非线性优化和最优控制问题中逻辑约束的精确连续重构

Jad Wehbeh [a]、Eric C. Kerrigan [a]

[a] *Imperial College London*、 *United Kingdom*

arXiv:2601.03906v1 [eess.SY] 7 Jan 2026

摘要

许多非线性最优控制和优化问题都涉及将连续动力学与离散逻辑条件相结合的约束。标准方法通常依赖混合整数规划，这会带来可扩展性方面的挑战，并需要专用求解器。本文提出了一种针对广泛类别逻辑约束的精确重构方法，将其表示为不含二进制变量的表达式；这些表达式的可微性与底层谓词的可微性一致，从而能够将其直接整合到非线性规划模型中。我们的方法将任意逻辑命题重写为合取范式，将其转换为等价的最大–最小约束，并应用保留精确可行集的平滑过程。该方法在两个基准问题上进行了评估：带避障约束的四旋翼轨迹优化问题，以及带时序逻辑约束的混合双水箱系统。结果表明，与现有的二进制变量消除技术相比，该方法能够更加稳定且高效地获得最优解。

Key words: 混合系统；可微逻辑约束；时序逻辑约束；约束平滑；无整数变量重构；非线性最优控制；连续优化；混合整数优化。

1 引言

在许多最优控制问题中，系统行为不仅受连续动力学支配，也受影响状态演化和约束激活的离散逻辑条件支配。这些条件通常通过二进制变量或整数变量进行编码，并可采用多种混合整数方法求解 [17,23]。然而，在最优控制场景中，混合整数优化可能并不理想，甚至难以实施，尤其是在系统为非线性或非凸，或离散变量的数量相对于连续动力学较少时。这些问题需要专用求解器，而这些求解器在不同应用中可能难以获取，并会带来显著的计算挑战 [13,15]。非凸连续动力学也使混合整数方法计算全局解的成本高得难以承受，实际上削弱了混合整数求解技术的一项关键优势 [6]。因此，通常更适合采用标准连续非线性优化求解器，因为这类求解器广泛可用，并且在处理高维问题时往往具有更好的可扩展性。在存在可微重构的情况下，这一点通常十分明显：可微重构的表现往往显著优于原始方法，而这类方法一旦存在，通常显著优于原始方法 [11]。本文将 [21] 的工作推广，为一般逻辑约束提供一种精确重构，将其表示为具有相同解集的连续函数。

Email addresses: j.wehbeh22@imperial.ac.uk（Jad Wehbeh）、e.kerrigan@imperial.ac.uk（Eric C. Kerrigan）。

1.1 研究现状

在最优控制的背景下，人们已经提出了多种用于建模逻辑约束的方法。最简单的形式化方法是直接使用整数变量描述问题中的逻辑关系：当某个条件成立时取值为 1，否则取值为 0。通常，这是通过采用大M法 [4], [1, 第 4] 章，或引入适当的互补约束 [2]来实现的。然而，由于这些方法引入了离散优化变量，直接求解必然需要使用混合整数求解器。

可以通过等式约束（如 $x(x-1) = 0$）将二进制变量嵌入连续优化求解器中，以强制实现 $x \in \{0,1\}$，但众所周知，这类方法在实践中表现不佳。另一种方法是使用对数求和指数函数 [5, Chap. 3], Sigmoid 函数 [10], 或类似技术来近似二进制行为。这些近似方法需要在数值性能与精度之间权衡，并且无法保证逻辑行为完全精确。

对于某些问题，混合整数问题的连续松弛可以在不强制满足整数性的情况下得到可行解 [9]，但这些方法的适用范围有限，不适用于本文所讨论的一般非线性最优控制问题。

时序逻辑约束为在优化中表达逻辑提供了另一种方式，并能自然地表示随时间变化的行为 [24]。然而，标准方法通常会将这类约束转化为混合整数规划 [14]，而这类规划存在前文所述的难题；或者采用在非线性优化中表现不佳的 SMT 求解器 [3]。将时序逻辑直接应用于非线性动力系统，会产生难以处理的非凸混合整数规划，因此需要对问题进行近似 [18]。约束规划是另一种选择，但当连续变量数量很大时，其扩展性较差 [19]。

与我们的问题最相关的方法见于近期文献 [7, 17]，这些文献提出了不同的方法来处理约束中的如下形式的逻辑蕴含：$g_1(x) \leq 0 \implies g_2(x) \leq 0$。然而，这些方法难以高效处理更一般形式的逻辑，因为蕴含不具备函数完备性，并且会迫使人们间接、展开地表示基本布尔运算符。在 [17] 中，作者利用同伦映射将蕴含约束纳入序列二次规划框架，从而避免使用二元变量。另一方面，[7] 提出了一种平滑的分段二次惩罚函数，利用连续优化求解器直接施加蕴含约束，并在一个可行性问题上展示了该方法的有效性。

1.2 贡献

本文的主要贡献如下：

- 我们提出了一种消除逻辑条件中二进制变量的新方法，同时保留可行集的精确性。如果逻辑命题是r可微的（包含于 $\mathcal{C}^r$），那么重新表述后的逻辑也 r可微。

- 我们为“直到”和“释放”等时序逻辑算子提供了紧凑的表示，其重新表述仅引入与输入维度呈线性关系的约束。
- 我们在带有条件障碍物避让的四旋翼轨迹规划，以及带有时序条件的混合双水箱控制问题上评估了我们的方法。在这两种情况下，与现有的可微逻辑公式相比，该方法取得了更高的成功率，求解速度也更快。

