《人工智能与机器学习进展》1（1）41-46 投稿日期：4/00；出版日期：10/00

意图驱动的城市空中交通重新调度

JEONGSEOK KIM JUSTIN@CLEVERPLANT.COM

Cleverplant，首尔，*04595*，大韩民国

Kangjin Kim KANGJINKIM@CDU.AC.KR

大韩民国全罗南道*58530*，草堂大学
无人机系统系

编辑： Kangjin Kim

arXiv:2512.15462v1 [cs.AI] 17 Dec 2025

摘要

由于资源受限，城市空中交通（UAM）领域对垂直起降机场的高效调度日益关注。对于调度问题，我们采用混合整数线性规划（MILP），该问题通常被建模为资源受限项目调度问题（RCPSP）。本文介绍了我们处理动态运行需求和人类提出的模糊重新调度请求的方法。具体而言，我们采用三值逻辑来解读含糊的用户意图，并利用决策树，提出一种融合答案集编程（ASP）与混合整数线性规划（MILP）的新型集成系统。该集成框架可优化计划，并透明地支持人类输入。借助这一系统，我们为可解释、自适应的城市空中交通调度提供了稳健的架构。

关键词： 城市空中交通（UAM）、答案集编程（ASP）、意图识别、资源受限项目调度问题（RCPSP）、重新调度、可解释人工智能（XAI）

1. 引言

城市空中交通（UAM）引入了一种新的交通范式，需要对垂直起降（VTOL）资源进行高效管理。在此背景下，调度问题被表述为资源受限项目调度问题（RCPSP），通常采用混合整数线性规划（MILP）方法求解。然而，城市空中交通的动态特性也要求人工操作员能够灵活地控制计划，以满足实时要求。

问题大多会在操作员的请求存在歧义时出现。在这种情况下，自动化系统必须先更准确地解释潜在意图，然后才能修改计划。这 41

引用： J. Kim 和 K. Kim，意图驱动的城市空中交通重新调度。《人工智能与机器学习进展》。2025；8(6)：6。© 1 Jeongseok Kim 和 Kangjin Kim。

这是因为误解意图会对效率和信任造成负面影响。为此，我们提出一种将基于混合整数线性规划的调度器与用于意图解释的答案集编程（ASP）模块相结合的系统。该模块创新性地使用三值逻辑和决策树来消除歧义。

为了便于理解系统在调度修订过程中的推理过程，我们采用了答案集编程（ASP）的声明式方法。该方法能够处理明确和模糊的输入，并推理如何妥善处理这些输入，同时保持可解释性。这提升了系统的响应能力，增强了用户的信任，并符合可解释人机交互规则。

我们此前研究过城市空中交通管理（UATM）系统中的人机交互[1, 2, 3, 4]。这些研究主要关注利用知识表示与推理处理航线变更和反应式事件（如绕行和垂直起降机场关闭）。在此基础上，我们近期研究了垂直起降机场资源调度问题。具体而言，我们在资源受限项目调度问题（RCPSP）表述中采用了答案集编程（ASP），以满足各类利益相关者的约束 [5]。本文基于上述研究，聚焦于垂直起降机场重新调度，探讨在动态运行环境下如何处理对人工操作员模糊请求的解读问题。

本文的贡献有三点：首先，我们将城市空中交通调度建模为资源受限项目调度问题，以便使用混合整数线性规划求解器计算计划；其次，我们引入了一种采用三值逻辑的基于答案集编程的意图识别机制；最后，我们提出了一个融合人类意图与自动化调度的可解释系统。接下来的章节将详细介绍相关工作、方法、实验和结论。

2. 相关工作

垂直起降机场是为在城市环境中运行电动垂直起降飞行器而建造的枢纽。 [6,7, 8] 中的研究表明，城市空中交通的效能在很大程度上取决于合理的调度。本质上， [9, 10] 中的作者指出，由于资源有限、需要与城市系统集成以及安全约束，调度这些作业十分复杂。因此，该问题被建模为资源受限项目调度问题。这属于 NP 难问题类，其目标是在满足资源和优先约束的同时优化总工期（参见 [11, 12]）。对于这类约束，混合整数线性规划已得到广泛应用[13, 14]，尽管在多个领域中，要据此求得最优调度方案需要付出较高的计算成本 [15]。此外，人们还开发了元启发式算法和各种优化策略，以应对不确定性 [16, 17, 18, 19, 20]。

[21, 22, 23] 中的研究表明，将人工操作员纳入自动化调度系统后，要准确识别协调过程中的用户意图会更具挑战性。尤其是，当人类输入含糊不清或定义不当时，消除歧义是关键问题。 [24, 25, 26] 中的研究证明，采用多值逻辑系统和决策树的方法在处理决策情境中的不确定性方面具有一定潜力。

声明式编程范式能够在系统化且可验证的框架内解决复杂问题。尤其是，答案集编程在知识表示与推理领域的有效性已得到证明。 [27, 28, 29, 30] 展示了其表达能力，能够让我们对复杂问题进行建模，例如不确定性下的调度和决策

。这一透明性符合人机交互领域可解释人工智能（XAI）研究的目标。清晰解释系统的决策过程，有助于建立用户信任、确保问责，并促进有效的人机协作（参见 [31, 32, 33, 34, 35]）。本研究通过答案集编程实现了一个能够响应人类意图的可解释城市空中交通调度系统，并遵循歧义消解规则。

3. 预备知识

3.1 资源受限项目调度问题

资源受限项目调度问题（RCPSP）旨在为任务$J = \{0..n+1\}$找到一个使项目完工时间（截止期限）最短的计划。该计划受任务之间的优先约束S^0以及一组可用可再生资源$R = \{1..m\}$的限制。必须遵守资源$R = \{1..m\}$的上限C_r。每个作业j都有持续时间p^j，并且在作业进行期间需要单位数量为u^0_{jr}的资源r。作业 0 和$n+1$是虚拟的起始和结束作业。

