

CP-SAT 模型驱动的研发台架任务调度优化策略

孙淑军^{1, 2}, 李德明^{1, 2}, 王长通^{1, 2}, 尹世昌^{1, 2}, 朱雪莲^{1, 2}, 赵文耿^{1, 2}

(1. 内燃机与动力系统全国重点实验室, 山东 潍坊 261061; 2. 潍柴动力股份有限公司, 山东 潍坊 261061)

摘要: 本研究针对研发台架环境的特殊性和灵活性, 提出了一种基于 CP-SAT 模型的任务调度优化策略。研究集中于解决无严格顺序约束的研发任务调度问题, 充分考虑了资源限制、任务周期及任务与台架的特定绑定关系。通过精心设计的 CP-SAT 模型, 我们构建了一套适应性强且高效的调度方案, 并通过实际场景验证了该策略在优化研发台架资源分配、提升任务执行效率方面的积极作用。该策略通过深入的建模分析, 展示了在复杂研发台架环境下解决任务调度问题的能力, 特别是在优化资源配置、减少冲突和提高研发效率方面表现突出。本研究为非流水线式研发台架任务调度提供了一种创新的解决方案, 具有广泛的应用前景和理论价值。

关键词: CP-SAT 模型; 研发台架; 任务调度; 资源约束; 任务周期; 优化策略; 研发效率

中图分类号: TP391.4 **文献标志码:** A

1 引言

自主研发是现代企业在激烈竞争环境中维系生存并持续推动产业升级与商业成功的核心驱动力之一。作为一种战略性举措, 自主研发活动的有效调度管理不仅是确保各研发项目按预定时间节点顺利完成的关键, 更是降低项目延误风险、加快创新成果商品化进程、及时捕捉并占领市场的决胜之举。然而, 现实状况表明, 众多企业在实施自主研发任务调度规划时, 仍然很大程度上依赖于传统的人工管理模式, 尤其是在面对研发任务的高度动态性、需求不确定性、周期变异性及资源配置复杂性等固有特征时, 挑战尤为突出。

研发任务因其独特的灵活性与突发性事件频发的特性, 导致其调度问题呈现出极大的复杂度和广阔的解决方案空间。设计一套既能严格遵守各项内外部约束条件, 又能充分挖掘潜在优化潜力, 实现资源利用效率最大化和项目进度全局优化的智能调度策略是一项极具挑战性的课题。为此, 有必要引入科学化、自动化和智能化的调度方法, 结合先进的运筹学理论^[1]与优化算法, 以克服人工调度的局限性, 有效应对日益复杂和紧迫的研发任务调度难题。

2 任务调度算法研究与分析

在任务调度研究领域, 遗传算法 (Genetic Algorithms,

GA)、启发式算法(Heuristic Algorithms)、贪心算法(Greedy Algorithms)及约束规划(Constraint Programming, CP),尤其是其布尔可满足性问题(Boolean Satisfiability Problem, CP-SAT)形式化方法,均在诸多实践场景中发挥着关键作用。随着人工智能技术的飞速发展,深度学习(Deep Learning, DL)和强化学习(Reinforcement Learning, RL)也在逐渐成为解决调度问题的新一代强有力工具。针对研发台架任务调度这一特殊而复杂的环境,本章节依据广泛的学术文献深度剖析了各算法的适用性及局限性。

启发式方法因其快速收敛和易于初步求解的特点,在多种调度场景中表现出了实用性^[2]。然而,鉴于研发任务的高度动态性和灵活性,现有启发式策略在处理此类问题时面临挑战,由于它们普遍依赖经验和近似原则,故易陷入局部最优陷阱^[3],且算法参数难以微调以适应不确定性大、变化频繁的约束条件,导致其通用性减弱。

遗传算法展现了强大的全局寻优潜力,尤其是在探索复杂解空间方面^[4]。然而,GA对诸如种群规模、交叉概率和变异概率等核心参数配置极其敏感,不恰当的设定可能导致搜索效率低下甚至过拟合,加上处理大型、多维研发调度问题所需的高昂计算资源,限制了其在该领域内的有效应用。

贪心算法凭借逐次选取局部最优解的策略在某些情况下展现一定优势,但在研发任务调度的背景下,由于任务间的强依赖性和长远优化诉求,其固有的短视性可能会致使整体调度方案不尽理想,无法确保全局最优^[5]。

深度学习,以其强大的数据处理能力和模式识别特性,在调度领域开辟了新的可能性^[6]。深度学习模型,如CNN和LSTM,能够从大量数据中自动提取特征,提供精准的预测和决策支持。然而,深度学习模型的训练通常需要庞大的数据集,且模型的解释性较差,这在调度问题中可能限制了其应用范围,尤其是在数据稀缺或对决策过程透明度有要求的场景下^[7]。

强化学习,则通过智能体与环境的互动学习,展现了在动态和不确定性环境中的出色适应能力^[8]。RL算法,如Q-learning和DQN,能够通过试错学习,逐步优化调度策略,尤其适用于实时调度和自适应系统。不过,强化学习的训练周期较长,且在处理具有复杂约束条件的优化问题时,可能面临收敛速度慢和稳定性不足的问题。

相比之下,CP-SAT约束规划算法以其严格的约束遵

从性和高效的求解机制脱颖而出。此算法严格遵守所有定义的调度约束,包括任务间的互斥性、优先级规则、固定资源分配及其他复杂的一致性条件,确保所得解集完全合规有效^[9]。CP-SAT尤为擅长处理多约束、组合优化问题,特别是在离散变量决策和复杂逻辑关联条件下,依托约束传播技术显著减少了无效搜索区域,聚焦于潜在可行解集合^[10]。该算法整合了分支定界、回溯、割平面等多种高级智能搜索技术,旨在高效地遍历解决方案空间并借助剪枝等技术加速大规模问题的求解进程。尤为重要的是,CP模型具备较高的动态适应性,能够在实际运营过程中应对新增任务、优先级调整或约束条件变更等不确定性因素,及时进行灵活调整和重新规划^[11]。