第 2 节将逻辑约束离散优化问题形式化。第 3 节随后介绍所提出的二进制消除过程，并在第 4 节详细介绍其在常见逻辑运算符中的应用。最后，第 5 节给出数值结果。

2 问题陈述

考虑一个决策变量为 $z \in \mathbb{R}^n$ 的逻辑约束优化问题。该问题可以写为

$$\min_{z} \; J(z) \tag{1a}$$
$$\text{s.t.} \quad g(z) \leq 0 \tag{1b}$$
$$h(z) = 0 \tag{1c}$$
$$L(z) \text{ is } \mathbf{true} \tag{1d}$$

其中，$J(\cdot): \mathbb{R}^n \to \mathbb{R}$ 是待最小化的代价函数，$g(\cdot): \mathbb{R}^n \to \mathbb{R}^{n_g}$ 是优化问题的解必须满足的一组不等式约束，$h(\cdot): \mathbb{R}^n \to \mathbb{R}^{n_h}$ 是一组类似的等式约束，而 $L(\cdot): \mathbb{R}^n \to \{\textbf{真}, \textbf{假}\}$ 是一个逻辑程序，其输出必须为 **真**。(1d) 中的约束使我们能够对 z 的演化过程中的任意不可微要求进行编码，并扩展了 (1a) 至 (1c) 中的标准光滑优化形式。所得描述具有足够的通用性，可以通过将状态 x 和控制输入 u 作为 $z = (x, u)$ 的一部分来表述最优控制问题，如第 5 节的示例所示。这样做时，系统动力学会加入到 $h(\cdot)$ 中；如果其演化依赖于逻辑，则加入到 $L(\cdot)$ 中。

现在，我们将 $L(\cdot)$的允许形式形式化。定义等式命题 $Q(\cdot): \mathbb{R}^n \to \{\textbf{真}, \textbf{假}\}$；当且仅当其关联函数$q(\cdot)$满足 $q(z) = 0$时，该命题的值为**真**，其中 $q(\cdot): \mathbb{R}^n \to \mathbb{R}$。类似地，定义不等式命题 $P(\cdot): \mathbb{R}^n \to \{\textbf{真}, \textbf{假}\}$；当其关联函数 $p(\cdot): \mathbb{R}^n \to \mathbb{R}$ 满足 $p(z) \leq 0$时，该命题的值为**真**，否则为**假**。

假设 1 令 $\mathcal{L}$ 表示由 $n_q \in \mathbb{N}_0$ 等式命题 Q_i、$i \in \{1, \ldots, n_q\}$和 $n_p \in \mathbb{N}_0$ 不等式命题 P_j、$j \in \{1, \ldots, n_p\}$ 组成的所有逻辑问题的集合。这些命题通过有限个逻辑运算符——非（$\neg$）、与（$\wedge$）和或（$\vee$）——组合，输出单个布尔值。在本文余下部分，令 $L \in \mathcal{L}$。

在L的定义中加入运算符 $\neg$、$\wedge$和 $\vee$，便足以表达命题之间的任意布尔关系，因为这组运算符是函数完备的 [8, 定理15D]。结合等式命题和不等式命题，这意味着 $\mathcal{L}$ 能够表示对 x 和 u的任意要求。原则上，L 也可以采用结构化方式，将约束 g 和 h纳入其中。不过，我们选择将它们分开处理，以区分常规光滑约束与那些本质上需要逻辑结构的约束。

在本文中，我们的目标是将(1d)中的逻辑约束替换为不含二元变量的等式约束和不等式约束，并将其嵌入 g 和h 中，同时不改变问题 (1)的可行集。由此得到一个标准的连续最优控制问题，可采用常规非线性规划方法求解。

3 连续约束重构

本节将证明，在一些温和条件下，任何满足假设 1 的逻辑问题都存在一个连续重构，且其解集与原问题完全相同。为此，我们先介绍以下结果，其证明展示了重写此类问题时所用的构造性过程。

3.1 逻辑简化

引理 1 $\mathcal{L}$中的任何问题都可以仅使用不等式命题、 $\wedge$和 $\vee$运算符来重写。

证明。 我们首先说明，等式命题可以用不等式命题等价地表示。为此，我们引入另外一对不等式

$$P^+_{n_p+i} \wedge P^-_{n_p+i} \equiv Q_i \qquad \forall i \in \{1,\dots,n_q\} \tag{2a}$$
$$p^+_{n_p+i}(z) \coloneqq q_i(z) \tag{2b}$$
$$p^-_{n_p+i}(z) \coloneqq -q_i(z) \tag{2c}$$

使得 $P^+_{n_p+i}$ 强制满足 $q_i(z) \le 0$，而 $P^-_{n_p+i}$ 强制满足$-q_i(z) \le 0$。这两者同时成立，当且仅当 $q_i(z) = 0$。接下来，我们利用德摩根定律将否定消去，即将 $\neg$ 运算符下推至命题层级：

$$\neg(P_1 \wedge P_2) = \neg P_1 \vee \neg P_2 \tag{3a}$$
$$\neg(P_1 \vee P_2) = \neg P_1 \wedge \neg P_2 \tag{3b}$$
$$\neg\neg P = P. \tag{3c}$$

这样， $\mathcal{L}$ 中的任意问题都可以表示为使用 $\wedge$、 $\vee$和 n_p+2n_q （可能取反的）不等式命题。接着，我们注意到以下事实，从而完全消除取反：

$$\neg[p_i(z) \le 0] \iff [p_i(z) > 0] \qquad \forall z \in \mathbb{R}^n,\ i \in \{1,\dots,n_p+2n_q\}. \tag{4}$$