通用的混合整数线性规划（MILP）表述使用二元变量x_{jt}，其中$x_{jt}= 1$表示任务j在时间t开始，否则为 0。目标是使最后一个任务的开始时间最小化：

$$\min \sum_{t=0}^{H} t \cdot x_{n+1,t} \tag{1a}$$

$$\text{s.t.} \sum_{t=0}^{H-p_j} x_{jt} = 1 \qquad \forall j \in J \tag{1b}$$

$$\sum_{j \in J} \sum_{\tau=\max(0,t-p_j+1)}^{t} u^0_{jr} \cdot x_{j\tau} \leq C_r \qquad \forall r,t \tag{1c}$$

$$\sum_t t x_{st} - \sum_t t x_{jt} \geq p_j \qquad \forall (j,s) \in S^0 \tag{1d}$$

$$x_{jt} \in \{0,1\} \qquad \forall j,t \tag{1e}$$

这些约束确保每项任务恰好开始一次（1b）、资源容量不超限（1c），且满足优先关系（1d）。表1给出了一个资源受限项目调度问题实例。图 2b直观展示了与相关计划相对应的计划。

3.2 重新调度选项概述

表 2 根据 [36] 对相关文献的综述，列出了典型重新调度策略的分类。我们的框架围绕以下突出显示的选项展开：部分重新调度和资源重新分配，更确切地说，是对单个作业的开始时间或所需资源数量进行调整。

表1：示例资源受限项目调度问题实例数据（图 2b 中引用）。

Task	持续时间 (p_j)	需求（u_{jr}）			后继任务
		R1	R2	R3	
1	5	3	3		3
2	4	3		2	3
3	1			2	
4	3	2		3	5, 6
5	4		3		
6	3	2	1		
7	1			3	8
8	4	1		2	
9	5	2			10
10	3		3		

表 2：重新调度选项（本文涵盖高亮显示的行）

选项类别	**变更重点**	**优先关系**	**资源分配-ment**	**典型意图-pact**
全面重新排程	所有未来任务	Re-优化整个图	重新分配 all re-来源	High 成本；扰乱基线
部分重新调度	仅受影响的任务	修改 only af-受影响的链接	有限重新分配	平衡稳定性与优化-tion
时间偏移	开始时间调整	保持原有顺序	通常不变	最少更改；高稳定-ity
任务重排序	调整序列顺序	更改顺序，同时保留任务	May in-直接影响 re-来源	改善性能；较不稳定
资源 重新分配 tion	修改 资源数量	通常保留顺序	更改重新资源分配任务	目标资源冲突
结构性 修改 tion	插入/移除 任务，更改模式	移除或添加链接	May in-发展重新分配-tion	重大结构性变化

3.3 城市空中交通背景：意图驱动、可解释的重新调度

城市空中交通系统需要频繁且快速地进行重新调度，以维持运营。我们的系统采用可解释的答案集编程（ASP），并结合基于混合整数线性规划的资源受限项目调度，以有效应对这些变化。在城市空中交通中，可解释性在人机交互中发挥着重要作用，因为它通过为系统决策提供透明的推理过程来帮助建立信任。[31, 32]。

我们的方法是“意图驱动”的：它会解读用户请求调度变更时所隐含的目标。通过分析请求类型（开始时间或资源调整）以及可能含糊不清的用户输入，ASP模块确定适当的重新调度操作，并解释其原因。

范围： 本研究主要关注城市空中交通运行中的单任务重新调度，该过程通过部分重新调度或资源重新分配来实现。在此范围内，我们考虑两种需求选项：调整特定作业的开始时间，以及调整所需的资源。这一覆盖范围的一个重要方面是决定是否应保留任务原有的优先约束，或是否应将其移除，以便在调整过程中进行独立优化。

排除项： 该范围不包括全局调度重新优化、多个任务的重新排序，也不包括涉及添加或移除多个任务的重大结构性变更。

4. 交互式意图提取过程

本节详细说明如何从可能存在歧义的重新安排请求中提取用户意图，并构建相应的重新调度问题。

4.1 歧义重新调度请求表示

我们认为，针对特定任务Q 的初始重新调度请求t^* 可能存在歧义。该请求表示如下：

$$Q = (\delta, \rho, \eta, \theta, \tau^*, R^*)$$

其中：

- $\delta \in \{未知 0, 1\}$用于表示时间变更请求的优化类型。0 表示遵守其他任务的开始时间，仅允许局部变更；1 表示进行全局优化；“未知”表示存在歧义。
- $\rho \in \{未知\ 0,\ 1\}$用于表示时间变更请求中需要维持的关系类型。0 表示放宽t^*的优先约束；1 表示维持优先关系；“未知”表示存在歧义。
- $\eta \in \{未知 0,\ 1\}$用于表示资源数量变更请求的优化类型。0 表示遵循其他任务的开始时间，仅允许局部变更；1 表示进行全局优化；“未知”表示存在歧义。
- $\theta \in \{未知\ 0,\ 1\}$用于表示资源数量变更请求中需要维持的关系类型。0 表示放宽t^*的优先约束；1 表示维持优先关系；“未知”表示存在歧义。
- $\tau^* \in \mathbb{Z}_{\geq 0}$ 是任务t^*的期望开始时间。

• $R^* \in \mathbb{Z}_{m\geq 0}$ 是任务 t^*的期望资源需求向量，其中m 是资源类型的数量。

如果 δ，ρ，η，θ 中的某一项的值为“未知”，则需要通过交互过程来澄清用户的真实意图。

4.2 交互式意图提取

我们通过与用户交互，消除初始请求 Q 所存在的歧义。我们利用交互历史 这一概念来实现这一交互过程。我们将此历史记为 $\mathscr{H}$。 $\mathscr{H}$由一系列元组构成，每个元组记录一次交互。

$$\mathscr{H} = [(q^{(0)}, a^{(0)}), (q^{(1)}, a^{(1)}), \ldots, (q^{(T)}, a^{(T)})]$$

此处，$q^{(t)}$ 表示系统在t 步询问用户以澄清歧义的内容，而$a^{(t)}$ 则表示用户相应的回复。每个交互步骤都会作为一个新的（q，a）元组追加到 $\mathscr{H}$。

系统利用初始请求 Q 和累积的历史记录 $\mathscr{H}$，通过交互映射函数 $\mathscr{I}$确定用户已充分澄清的意图：

$$\mathscr{I} : Q \times \mathscr{H} \to \{0,1\}^4$$

输出为一个意图向量，其内容已充分澄清，如下所示：

$$\mathbf{b}^* = \mathscr{I}(Q, \mathscr{H}) = (b_1^{(s)}, b_2^{(s)}, b_1^{(r)}, b_2^{(r)})$$

