNN 训练：3 个符号到 4 个符号的转换

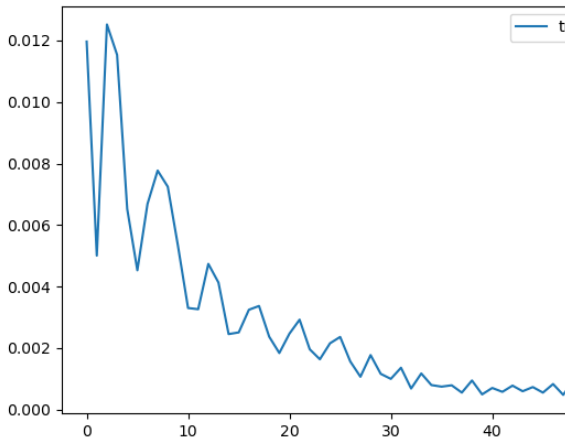

图 6. 展示经过 50 个训练轮次后 3-4 转换误差曲面的图。

VI. FBD 程序自动生成

我们引入动作模型这一概念，它是一种时间同步模型，可在离散设计时间步产生结果。在 FBD 程序的语境中，t=0 时刻的动作模型会执行 FBD 程序设计中第一个符号的初始选择。另一个动作模型则会在下一个设计时间步选择第二个功能块，或更一般地说，选择下一个符号。设计时间是一个离散随机变量，只有在执行设计动作时才会取值，也就是说，只有在选择一个符号或功能块并将其添加到 FBD 程序时才会取值。

在这一范式中，FBD 程序被视为一组时间同步的动作模型的实现。时间 t的动作模型A 学习符号宇宙的概率分布，并选择使条件概率最大化的符号或功能块。

假设对于每组符号 $U = \{u_i\}$，我们定义 FBD 程序 S 的概念，使得在每个设计步骤中，动作模型

$$A_t: U \rightarrow S^k$$

其中，t∈= 1,...,N，且为离散设计时间步。

K 对 U 上可能的 FBD 程序集合进行索引

我们认为，具有相同目标的不同工厂和 FBD 程序中观察到的差异是工程偏好、经验和设计指南解读的函数，并且对于特定任务而言，这种差异被S^k 在U 上的统计分布所体现。

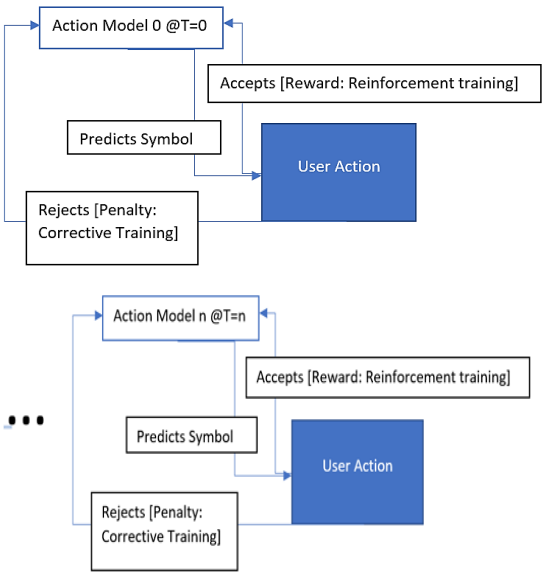

图 7. FIONA-4 中针对唯一时间步上的单一动作训练动作模型

我们介绍如图 7 所示的 FIONA-4 机器学习（ML）系统，该系统对神经网络模型（循环神经网络）进行定位，并根据处理类似问题的工程师的操作对其进行训练，以学习这种变化。每个已定位神经网络模型集群都会在每个设计时间步 t 为唯一的神经网络模型分配一个训练任务，其工作是学习 S_{t-1}^{k} 上 U 的概率分布，以便随后在时间 t 执行选择最佳 $u \in U$ 的功能；对于给定的任意“样本外”任务，如果设计在时间 $t-1$ 的状态为 $S_{t-1}^{arb.\ task}$。

动作模型的目标定义得更加明确，如下所示：

$$A_t: argmax\ (U \mid S_{t-1}^{k}) \rightarrow S_t^k$$

各个时间同步动作模型训练的示意图请参见上方图 7。

FIONA-4 ML 将局部化的神经网络模型集群联合起来，并以此为基础，通过自动接受动作模型 A_t 的选择，实现 FBD 程序自动生成系统。

VII. 结论

展示了一种训练启发式方法的新方法，并介绍了其对使用客户数据进行训练时收敛性的整体影响。此外，还将训练好的循环神经网络用作生成模型，通过递归方式使用循环神经网络的预测结果来创建 FBD 程序，或者更一般地，选择基于设计状态或先前模型选择而具有最高条件概率的功能块。

参考文献

[1] 国际电工委员会（IEC），《IEC61131-3：可编程控制器国际标准——第3部分：编程语言》，2013年。[2] IEC 61499，《功能块——第1部分：架构（第二版）》，国际电工委员会，瑞士日内瓦，2012年。[3] Qamsane, Y., Tajer, A., & Philippot, A. (2017). 基于事件排序关系的自动化制造系统分布式控制综合、验证与实施方法探索。《国际生产研究期刊》，55(21), 6235-6253。[4] Park, H. T., Kwak, J. G., Wang, G. N., & Park, S. C. (2010). 面向PLC仿真的被控对象模型生成。《国际生产研究期刊》，48(5), 1517-1529。[5] Darvas, D., Viñuela, E. B., & Majzik, I. (2016, July). 基于形式化规约语言的PLC代码生成。载于2016年IEEE 14th工业信息学国际会议（INDIN）论文集（第389-396页）。IEEE。[6] Mao, X., Zhang, Y., Shi, J., Huang, Y., & Li, Q. (2022). 采用Event-B的精化开发方法，以提升PLC程序的安全性。《计算机程序科学》，215, 102763。[7] Qamsane, Y., Tajer, A., & Philippot, A. (2017). 采用Grafcet实现的制造系统分布式监督控制设计综合方法。《国际生产研究期刊》，55(15), 4283-4303。[8] Kaid, H., Al-Ahmari, A., & Li, Z. (2020). 基于彩色资源导向Petri网的梯形图，用于可重构制造系统中的PLC实现。《IEEE Access》，8, 217573-217591。[9] Li, Y., Wang, Y., Zhu, G., Yin, L., & Zhang, H. (2022). 使用Petri网诊断基于PLC的离散事件系统故障。《测量与控制》，00202940221117098。[10] Oliveira, G. F., Candido, R. M. F., Gonçalves, V. M., Maia, C. A., Cottenceau, B., & Hardouin, L. (2020). Max-Plus代数中的离散事件系统控制：在制造系统中的应用。《IFAC论文集》，53(4), 143-150。[11] Ammour, R., & Amari, S. (2015). 使用（max,+）代数对网络化控制系统建模并评估其时序性能。《国际系统科学期刊》，46(1), 18-30。[12] He, J., Bartocci, E., Ničković, D., Isakovic, H., & Grosu, R. (2022, May). DeepSTL：从英文需求到信号时序逻辑。载于 44th 软件工程国际会议论文集（第610-622页）。[13] Farooqui, A., & Fabian, M. 使用主动学习为未知被控对象模型综合监督器。载于2019年IEEE 15th 自动化科学与工程国际会议（CASE）论文集（第502-508页）。IEEE。[14] Howar, F., & Steffen, B. (2018). 实践中的主动自动机学习。载于《动态软件分析的机器学习：潜力与局限》（第123-148页）。Springer，瑞士Cham。