对于任意严格不等式，我们知道：

$$p_i(z) > 0 \iff \exists \eta_i \in \mathbb{R} :\ -p_i(z) + \exp(\eta_i) \le 0 \tag{5}$$

定义取反后的命题

$$\bar{P}_i \equiv \neg P_i \qquad \forall i \in \{1,\dots,n_p+2n_q\} \tag{6a}$$
$$\bar{p}_i(z,\eta_i) \coloneqq -p_i(z) + \exp(\eta_i). \tag{6b}$$

由 (5) 可知，为每个取反后的不等式引入一个辅助变量 η_i ，通过施加条件 $\neg P_i \ \bar{p}_i(z,\ \eta_i) \ \le\ 0$ 即可满足该条件。为保持假设 1 的结构，随后可用这些新的辅助变量 η 构成的向量扩充优化问题的决策向量，并将其重新定义为 $z' = (z,\ \eta)$。

因此， $\mathcal{L}$ 中的任意问题都可以改写为仅使用不等式命题、 $\wedge$ 和 $\vee$ 运算符 □

在有限精度求解器中，必须将变量 η_i 保持在界限范围内，以避免数值溢出或下溢。此外，约束只能在规定的容差范围内得到满足。因此，在实际应用中，通常只需选择 $\bar{p}_i = -p_i$，或将 $\exp(\eta_i)$ 固定为小于优化器容差的值，从而无需显式引入辅助变量 η。基于这些原因，并为使符号表示更简洁，在本文余下部分中，我们仍将 z 视为完整的决策向量。如果需要在无限精度下使可行集完全等价，则 z 也可以解释为包含辅助变量向量 η。

3.2 最大-最小重构

接下来，我们尝试摆脱逻辑运算符 $\wedge$ 与 $\vee$。这些运算符无法在连续优化求解器中进行内在表示，因此我们将其替换为使用连续（但不一定可微）最大值和最小值算子的形式。

引理 2 对于任意问题 $L \in \mathcal{L}$，以及 $z \in \mathbb{R}^n$ 的任意选择，以下陈述是等价的：

- *The constraint of* (1d) *holds*
- $$\bigwedge_{i=0}^{n_\wedge} \bigvee_{j=0}^{n_\vee(i)} P_{i,j} \ \textit{is}\ \textbf{\textit{true}} \tag{7}$$
  其中，(7) 是根据引理 *1* 得到的 L的合取范式（*CNF*）。该形式包含由 $\wedge$ 运算符连接的 $n_\wedge + 1$ 个子句，每个子句包含 $n_\vee(i)\ + 1$ 个不等式命题，这些命题由 $\vee$ 运算符连接，其中 $i \in\{0,\dots,n_\wedge\}$ 对应于子句的索引。
- $$\max_{i\in\{0,\dots,n_\wedge\}} \ \min_{j\in\{0,\dots,n_\vee(i)\}} \ p_{i,\ j}(z) \le 0 \tag{8}$$
  其中 $p_{i,\ j}$ 在 (8) 中对应于 $P_{i,j}$ 在 (7) $\forall i \in \{0,\dots,n_\wedge\}, j \in \{0,\dots,n_\vee(i)\}$.

证明。 (1d) 与 (7) 之间的等价关系相对直接。根据引理 1， L 可以在不使用 $\neg$运算符且不包含等式命题的情况下表示。此外，任意命题公式都存在一个与之等价的合取范式（即析取的合取，或由与、或构成的形式），且该形式保留真值 [8, 第 1]章。因此，(1d) 总能改写为 (7)。

现在我们证明 (7) 与 (8) 等价。考虑一个由下式定义的子句 i

$$[p_{i,0}(z) \le 0] \vee \cdots \vee \left[p_{i,n_\vee(i)}(z) \le 0\right]. \tag{9}$$

当且仅当至少一个不等式成立时，该子句为**真**，即

$$\min_{j\in\{0,\dots,n_\vee(i)\}} p_{i,j}(z) \le 0. \tag{10}$$

类似地，多个子句的合取

$$\bigwedge_{i=0}^{n_\wedge} \left(\min_j p_{i,j}(z) \le 0\right) \tag{11}$$

当且仅当其中最大的内部最小值也满足该不等式时为真，即

$$\max_{i\in\{0,\dots,n_\wedge\}} \min_{j\in\{0,\dots,n_\vee(i)\}} p_{i,j}(z) \le 0. \tag{12}$$

这证明了 (7) 与 (8) 等价。结合这两个步骤，证明即告完成。□

回顾一下，(7) 和 (8) 中出现的不同命题总数为 $n_p + 2n_q$。转换为合取范式不会引入任何新命题，只会重新排列或重复已有命题。因此，记号 $P_{i,\,j}$ 应理解为为了方便记号而对命题进行简单的重新编号。

3.3 约束平滑

为了实现对 (8) 的平滑处理，我们首先引入以下假设。

假设 2 对于所有指标 $i \in \{0,\dots,n_\wedge\}$ 和 $j \in \{0,\dots,n_\vee(i)\}$，函数 $p_{i,j} \in \mathcal{C}^r$ 均为 r 次连续可微函数，$r \in \{0,\dots,+\infty\}$。