其中：

• $b_{(s)}$ 1（已澄清的 δ）是一个布尔值，用于指示时间变更请求的优化类型。1 表示意图是进行全局优化，0 表示意图是在尊重其他任务的前提下进行局部优化。

• $b_{(s)}$ 2（已澄清的 ρ）是一个布尔值，用于指示时间变更请求的优先约束。1 表示应维持该关系；0 表示放宽该关系。

• $b_{(r)}$ 1（已澄清的 η）是一个布尔值，用于指示资源数量变更请求的优化类型。1 表示意图是进行全局优化，0 表示意图是在尊重其他任务的前提下进行局部优化。

• $b_{(r)}$ 2（已澄清 θ）是一个布尔值，用于指示资源数量变更请求的优先约束。1 表示应维持该关系；0 表示放宽该关系。

该向量$\mathbf{b}^*$ 明确规定了后续重新排程流程的条件。

5. 问题形式化

我们将重新排程流程定义为：针对任务t^*，根据请求Q，对基准计划S_0,应用转移算子 $\mathscr{R}$ 。

$$\mathscr{R} : (S_0, t^*, Q) \mapsto (S, E)$$

其中：

- **输入**：

  1. S_0 是当前的基准计划。2. $t^* \in J$ 是目标重新排程任务。3. Q 是一个包含可能存在歧义的重新排程请求的元组。

- **流程**：

  1. 意图提取：通过 $\mathscr{I}$(Q $\mathscr{H}$) 确定澄清后的意图向量 $\mathrm{b}^* = (\mathrm{b}_1^{(s)}, \mathrm{b}_2^{(s)}, \mathrm{b}_1^{(r)}, \mathrm{b}_2^{(r)})$。2. 参数更新：更新 t^* 的资源受限项目调度问题参数。首先，根据 τ^* 设置 t^* 的期望开始时间。其次，根据 R^* 更新资源需求 $\mathrm{u}_{r^*}\mathrm{r}$。3. 约束调整：根据 b^* 修订资源受限项目调度问题约束：– 优先关系：如果 $\mathrm{b}_2^{(s)} = 0$ （针对时间变更请求），则放宽 t^* 的约束；否则，保留 t^* 的约束。资源数量变更请求采用相同方法。– 范围：如果 $\mathrm{b}_1^{(s)} = 0$ （针对时间变更请求），则保留无关任务的开始时间；否则，对开始时间进行全局优化。资源数量变更请求采用相同方法。4. 重新排程优化：求解修改后的资源受限项目调度问题实例，得到新的调度 S。5. 解释生成：根据 Q、b^* 和结果生成 E，以说明 S。

- **输出**：

  – S：新的更新后计划。– E：对重新排程操作的解释。

符号说明。 意图变量$b_1^{(s)}$ ， $b_2^{(s)}$ ， $b_1^{(r)}$ ， $b_2^{(r)}$ 用于指定开始时间（s）和资源（r）请求的范围（1）与优先关系（2）处理方式。为简洁起见，当请求类型明确（开始时间或资源）时，我们将其简化为b_1 （范围）和b_2 （优先关系）。

5.1 问题陈述

问题 1 给定一个基准计划 S_0 以及针对任务t^*的（可能存在歧义的）重新排程请求 Q，首先通过 $\mathscr{H}$ 与用户交互，澄清其请求 Q 中的意图$\mathbf{b}^*$ ，然后执行意图驱动的重新排程流程 $\mathscr{R}$，以计算新的计划 S 和相应的解释 E。

5.2 解决方案概述

图 1 展示了系统的流程。用户提示会启动该流程。系统使用逻辑对重新安排请求进行分类和验证。解释请求后，决策树可以通过提示与用户交互，以消除歧义并确定用户意图$\mathbf{b}^*$。该意图用于识别并执行适当的重新安排操作（参数／约束修改），以解决资源受限项目调度问题。系统会返回更新后的计划 S 和解释 E 。中央数据库负责维护规则、决策树和计划。

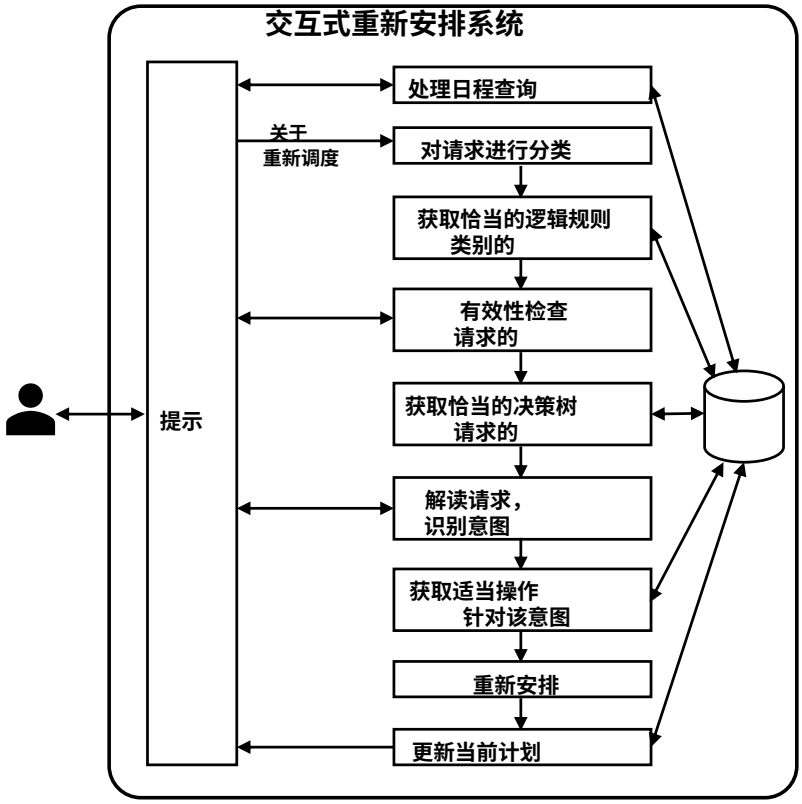


图 1：高层解决方案的简单流程

6. 提出的解决方案

本节介绍如何从可能存在歧义的请求中提取用户意图，以及后续的重新调度流程。

6.1 意图提取

流程首先验证用户请求，然后使用答案集编程（ASP）和交互式决策树方法解读其潜在意图。

6.1.1 请求的有效性检查

ASP规则会检查请求参数（任务ID、期望时间/资源）是否符合问题约束。

下面列出了开始时间变更请求的有效性检查。

$$\text{请求}(n,i)。 \tag{2}$$

$$valid_task \leftarrow request(N,I), task(N). \tag{3}$$

$$valid_time \leftarrow request(N,I), I \leq I2, step(I), total_dur(I2). \tag{4}$$

$$result(valid_request) \leftarrow valid_task, valid_time. \tag{5}$$

$$result(invalid_task) \leftarrow \neg valid_task, valid_time. \tag{6}$$

$$result(invalid_time) \leftarrow \neg valid_time, valid_task. \tag{7}$$

下面列出了资源数量变更请求的有效性检查。

$$\text{请求}(n,r,a)。 \tag{8}$$

$$resource(R) \leftarrow rsreq(N,R,A). \tag{9}$$

$$\text{限制}(R,L) \leftarrow \text{请求资源}(N,R,A), L = \#max\{L1 : rsreq(N,R,L1)\}. \tag{10}$$

$$valid_task \leftarrow request(N,R,A), task(N). \tag{11}$$

$$valid_resource \leftarrow request(N,R,A), resource(R). \tag{12}$$

$$valid_amount \leftarrow request(N,R,A1), A1 \leq A2, limit(R,A2). \tag{13}$$

$$\text{结果(有效_请求)} \leftarrow \text{有效_任务，有效_资源，} valid_amount. \tag{14}$$

$$\text{结果(无效_任务)} \leftarrow \neg\text{有效_任务。} \tag{15}$$

$$\text{结果(无效_资源)} \leftarrow \neg\text{有效_资源。} \tag{16}$$

$$\text{结果(无效_数量)} \leftarrow \neg\text{有效_数量。} \tag{17}$$

6.1.2 意图解释

答案集编程结合三值逻辑（$\top$，$\bot$，υ 分别表示真、假、未知）和外部原子（`#external sym/2`），用于对澄清意图的决策树建模。当信息未知时，系统会查询用户（`query/1`）（υ）。

下面的代码展示了适用于n个布尔决策点的通用 ASP 决策树生成逻辑。

$$\begin{aligned}
&\#external \quad sym(S,V): \quad symbol(S), value(V). && (18)\\
&\qquad\qquad symbol(1..n). && (19)\\
&\qquad\qquad value(\text{“}\top\text{”};\text{“}\bot\text{”}). && (20)\\
&1\{\top(S);\bot(S);\mho(S)\} \leftarrow symbol(S), value(V). && (21)\\
&\qquad\mho(S) \leftarrow \neg sym(S,\text{“}\top\text{”}), \neg sym(S,\text{“}\bot\text{”}),\\
&\qquad\qquad symbol(S). && (22)\\
&\quad contradiction(S) \leftarrow sym(S,\text{“}\top\text{”}), sym(S,\text{“}\bot\text{”}), symbol(S). && (23)\\
&\qquad\mho(S) \leftarrow contradiction(S), symbol(S). && (24)\\
&\qquad\qquad \leftarrow \neg\top(S), sym(S,\text{“}\top\text{”}), symbol(S). && (25)\\
&\qquad\qquad \leftarrow \neg\bot(S), sym(S,\text{“}\bot\text{”}), symbol(S). && (26)\\
&\qquad query(S1) \leftarrow O1 = \#min\{O : sym_order(S,O), \mho(S)\},\\
&\qquad\qquad sym_order(S1,O1), symbol(S1). && (27)\\
&\qquad 已观察(S,V) \leftarrow 对称(S,V)，\ 符号(S)，\ 值(V)。 && (28)\\
&\quad 已假定(S,\ \text{“}\neg\bot\text{”}) \leftarrow 已观察(S,\ \text{“}\top\text{”})\ \neg 对称(S,\ \text{“}\bot\text{”}),\\
&\qquad\qquad 符号(S)。 && (29)\\
&\quad 已假定(S,\ \text{“}\neg\top\text{”}) \leftarrow 已观察(S,\ \text{“}\bot\text{”})\ \neg 对称(S,\ \text{“}\top\text{”}),\\
&\qquad\qquad 符号(S). && (30)\\
&\qquad 已知(S,\text{“}\top\text{”}) \leftarrow 已观察(S,\text{“}\top\text{”}), 已假定(S,\text{“}\neg\bot\text{”}),\\
&\qquad\qquad 符号(S). && (31)\\
&\qquad 已知(S,\text{“}\bot\text{”}) \leftarrow 已观察(S,\text{“}\bot\text{”}), 已假定(S,\text{“}\neg\top\text{”}),\\
&\qquad\qquad 符号(S). && (32)\\
&\qquad\qquad \leftarrow 已知(S,\text{“}\top\text{”}), 已知(S,\text{“}\bot\text{”}),\\
&\qquad\qquad\ 符号(S)。 && (33)
\end{aligned}$$

我们假设一个默认符号顺序：

$$sym_order(i,i). \quad \triangleright \forall i \in \mathbb{N}^{+} \ \texttt{s.t.} \ \ 1 \le i \le n \qquad (34)$$

意图基于最终已知状态定义。对于$n = 2$：

$$\begin{aligned}
&意图(\text{“}op_00\text{”}) \leftarrow 已知(2,\text{“}\bot\text{”}), 已知(1,\text{“}\bot\text{”})。 && (35)\\
&意图(\text{“}op_01\text{”}) \leftarrow 已知(2,\text{“}\bot\text{”}), 已知(1,\text{“}\top\text{”})。 && (36)\\
&意图(\text{“}op_10\text{”}) \leftarrow 已知(2,\text{“}\top\text{”}), 已知(1,\text{“}\bot\text{”})。 && (37)\\
&意图(\text{“}op_11\text{”}) \leftarrow 已知(2,\text{“}\top\text{”}), 已知(1,\text{“}\top\text{”}). && (38)
\end{aligned}$$