假设 2 并不苛刻，它只是为定理 1 的证明提供了可微阶数。对于许多问题，可以通过引入适当的逻辑，在不可微点处分解逻辑命题，从而提高可微阶数 r。

定理 1 设某个约束具有 (1d) 的形式，并且 $L \in \mathcal{L}$ 满足假设 *1* 和假设 *2*。那么，该约束可以等价地改写为

$$\sum_{j=0}^{n_\vee(i)} \lambda_{i,j}\, p_{i,j}(z) \le 0, \quad \forall i \in \{0,\dots,n_\wedge\}, \tag{13}$$

且不改变可行解集。这里，

- $\lambda_i := (\lambda_{i,\,0},\ \lambda_{i,\,1},\ \dots\dots,\ \lambda_{i,\,n_\vee(i)}) \in \Lambda_{n_\vee(i)+1}$ 是一组辅助平滑变量 $\forall i \in \{0,\dots,n_\wedge\}$，其中 Λ_k 表示 $\mathbb{R}^k$ 中的概率单纯形：

$$\Lambda_k := \left\{ l \in \mathbb{R}^k \;\middle|\; \sum_{i=1}^{k} l_i = 1,\, l_i \ge 0 \right\}.$$

- $p_{i,j}(z)$, $n_\wedge$, *and* $n_\vee(\cdot)$ *are as defined in* (8).

此外，(13) 中的约束的可微阶数与函数 $p_{i,j}$ 各自的可微阶数相同，因此至少是 r 次可微的。

证明。引理 2 允许我们用 (8) 的约束替换 (1d) 的约束，而不改变解集。然后，在假设 2 下，我们可以应用 [12] 的平滑方法，将有限个最大值分解为 $n_\wedge + 1$ 个不同的约束，并对每个最小值进行平滑，从而得到 (13) 的形式。根据 [12], 的主要结果，这组平滑约束与 (8) 具有相同的可行域，由此证明 (1d) 与 (13) 等价。在假设 2 下，对于所有 i，映射 $(z,\lambda) \mapsto \sum_{j=0}^{n_\vee(i)} \lambda_{i,j}\, p_{i,j}(z)$ 是 $\mathcal{C}^r$。□

3.4 无二进制变量的最优控制问题表述

利用定理 1，可以将 (1) 形式的问题转化为等价问题

$$\min_{z,\lambda} \ J(z) \tag{14a}$$
$$\text{s.t.} \quad g^*(z,\lambda) \le 0 \tag{14b}$$
$$h^*(z,\lambda) = 0 \tag{14c}$$

其中，$g^*(\cdot,\cdot) : \mathbb{R}^n \times \prod_{i=0}^{n_\wedge} \Lambda_{n_\vee(i)+1} \to \mathbb{R}^{n_g^*}$ 将 g 的约束与重述后的逻辑约束以及针对 λ 变量的正性约束组合起来；$h^*(\cdot,\cdot) : \mathbb{R}^n \times \prod_{i=0}^{n_\wedge} \Lambda_{n_\vee(i)+1} \to \mathbb{R}^{n_h^*}$ 将 h 的约束与 Λ 集合定义中的单位约束组合起来；$n_g^* := n_g + n_\wedge + 1 + \sum_{i=0}^{n_\wedge} (n_\vee(i) + 1)$，$n_h^* := n_h + n_\wedge + 1$。

该问题表述是 r 可微的，可以直接使用标准连续优化算法求解。由此得到的约束 g^* 通常是非凸的，因此会产生多个局部极小值；然而，这种非凸性源于逻辑本身，而非重新表述。重要的是，满足逻辑的局部极小值很容易与不可行的局部极小值区分开来，因此可以通过多起点方法有效地找到可行轨迹。我们还注意到，合取范式表示并非唯一，在实践中，某些选择可能会带来显著更好的性能，如第 4 节所示。

4 常见逻辑运算符

本节介绍如何将一些常见的时序逻辑运算符表示为合取范式，从而能够高效地构造式 (7) 所示形式的表示。为此，我们引入两个逻辑命题 A 和 B，它们可以在离散有限时域问题中从时间步 0 到 N 之间的任意时刻进行检验。我们用 $A[i]$ 表示在时间步 i 检验的命题 A。

4.1 蕴含运算符

蕴含运算符 $A[i] \implies B[j],\ i,j \in\{0,\dots,N\}$要求：如果 $A[i]$ 成立，则 $B[j]$ 成立。它可以轻松地应用于我们的框架，因为使用基本联结词表示的等价形式已经是合取范式：

$$\neg A[i] \vee B[j] \tag{15}$$

4.2 等价运算符

等价运算符 $A[i] \leftrightarrow B[j], i,j \in \{0,\dots,N\}$要求：当且仅当 $A[i]$ 也成立时，$B[j]$ 才成立。它可以改写为 ($A[i] \implies B[j]$) ∧($B[j] \implies A[i]$)，其等价的合取范式为

$$(\neg A[i] \vee B[j]) \wedge (\neg B[j] \vee A[i]) \tag{16}$$

4.3 直到运算符

直到运算符 $(A \,\mathcal{U}\, B)[i], i \in \{0,\dots,N\}$ 在线性时序逻辑中很常用，它要求从起始时间步 i 开始，A 必须保持**真**，直到B 为**真**，并且 B 必须在时间步 N之前取**真**值。直到运算符的形式化定义如下

$$(A \,\mathcal{U}\, B)[i] \ \equiv\ \exists j \geq i, j \leq N \ :\ B[j] \wedge \bigwedge_{k=i}^{j-1} A[k]. \tag{17}$$