算法1详细说明了交互式意图解释过程。它对请求进行分类和验证，然后迭代使用ASP决策树逻辑，并可能通过交互历史 $\mathscr{H}$ 向用户提问，直到意图 $\mathbf{b}^{*}$ （以 已知谓词 事实表示）得到完全解析或发生超时。它返回最终意图 $\mathbf{b}^{*}$、请求类型 γ和解释E。

算法1： 意图解释

输入： 一个基准计划 S_0, 、一个任务t^*和一个初始请求 Q
输出： 一个意图 $\mathbf{b}^*$、一个请求类型 γ和解释 E

根据给定输入对请求类型 γ进行分类;
使用公式 (2)–(7) 或公式 (8)–(17) 检查 Q 和 γ 的有效性;
如果 ¬result(valid_request) **则**
    返回 $\langle$∅, γ, {result/1}$\rangle$
$n \leftarrow 2$;
▷ 两种请求 γ类型的树深度相同。
通过公式 (18)–(38) 生成一棵包含 n 的决策树 Δ ;
foundIntention $\leftarrow \perp$;
$\mathscr{K}_{base}$ ←获取当前时间;
$\mathscr{H}$ ←NIL;
▷ 初始化交互历史
谓词 : = {} ;
▷ 初始化谓词字典
$\mathbf{b}^*$: = $\langle$ʊ, ʊ, ʊ, ʊ$\rangle$;
▷ 初始化 $\mathbf{b}^*$ 为 ʊs，表示未知
当 ¬foundIntention**时**
    如果 意图谓词在谓词中**则**
        foundIntention $\leftarrow \top$;
        $E \leftarrow$ {意图谓词，已知/2} ;
        更新 $\mathbf{b}^*$ ，使用 γ和 已知谓词;
        继续;
    否则，如果 query/1属于谓词 **则**
        对$C_{sym} \in$ 谓词[contradiction/1] **中的每一项执行**
            释放外部 对称/2 ，使用C_{sym} 和 {“⊤”，“⊥”} ;
        $\mathscr{H} \leftarrow \mathscr{H} + \langle i, \emptyset \rangle$;
        ▷ 来自 query/1 的符号索引i
        根据 $\mathscr{H}$，询问用户并等待时间 κ。;
        将外部 sym/2 指定为引用响应内容 $\mathscr{H}$;
    else
        ▷ 对于谓词 为空的情况
        将外部 sym/2 指定为引用给定的输入;
    执行 Δ，解析结果，并更新谓词 ;
    $\mathscr{K}_{curr}$ ←获取当前时间;
    如果 $\mathscr{K}_{curr} - \mathscr{K}_{base} \geq$ 时间_限制 **则**
        foundIntention $\leftarrow \top$;
        $E \leftarrow$ {超时/0} ;
返回 $\langle \mathbf{b}^*, \gamma, E \rangle$

6.2 RCPSP 求解器的预处理

算法 2 根据解释后的意图（γ,b_1,b_2）、期望开始时间 τ^* 和任务 t^* 所需的资源 R^* ，为标准资源受限项目调度问题求解器准备问题。它根据意图标志修改基线资源需求（$u^0 \rightarrow u'$）和优先关系（$S^0 \rightarrow S'$）。关键是，它确定必须在特定开始时间 τ_f 固定的任务集合 T_f。该集合始终包含在 τ^* 时开始的任务 t^*。如果 $b_1 = 0$ （局部范围），则尝试将 t^* 的前驱任务固定在其基线时间（S_0），但前提是这些任务与 t^* 或彼此之间不冲突，且不会因从 t^* 传播而来的冲突而必须保持灵活。可行性检查确保 t^* 的请求安排符合优先关系，且最终固定任务集合 T_f 满足资源可行性。输出为适合求解器使用的修改后问题实例（$\langle S', u', T_f, \tau_f \rangle$），或不可行状态。

该算法使用以下辅助函数:

- `CheckTotalConflicts`：识别在资源方面与t^* 在 τ^* 处冲突的基线任务。
- `PropagateFlexibility`：找出所有无法在基线时间固定的任务（冲突任务及其后继任务）。

- 检查资源可行性：验证给定的一组固定任务（T_f τ_f）是否资源可行。
- 构建后继映射：根据 S' 创建任务后继的查找映射。

算法 2：预处理

输入：

- 请求数据：目标任务t^*、期望时间 τ^*、期望资源 R^*、请求类型 γ、意图 b_1 和 b_2；
- 基线数据：任务 T、资源 R、持续时间 p、优先关系 $S0$、资源函数 $u0$、容量 C、规划时域 H、基准计划 S_0；

输出： $\langle S', u', Tf, \tau_f \rangle$，或带原因的不可行状态；

初始化 $u'_{tr} \leftarrow u^0_{tr}$ 对所有 t，r；▷ 有效资源需求

如果 $\gamma =$ 'r'，**则** ▷ 对于资源数量变更请求

  $u'_{t^*r} \leftarrow R^*_{t^*r}$ 对于所有 $r \in R$;

初始化 $S' \leftarrow S^0$；▷ 有效优先关系

$P^0_{t^*} \leftarrow \{i \mid (i,t^*) \in S^0\}$；▷ 的原始前驱任务$t^*$

$S^0_{t^*} \leftarrow \{j \mid (t^*, j) \in S^0\}$；▷ t 的原始后继$t^*$

如果 $b_2 = 0$ **则**

  对所有 $i \in P^0_{t^*}$，从 S' 中移除 (i,t^*);

  对所有 $j \in S^0_{t^*}$，从 S' 中移除 (t^*, j);

  对所有 $i \in P^0_{t^*}, j \in S^0_{t^*}$，将 (i, j) 添加到 S'；

后继任务′ ←构建后继映射(S')；▷ S的后继映射′

$S^{min}_{t^*} \leftarrow$ 最大（$\{0\} \cup \{S_0(i) + p_i \mid i \in P^0_{t^*} \wedge (i,t^*) \in S'\}$）；

如果 $\tau' <$<style id="4">$S^{min}_{t^*}$ **then**

  返回 不可行：τ^* <最早前驱完成时间

$\langle Tf, \tau_f \rangle \leftarrow \langle \{t^*\}, \{t^* \mapsto \tau^*\} \rangle$；▷ 任务集和开始时间映射