如果直接用合取范式表示，这会转化为 $(N-i+1)^2$ 个析取子句。我们不采用这种做法，而是断言如下内容。

引理3 $(A \,\mathcal{U}\, B)[i]$ 可以表示为 $N-i+1$ 个子句，如下所示

$$\bigwedge_{j=i}^{N-1} \left(A[j] \ \vee \ \bigvee_{k=i}^{j} B[k] \right) \ \wedge \ \bigvee_{j=i}^{N} B[j]. \tag{18}$$

证明。我们首先说明（17）如何推出（18）。由（17）可知，在 i 与 N 之间存在最小的 j^*，使得 $B[j^*]$ 为**真**，并且 $A[j]$ 必须满足 $\forall j < j^*$。$B[j^*]$ 的存在意味着 $\bigvee_{j=i}^{N} B[j]$ 成立，并且（18）的其余部分同理可得，因为对于 $j < j^*$，$A[j]$ 为**真**；对于 $j \geq j^*$，$\bigvee_{k=i}^{j} B[k]$ 为**真**。

接下来，我们说明为什么 (18) 蕴含 (17)，因此这两种形式等价。与之前一样，存在满足 $B[j^*]$的最小 j^*，可由项 $\bigvee_{j=i}^{N} B[j]$得出。我们还知道，$\forall j < j^*$（即 (18) 的第一部分）意味着 $A[j]$ 必须为**真**，因为根据 j^*的定义，$\bigvee_{k=i}^{j} B[k]$ 为**假**。因此，(17) 在 j^* 处成立，满足存在性要求。□

4.4 释放算子

释放算子（$A \,\mathcal{R}\, B$）$[i], i \in \{0,\dots,N\}$要求从时间步 i 开始，B 保持**为真**，直到且包括 A 变为**真**的时刻。其形式化表达为

$$(A \,\mathcal{R}\, B)[i] \ \equiv\ \forall j \geq i, j \leq N \ :\ B[j] \vee \bigvee_{k=i}^{j-1} A[k]. \tag{19}$$

将 $\forall$ 算子转换为逻辑与，可得到合取范式

$$\bigwedge_{j=i}^{N} \left(B[j] \vee \bigvee_{k=i}^{j-1} A[k] \right). \tag{20}$$

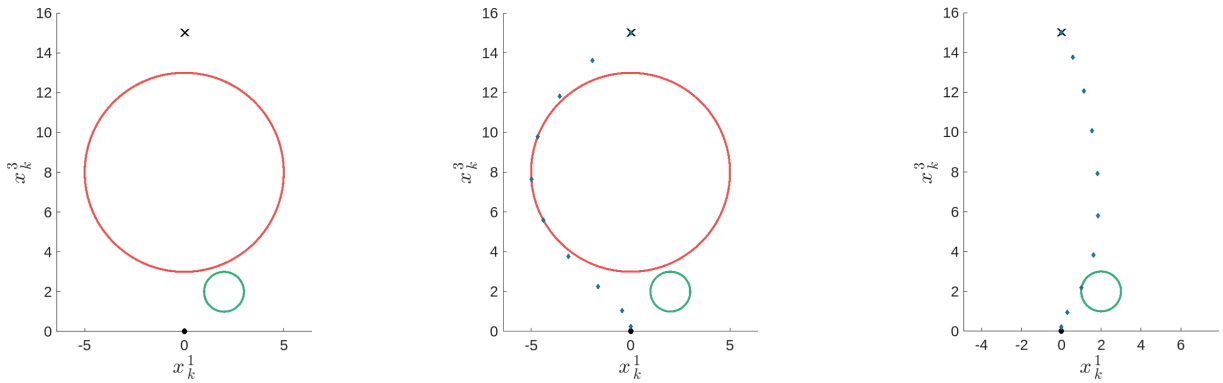
5 数值示例

5.1 四旋翼控制

考虑具有以下动力学的四旋翼控制问题

$$\begin{bmatrix} x_{k+1}^1 \\ x_{k+1}^2 \\ x_{k+1}^3 \\ x_{k+1}^4 \\ x_{k+1}^5 \\ x_{k+1}^6 \end{bmatrix} = \begin{bmatrix} x_k^1 \\ x_k^2 \\ x_k^3 \\ x_k^4 \\ x_k^5 \\ x_k^6 \end{bmatrix} + T_s \begin{bmatrix} \left(x_k^2 + x_{k+1}^2\right)/2 \\ \sin(x_k^5)\left(u_{k+1}^1 + u_{k+1}^2\right)/\gamma \\ \left(x_k^4 + x_{k+1}^4\right)/2 \\ \cos(x_k^5)\frac{\left(u_{k+1}^1 + u_{k+1}^2\right)}{\gamma} - g_{\circlearrowleft} \\ \left(x_k^6 + x_{k+1}^6\right)/2 \\ \ell(u_{k+1}^1 - u_{k+1}^2)/I \end{bmatrix} \tag{21}$$

其中，x_k^1 至 x_k^6 是与连续值 r、$\dot{r}$、s、$\dot{s}$、ψ和 $\dot{\psi}$ 分别对应的离散时间四旋翼状态，如图2所示。这里，u_k^1 和 u_k^2 是时间步 k时的推力，$T_s = 0.25$s为离散化时间步长，$\gamma = 0.15$kg为飞行器质量，$I = 0.00125$kg.m^2 为转动惯量，ℓ =0.1 m为电机力臂，$g_{\circlearrowleft} = 9.81$m/s^2 为作用于系统的重力。产生的推力