$T_{conf} \leftarrow$ CheckTotalConflicts($t^*, \tau^*, p, S^0, u', T, R, C$)；

$T_{flex} \leftarrow$ PropagateFlexibility($S', T_{conf}, T,$后继′)；

如果 $b_1 = 0$ **则**

  对每个 $i \in P^0_{t^*} \setminus Tflex$ **执行** ▷ 对于每个可行候选项

    $\left\langle T'_f, \tau'_f \right\rangle \leftarrow \langle T_f \cup \{i\}, \tau_f \cup \{i \mapsto S_0(i)\} \rangle$；▷ 暂时

    $f_R \leftarrow$ 检查资源可行性(T'_f，τ'_f，u'，p，C，R，H)；

    如果 f_R**则** ▷ 任务i 在资源方面具有可行性

      $\langle Tf, \tau_f \rangle \leftarrow \left\langle T'_f, \tau'_f \right\rangle$；▷ 永久添加任务$i$

$f_{final} \leftarrow$ 检查资源可行性($Tf, \tau_f, u', p, C, R, H$)；

如果 $\neg f_{final}$ **那么**

  return 不可行：最终固定集合 Tf 违反资源约束

返回 $\langle S', u', Tf, \tau_f \rangle$

6.3 重调度执行

整体重新调度流程由调度器机制管理。INTERPRETINTEN- TION（算法 1）确定请求类型 γ 和意图标志（b_1,b_2）后，调度器会选择适当的逻辑流程。具体而言，它会根据特定的意图标志调用 PRE-PROCESS 函数（算法 2）。如果预处理成功并返回一个可行的修改实例（$\langle S', u', Tf, \tau_f \rangle$），调度器就会将其表述为一个包含固定任务的资源受限项目调度问题，并将其交给标准 RCPSP 求解器处理。如果预处理失败，或求解器返回不可行结果，调度器就会报告失败及从预处理或求解阶段得出的原因。否则，求解器返回的最终调度即为输出。这种结构使同一套核心预处理和求解步骤能够处理不同的用户意图组合，区别仅在于传递给 PRE-PROCESS 的参数。

6.4 交互机制的形式化验证

为了通过人机交互建立信任，我们展示了关于意图提取过程终止性及其正确性的形式化保证。这里，我们将步骤 k 时用户意图在域 $\{\top,\bot,\mho\}$ 中的状态定义为向量 $\mathbf{v}_k$。

定理1（收敛）设 $\mathscr{S}$ 表示一组包含 n 个决策符号的集合，这些符号是完整指定请求所必需的。假设用户始终在规定时间内回答系统的询问，则 *InterpretIntention*算法最多经过 n 次交互后终止。

证明 令$U_k = \{s \in \mathscr{S} \mid \text{value}(s) = \mho\}$ 表示第k次迭代时未决符号的集合。由于初始时$U_0 = \mathscr{S}$, $|U_0| = n$。此过程以迭代方式执行。在算法1的 While循环中，系统识别符号$s^* \in U_k$并生成一项询问。当返回回复$a \in \{\top,\bot\}$（或超时）时，ASP求解器推导出谓词 known(s^*,a)。因此， $\mho$ 的默认否定规则不再成立，如下所示：

$$U_{k+1} = U_k \setminus \{s^*\} \tag{39}$$

这意味着 $|U_{k+1}| = |U_k| - 1$。随后，该集合的基数会持续严格递减，直至达到下界 0。当 $|U_k| = 0$时，所有符号的集合都会转为已知状态。意图 的 ASP规则（式 38）要求每个$s \in \mathscr{S}$都处于已知状态。当 $|U_k| = 0$时，该条件成立，随后便推导出`foundIntention`，从而终止循环。 ■

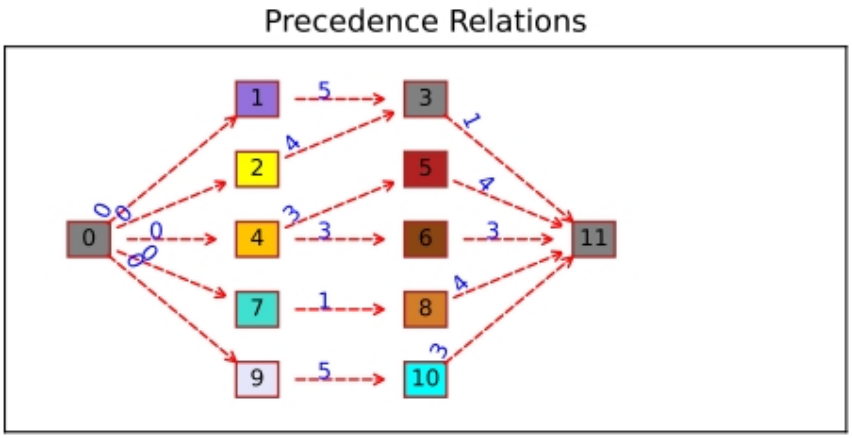
定理2（正确性）系统推导出的意图代码$\mathbf{b}^*$与用户回答序列的合取完全一致。

证明决策树 Δ用于实现布尔函数f： $\{0,1\}^n \to \mathscr{I}$，其中 $\mathscr{I}$是意图标签集合。ASP编码确保观测历史 $\mathscr{H}$与内部状态之间构成双射关系。对于任意输入向量$\mathbf{a}$ =（$a1$，...，a_n），交互循环保证断言一组事实$F = \{\text{known}(i,\ a_i) \mid 1 \le i \le n\}$。意图规则的形式如下：

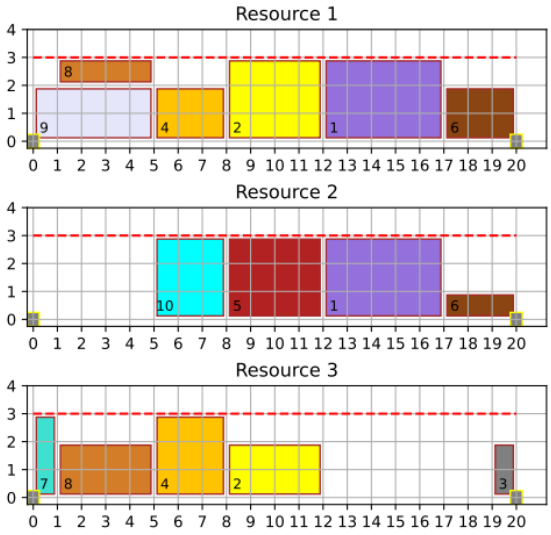
$$\text{intent}(L) \leftarrow \bigwedge_{i=1}^{n} \text{known}(i, v_i) \tag{40}$$