(a) 穿过绿色圆圈可避免经过红色障碍物。 (b) 次优轨迹示例。 (c) 最优解轨迹。

图1. 问题1中的障碍物（红色）、逻辑触发区域（绿色）、目标（黑色）和解轨迹（蓝色）。

受到约束，使得 $0 \le u_k^i \le 2\ \forall i \in \{1,2\},\ k \in \{1,\dots,N\}$ 在 $N = 10$ 个预测时间步内成立。系统从 $x_0 = [0,0,0,0,0,0]$初始化，并尝试最小化目标函数

$$\sum_{k=1}^{N} (u_k^1)^2 + (u_k^2)^2. \tag{22}$$

该问题受终端约束 0 $x_N^1 =$ 和 $x_N^3 = 15$ 限制，并且必须满足逻辑程序

$$\begin{aligned} &\neg \bigvee_{i=\{2,3\}} \left\{(x_i^1 - 2)^2 + (x_i^3 - 1)^2 - 1 \le 0\right\} \\ &\implies \neg \bigvee_{i=\{5,\dots,9\}} \left\{(x_i^1)^2 + (x_i^3 - 8)^2 - 25 \le 0\right\} \end{aligned} \tag{23}$$

如图1所示，这一约束迫使四旋翼飞行器在前往目标的途中避开红色圆圈，除非它在时间步2或3穿过绿色圆圈。按照第3节和第4节所述的步骤，可以将该约束平滑化为

$$\begin{aligned} &\lambda_1 \left((x_2^1 - 2)^2 + (x_2^3 - 1)^2 - 1\right) \\ &\quad + \lambda_2 \left((x_3^1 - 2)^2 + (x_3^3 - 1)^2 - 1\right) \\ &\quad - \lambda_3 \left((x_i^1)^2 + (x_i^3 - 8)^2 - 25\right) \le 0, \end{aligned} \tag{24}$$

对所有 $i \in \{5,\dots,9\}$，其中 $\lambda \in \Lambda_3$。可以使用相同的平滑变量 λ $\forall i$，因为如果某个实例满足绿色圆圈条件，那么红色圆圈的条件就都无需适用。随后，在配备第11代 Intel® Core™ i7-11370H CPU（主频 3.30 GHz）和 16GB 内存的笔记本电脑上，使用 JuMP [16] 软件包和 Ipopt [20] 优化器在 Julia 中求解该问题。我们将本方法与两种方法进行比较：一种通过互补条件强制二进制变量取值，另一种是 Big M 方法，该方法用连续变量 μ 表示条件 $\{a \le 0\} \vee \{b \le 0\} \vee \{c \le 0\}$ $\{a \le \mu_1 M\} \wedge \{b \le \mu_2 M\} \wedge \{c \le \mu_3 M\}$，使得 $\mu_1\mu_2\mu_3 = 0$。

问题1的性能指标（表1）

	最优解数量	次优解数量	不可行解数量	平均成本
平滑值	813	143	44	22.24
完整性	325	518	157	28.67
大M	660	274	66	23.73
	平均时间	平均时间（可行解）		最大时间
平滑值	22.4毫秒	19.4毫秒		466.8毫秒
完整性	76.1毫秒	78.2毫秒		419.7 毫秒
大M	56.7 毫秒	37.9 毫秒		1782.4 毫秒

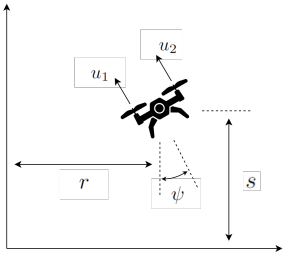


图2。四旋翼飞行器的水平位置（r）、高度（s）和倾斜角（ψ）示意图，取自 [22]。

由于该问题在所有情况下都是非凸的，且我们的求解方法通过局部优化器进行测试，因此我们使用从优化变量界限内的均匀分布中随机抽取的 1000 组初始值来评估各求解方法。表1所列结果表明，我们的方法优于其他方法，有 81.3% 的运行返回了最优解（如图1 (c) 所示），仅有 4.4% 的运行未能生成可行轨迹。即使排除耗时较长且未找到可行解的运行，平滑约束的计算时间仍然最短。再次指出，尽管平滑处理是精确的，由于优化问题具有非凸性，仍会出现次优结果。若使用全局优化求解器并采用我们提出的约束形式，则可保证得到最优轨迹。

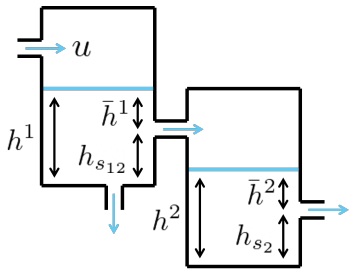


图3. 双水箱问题的设置，显示测量值（黑色）和水流（蓝色）。

5.2 双水箱问题

接下来，我们考虑图3所示双水箱问题的一个版本，其动力学为离散时间形式

$$h^1_{k+1} = h^1_k + T_s\left(\frac{u_{k+1}}{A_1} - \frac{s_1\sqrt{2g\,\bar{h}^1_k}}{A_1} - \frac{s_{12}\sqrt{2g\,\bar{h}^1_k}}{A_1}\right) \tag{25a}$$

$$h^2_{k+1} = h^2_k + T_s\left(\frac{s_{12}\sqrt{2g\,\bar{h}^1_k}}{A_2} - \frac{s_2\sqrt{2g\,\bar{h}^2_k}}{A_2}\right) \tag{25b}$$