由于稳定模型语义保证当且仅当规则体得到满足时才会推导出相应结论，因此会推导出唯一与$\mathbf{a}$ 对应的意图标签L 。约束矛盾/1 通过阻止冲突状态（例如同时设为 $\top$和 $\bot$）出现，来保证逻辑一致性。 ■

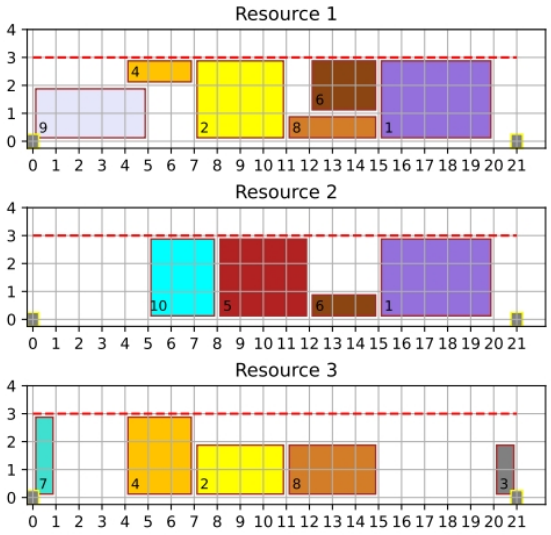
7. 实验与讨论



(a) 优先关系（任务 0 和 11 为虚拟任务）。



(b) 资源分配：基准解。



(c) 资源分配：重新调度解。

图2：任务 4 资源变更请求的基准计划与重新调度计划对比。

7.1 案例研究：任务 4 重新调度

我们使用表1中的实例演示重新调度（图2b展示了基准计划）。用户请求将任务 4 的开始时间改为 $\tau^* = 4$ （基线为5），并将资源 1 的用量降至$R^*_{4,\,1} = 1$ （基线为2）。列表1所示的交互阐明了意图——

tent 作为 γ='r'（以资源为重点）、$b_1 = 1$（允许全局优化）以及 $b_2 = 1$（保持优先关系链接）。

列表1：用于揭示用户意图的交互式询问。

```
结果：请求（4，1，1）        结果（有效_请求）

=====  交互 #00  =====
未知:   [未知 (1)          ，未知 (2)]
查询:     [查询 (1)]

=====  交互 #01  =====
您希望通过影响
其他任务的        开始时间来优化新计划吗？是为 1；否为 0。
?[1，0]: 是
无效答案：是

您希望通过影响
其他任务'         开始时间？是请输入 1；否请输入 0。
?[1，0]:2
无效答案：2

您希望新计划通过影响
其他任务'         开始时间？是请输入 1；否请输入 0。
?[1，0]:1
sym:  [对称 (1," 真 ")]
未知:   [未知 (2)]
查询:     [查询 (2)]

=====  交互 #02  =====
您是否希望该任务在依赖其他任务的同时保持 -
是否包含优先关系？是为 1；否为 0。
?[1，0]:1
sym:  [对称 (1," 真 "),        对称 (2," 真 ")]
意图:      [意图 (操作_11)]
```

在预处理步骤中更新 $u'_{4,1} = 1$ 后，任务 4（$T_{\mathrm{f}} = \{4\}$）被固定在 $\tau_4 = 4$，并在局部维持优先关系（S'）。通过可行性检查后，系统会求解更新后的资源受限项目调度问题实例。如图 2c 所示，任务 4 在 $t = 4$ 开始，并使用 1 单位的资源 1，使总工期变为 21。这表明系统能够生成可行且符合意图的计划。

7.2 讨论

我们的方法支持意图感知重调度，但仍有改进空间。虽然图 2b 和图 2c 等可视化方式有助于与用户沟通，但二元意图标志（b_1,b_2）可能会简化复杂的用户优先级，从而引发可用性问题。

要处理不可行计划，不能只报告其失败，还需要做更多工作。尽管计算成本可能较高，但提出可行替代方案或指出确切的冲突点，对用户而言都具有建设性且有用。

此外，当前的离线方法应扩展至动态环境。为适应滚动时域规划和局部修复等在线重调度技术，有必要对当前框架进行调整。然而，这种扩展需要在意图满足度、可行性和响应性之间取得平衡。

8. 结论

本研究提出了一种意图感知重调度方法，用于处理资源受限项目调度问题环境下城市空中交通垂直起降机场运营中的模糊人类请求。对于提出的解决方案，我们展示了如何将基于答案集编程的三值逻辑意图解释与基于混合整数线性规划的优化相结合。答案集编程可提高透明度、建立信任，并实现可解释的重调度系统。

未来应解决计算可扩展性、深入的意图建模，以及面向实时城市空中交通的在线动态调度问题。必须通过全面的可用性研究，确认其实际可行性和用户认可度。

致谢

本研究由教育部（MOE）资助的区域创新体系与教育（RISE）计划通过全罗南道RISE中心提供支持，资助方为大韩民国全罗南道。（2025-RISE-14-008）

参考文献

[1] J. Kim 和 K. Kim，“智能体3，改变你的航线：人类管理者与城市空中交通空中交通管理（UATM）之间可能的对话”，载于机器人学：科学与系统（*RSS*）研讨会《清晰表达的机器人：利用语言进行机器人学习》，韩国大邱，2023年。[在线]。获取地址：https://arxiv.org/abs/2306.14216[2] ——，“人类监督员与城市空中交通空中交通管理之间的对话可能性：航线变更”，人工智能与机器学习进展（*AAIML*），第3卷，第1352–1368页，2023年8月。[3]S. Woo、J. Kim 和 K. Kim，“我们，垂直起降机场6，暂时关闭：用于更改目的地的交互式本体方法”，载于*IEEE RO-MAN*（*RO-MAN 2023*）自主机器人本体（*RobOntics*）研讨会，韩国釜山，2023年8月。[在线]。获取地址：https://ceur-ws.org/Vol-3595[4] J. Kim 和 K. Kim，以人为因素为考量设计反应式航线变更规则：城市空中交通空中交通管理系统视角，2024年8月，第323–338页。[5]——，“通过声明式推理和利益相关者建模提升城市空中交通调度水平”，《航空航天》，第12卷，第7期，2025年。 [在线]。获取地址：https://www.mdpi.com/2226-4310/12/7/605[6] N. E. Safwat、R. Sabatini、加尔迪、哈菲兹和纽瓦吉，“考虑同信道干扰的城市空中交通通信性能”，《*IEEE*航空航天与电子系统汇刊》，2024年。[7] 王、德拉埃、法尔热和阿拉姆，“高强度城市空中交通运营的空中交通分配”，《航空航天信息系统期刊》，2021年。

[8] J. Li、R. Yang、C. Li、Y. Zhou 和 L. Huang，“城市空中交通垂直起降机场的初步研究”，2023年。[9] Q. Wei、G. Nilsson 和 S. Coogan，“容量约束的城市空中交通调度”，2021年。[10] D. P. Thipphavong、R. D. Apaza、B. Barmore、V. Battiste、B. K. Burian、Q. Dao、M. Feary、S. Go、K. H. Goodrich、J. Homola、H. Idris、P. Kopardekar、J. Lachter、N. Neogi、H. K. Ng、R. M. Oseguera-Lohr、M. D. Patterson 和 S. Verma，“城市空中交通空域融合的概念与考量”，2018年。[11] F. Zaman、S. Elsayed、R. Sarker 和 D. Essam，“不确定资源约束调度问题的基于情景的求解方法”，2018年。[12] A. H. Hosseinian、V. Baradaran 和 M. Bashiri，“考虑学习效应的时变多技能资源受限项目调度问题（RCPSP）建模”，《管理建模期刊》，2019年。[13] J. Gholamnejad、R. Lotfian 和 S. Kasmaeeyazdi，“采用整数线性规划的露天矿实用型长期生产调度模型”，《南非采矿与冶金学会期刊》，2020年。[14] Z. Liu、Z. Liu、M. Liu 和 J. Wang，“采用混合整数线性规划优化预制混凝土构件生产的流水车间调度”，《土木工程进展》，2021年。[15] S. Mouret、I. E. Grossmann 和 P. Pestiaux，“一种用于原油调度问题的新型基于优先级时段的连续时间公式”，《工业与工程化学研究》，2009年。[16] Q. Duan 和 T. W. Liao，“用于确定项目关键路径的改进蚁群优化算法”，《建筑自动化》，2010年。[17] F. Mahmud、F. Zaman、A. Ahrari、R. Sarker 和 D. Essam，“用于求解单一资源约束项目调度问题的遗传算法”，《*IEEE Access*》，2021年。[18] Y. Alipouri，“一种用于求解模糊随机资源约束项目调度问题的资源流分支定界算法”，《软计算》，2021年。[19] M. Davari 和 E. Demeulemeester，“一种用于机会约束资源受限项目调度问题的新型分支定界算法”，《国际生产研究期刊》，2018年。[20] M. A. Nemmich、F. Debbat 和 M. Slimane，“一种用于求解资源受限项目调度问题的基于排列的蜜蜂算法”，《国际群体智能研究期刊》，2019年。[21] Y. Li 和 C. M. Burns，“采用认知工作分析对自动化进行建模，以支持人机协调”，《认知工程与决策期刊》，2017年。[22] P. A. Hancock、R. J. Jagacinski、R. Parasuraman、C. D. Wickens、G. F. Wilson 和 D. Kaber，“人机交互研究”，《人体工程学设计：人为因素应用季刊》，2013年。

[23] T. Han 和 L. Pereira，“意图识别的最新进展及其在决策中的应用”，人工智能通信，第 26 卷，第 2 期，第 237–246 页，2013 年 7 月。[24] A. P. Le、H. N. Tran、T. U. Thi Nguyen 和 D. K. Tran，“基于语言多值逻辑的语言多属性决策方法”，智能信息学进展国际期刊，2022 年。[25] C. Liu，“第一印象偏好”，理论计算机科学电子论文集，2016 年。[26] M. A. Berawi、S. A. Onggar Siahaan、G. Gunawan、P. Miraj 和 P. Leviäkangas，“使用模糊逻辑和决策树方法确定地震灾害中的优先救助对象”，*Evergreen*，2020 年。[27] P. Kordjamshidi、D. Roth 和 K. Kersting，“作为人工智能系统接口的基于声明式学习的编程”，人工智能前沿，2022 年。[28] C. Dodaro、D. Ilardi、L. Oneto 和 F. Ricca，“用于答案集编程中启发式方法生成的深度学习：图着色案例研究”，2022 年。[29] G. Mazzotta、F. Ricca 和 C. Dodaro，“ASP系统中聚合的编译”，*AAAI*人工智能会议论文集，2022 年。[30] F. Calimeri、G. Ianni、F. Pacenza、S. Perri 和 J. Zangari，“通过 DLV2 和增量实例化实现基于 ASP 的多轮次推理”，2022 年。[31] Y.-F. Wang、Y. Chen 和 S. Chien，“公民遵循政府支持的人工智能系统建议的意图”，公共政策与行政，2023 年。[32] S. Kumar 和 S. M. Bargavi，“信任在人机通信与决策中的重要性”，工程与管理国际科学研究期刊，2024 年。[33] A. J. Karran、T. Demazure、A. Hudon、S. Sénécal 和 P. Léger，“以建立信心为设计目标：人工智能决策可视化的影响”，神经科学前沿，2022 年。[34] N. Vemuri，“利用人工智能驱动的人机交互促进工业4.0中的人机协作”，*PST*，2023 年。[35] A. Akula、K. Wang、C. Liu、S. Saba-Sadiya、H. Lu、S. Todorovic、J. Chai 和 S. Zhu，“Cx-tom：结合心智理论的反事实解释，以增强人们对图像识别模型的信任”，*Iscience*，2022 年。[36] J. Kuster 和 D. Jannach，“资源约束项目重调度的高效方法”，载于第三届人工智能研究者新秀研讨会论文集，2006 年 1 月，第 208–219 页。