其中，h^1_k 和 h^2_k 分别表示时间步 k时水箱1和水箱2中的水位，u_k 表示流入水箱1的水流，A_1和 A_2 分别表示水箱的横截面积。我们用s_1 表示水箱1底部开口的面积，用 s_{12} 表示水箱1与水箱2之间开口的面积，用 s_2 表示水箱2出口的面积。我们还定义

$$\bar{h}^1_k = \begin{cases} h^1_k - h_{s_{12}} & \text{if } h^1_k \geq h_{s_{12}}, \\ 0 & \text{otherwise}, \end{cases} \tag{25c}$$

$$\bar{h}^2_k = \begin{cases} h^2_k - h_{s_2} & \text{if } h^2_k \geq h_{s_2}, \\ 0 & \text{otherwise}. \end{cases} \tag{25d}$$

表示水位高于相应出口的高度，其中 $h_{s_{12}}$ 和 h_{s_2} 的定义如图3所示。所用参数值列于表2。对于所考虑的每个问题，输入流量受到约束，使得 $0 \leq u_k \leq 0.5 \ \forall k \in \{1,\ldots,N\}$，且优化问题的目标是最小化控制输入的平方和。

情形1： 从 $h^1_0 = 5$ 和 $h^2_0 = 5$出发，我们找到一条控制轨迹 u，使得 $h^1_N = 1.5$ 和 $h^2_N = 3.5$。将无二进制变量重构应用于约束 (25c) 和 (25d)，即可使用 Ipopt 求解该问题，并得到图 4 所示的轨迹。我们的求解器在 74.1% 的运行中仅收敛到最优解。使用大M或基于互补条件的可微重构，我们无法求解此问题。

表2 双水箱问题的系统参数

N	20	T_s	3 s	$h_{s_{12}}$	2 m
A_1	2 m^2	A_2	1 m^2	h_{s_2}	3 m
s_1	0.015 m^2	s_{12}	0.02 m^2	s_2	0.02 米^2

案例 2： 从 $h^1_0 = 5$ 和 $h^2_0 = 2$出发，我们找到一条控制轨迹 u，使得 $h^1_N = 2$ 和 $h^2_N = 4$。我们还要求解满足逻辑约束

$$\left[h^1_k \geq 4.5\right] \,\mathcal{U}\, \left[h^2_k \geq 4.5\right] \tag{26}$$

这可防止第一储罐中的液位降至 4.5 以下，直到第二储罐中的液位超过 4.5。我们按照情形1处理 (25c) 和 (25d) 的约束，并将第 4.3 节的方法应用于 (26)。图 5 中的最优轨迹出现在 38.3% 的运行中，而 86.2% 的运行得到了可行解。同样，使用平滑大 M 或基于互补性的重构无法求解该问题。

6 结论

本文提出了一种框架，通过精确无二进制重构将逻辑约束嵌入非线性优化问题。该方法将逻辑表示为最大-最小不等式，并对结果进行平滑处理，从而使连续优化求解器能够直接处理逻辑要求。四旋翼飞行器和双储罐问题的数值结果表明，所提出的重构方法具有诸多优势，包括计算速度更快、成本表现更佳。尽管逻辑表示的选择仍是一个开放问题，且随着逻辑结构变大，收敛性能可能下降，但对于逻辑复杂度适中的连续优化问题，该方法仍具有明显优势。

参考文献

[1] Mokhtar S Bazaraa、John J Jarvis 和 Hanif D Sherali。*Linear programming and network flows*。John Wiley & Sons, 2011。[2] Pietro Belotti、Pierre Bonami、Matteo Fischetti、Andrea Lodi、MicheleMonaci、Amaya Nogales-G´omez 和 Domenico Salvagnin。混合整数规划中指示器约束的处理。*Computational Optimization andApplications*, 65：545–566, 2016。[3] Calin Belta 和 Sadra Sadraddini。控制综合的形式化方法：一种优化视角。*AnnualReview of Control, Robotics, and Autonomous Systems*, 2(1)：115–140, 2019。[4] Dimitris Bertsimas、Ryan Cory-Wright 和 Jean Pauphilet。具有逻辑约束的混合整数优化问题的统一方法。*SIAM Journal on Optimization*, 31(3)：2340–2367, 2021。[5] Stephen P Boyd 和 Lieven Vandenberghe。*Convex optimization*。剑桥大学出版社，2004。

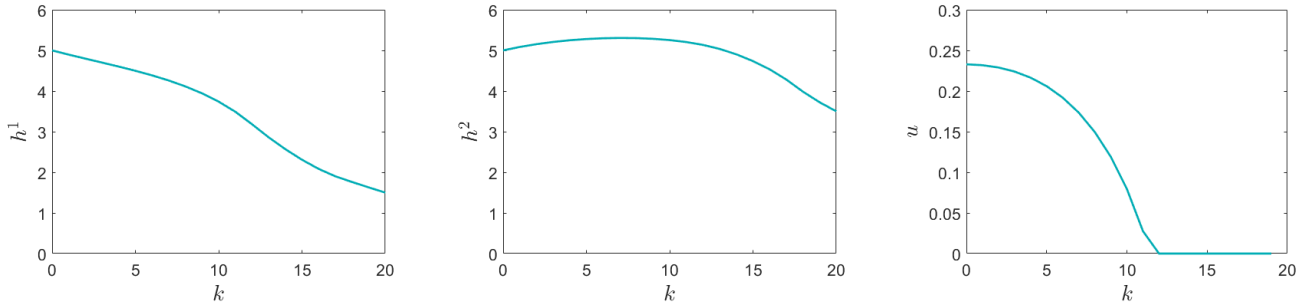


图 4. 情形1的最优轨迹。

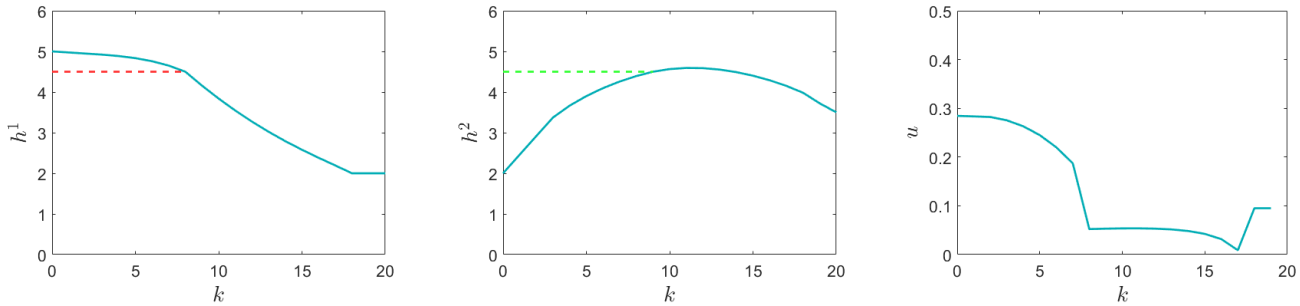


图 5. 最优　情形 2 的轨迹，其中对 h^2 （绿色）的条件释放了对 h^1 （红色）的约束　　.

[6] Samuel Burer 和 Adam N Letchford。非凸混合整数非线性规划：综述。*Operations Research and Management Science* 17 2 2012 ，（），。[7] Sonia Cafieri、Andrew R Conn 和 Marcel Mongeau。逻辑约束的连续象限惩罚公式。*Open Journal of Mathematical Optimization*，第 1–12 页，2023。[8] Herbert B Enderton。*A mathematical introduction to logic*。Elsevier，2001。[9] Oktay G¨unl¨uk 和 Jeff Linderoth。带指示变量的混合整数非线性规划的透视重构。*Mathematical Programming*，124：183–205，2010。[10]A Iliev、Nikolay Kyurkchiev 和 Svetoslav Markov。用某些 Sigmoid 函数逼近阶跃函数。*Mathematics and Computers in Simulation*，133 223：–234，2017。[11]Christian Kirches、Felix Lenders 和 Paul Manns。受约束混合整数最优控制的逼近性质与紧界。*SIAM Journal on Control and Optimization* 58 3 1371 1402, 2020 ，（）：–。

[12]C Kirjner-Neto 和 Elijah Polak。将具有最大最小约束的优化问题转化为标准优化问题。*SIAM Journal on Optimization*，8 4 887（）：–915，1998。[13] Jan Kronqvist、David E Bernal、Andreas Lundell 和 Ignacio E Grossmann。凸混合整数非线性规划求解器的综述与比较。*Optimization and Engineering*，20，2019。[14]Vincent Kurtz 和 Hai Lin。使用更少二进制变量的信号时序逻辑混合整数规划。*IEEE Control Systems Letters* 6 2635 2640, 2022 ：–。

[15]Leo Liberti。混合整数非线性规划中的不可判定性与困难性。*RAIRO-Operations Research*，53 1 81（）：–109，2019。[16] Miles Lubin、Oscar Dowson、Joaquim Dias Garcia、Joey Huchette、Beno ̂ıt Legat 和 Juan Pablo Vielma。JuMP 1.0：数学优化建模语言的最新改进。*Mathematical Programming Computation* 2023 ，.

[17]Danylo Malyuta 和 Beh ̧cet A ̧cıkme ̧se。用于航天器交会轨迹优化的快速同伦与离散逻辑。*Journal of Guidance, Control, and Dynamics*, 46 7 1262 1279, 2023 ( ): – 。[18]Ivan Papusha、Jie Fu、Ufuk Topcu 和 Richard M Murray。自动机理论与近似动态规划相遇：带时序逻辑约束的最优控制。载于 *2016 IEEE 55th Conference on Decision and Control (CDC)*，第 434–440 页。IEEE，2016。[19]Pascal Van Hentenryck、Laurent Michel 和 Fr ́ed ́eric Benhamou。非线性约束上的约束规划。*Science of computer programming*, 30(1-2):83– 118, 1998。[20] Andreas W¨achter 和 Lorenz T Biegler。关于面向大规模非线性规划的内点滤波线搜索算法的实现。*Mathematical Programming*, 106:25–57, 2006。[21]J. Wehbeh 和 E. C. Kerrigan。非线性优化和最优控制问题中的平滑逻辑约束。载于 *Proceedings of the 62th IEEE Conference on Decision and Control (CDC)* 2025 ，.

[22] Jad Wehbeh 和 Eric C Kerrigan。通过高效求解极小极大优化问题实现非线性系统的鲁棒输出反馈。载于 *2024 IEEE 63rd Conference on Decision and Control (CDC)* 8870 8875 2024 ，第 – 页。IEEE，。[23] Patrick M Wensing 、 Michael Posa 、 Yue Hu 、 Adrien Escande、Nicolas Mansard 和 Andrea Del Prete。面向动态足式机器人的基于优化的控制。*IEEE Transactions on Robotics* 40 43 ，：– 63, 2023。[24]Eric M Wolff、Ufuk Topcu 和 Richard M Murray。具有线性时序逻辑规范的非线性系统的基于优化的控制。载于 *Robotics and Automation* 5319 5325, 2014 *Proc. of Int. Conf. on* ，第 – 页。