近年来,CP模型在多个领域取得了显著的成功案例,有力地证明了其在解决实际问题中的强大潜力。例如,曾清华^[12]等运用CP模型对航空电子系统的任务分配与调度进行了深入研究,成功在短时间内求得了小规模 and 中等规模问题的最优解,以及较大规模问题的高质量解,彰显了算法在复杂任务调度场景下的高效性。邹鑫^[13]等则通过CP模型对重复性项目的离散时间费用权衡问题进行了细致探究,不仅确保了解的最优性,还显著减少了变量和约束的规模,极大提升了求解效率。

综上所述,在深入对比分析各类算法优劣的基础上,本研究坚定地选择了CP-SAT约束规划算法来应对研发台架任务调度难题,不仅因为它在理论上能够确保满足所有预定约束,而且在实践中表现出优越的运算效率和鲁棒性,特别适合解决高度复杂且具有不确定性的研发任务调度场景。后续章节将进一步阐述基于CP-SAT算法的研发台架任务调度模型构建及其在实际案例中的验证,以此证明其在该领域的可靠性和有效性。

3 模型设计与优化迭代策略

本章节聚焦于任务资源预调度模型的设计与优化迭代过程,旨在通过算法模型创新克服复杂约束条件下的效率与优先级平衡难题。我们的目标是通过模型迭代,实现任务调度方案的最优化,特别是在最小化最大完成时间(Makespan)的同时,满足多维度的调度需求。研究之初,我们明确了六项核心调度需求,这些需求指导了模型的全过程:①优先级调度,确保关键任务优先执行;②资源专属性与排他性,每任务仅分配单一台架;③时序与连续性,维护任务逻辑时序;④位置约束,考虑特

定执行地点限制；⑤时间窗口管理，严格控制任务启动时间范围；⑥公平性约束，防止任务过于集中。

3.1 初步模型构建：模型 1- 细粒度调度策略

设共有 T 个任务和 D 个台架，其中 $T \geq D$ ，每个台架上最多可安排 $Shift_j$ 个任务。设决策变量 $x_{i,j,k}$ 为三元变量，其中 $i \in \{1, 2, \dots, T\}$ ，表示第 i 个任务， $j \in \{1, 2, \dots, D\}$ 表示第 j 个台架， $k \in \{1, 2, \dots, Shift_j\}$ 表示第 j 台架上的第 k 个任务。如果任务 i 被分配到台架 j 上的第 k 个任务，则 $x_{i,j,k}=1$ ，否则 $x_{i,j,k}=0$ 。

设 S_i 为任务 i 的开始时间， c_i 为其完成时间， $duration_i$ 为其任务周期， $priority_i$ 为其优先级，其中 $i \in \{1, 2, \dots, T\}$ ，表示第 i 个任务。

目标函数为最小化最大完成时间

$$\text{minimize } Z = \max_{i=1}^T c_i \quad (1)$$

约束条件：

1) 每个任务只可以且必须分配到一个台架的一个任务序号上

$$\sum_{j=1}^D \sum_{k=1}^{Shift_j} x_{i,j,k} = 1, \forall i \in \{1, 2, \dots, T\} \quad (2)$$

2) 同一台架上不同任务执行时不能发生时间冲突

$$s_{j,k+1} \geq c_{j,k}, \forall j \in \{1, 2, \dots, D\}, \forall k \in \{1, 2, \dots, Shift_j - 1\} \quad (3)$$

3) 特殊台架需求约束、任务优先安排约束等其他特殊性约束，会根据具体案例需求的实际特殊约束模型中展示。

经过模型验证表明，尽管理论上强大，但模型 1 在大规模调度中因计算复杂度过高，响应速度缓慢，任务分配不均匀。在此基础上，通过重新设计模型、调整模型参数等，我们设计了一套新的优化迭代模型。

3.2 优化迭代：模型 2- 高效调度框架

设共有 T 个任务和 D 个台架，其中 $T \geq D$ 。设决策变量 $x_{i,j}$ 为二元变量，其中 $i \in \{1, 2, \dots, T\}$ ，表示第 i 个任务， $j \in \{1, 2, \dots, D\}$ 表示第 j 个台架。如果任务 i 被分配到台架 j 上，则 $x_{i,j}=1$ ，否则 $x_{i,j}=0$ 。

设 S_i 为任务 i 的开始时间， c_i 为其完成时间， $duration_i$ 为其任务周期， $priority_i$ 为其优先级， ω_1 为各任务结束时间和值的权重， ω_2 为最大结束时间值的权重。

目标函数为最小化各任务结束时间和与最大完成时间的加权和

$$\text{minimize } Z = \omega_1 \times \sum_{i=1}^T c_i + \omega_2 \times \max_{i=1}^T c_i \quad (4)$$

约束条件：

1) 每个任务只可以且必须分配到一个台架上

$$\sum_{j=1}^D x_{i,j} = 1, \forall i = \{1, 2, \dots, T\} \quad (5)$$

2) 同一台架不同任务执行时不能发生时间冲突，即对于每个台架上的任务集合：

$$s_i + duration_i \leq s_k \text{ or } s_k + duration_k \leq s_i \quad (6)$$

式中， $i, k \in Task_j, Task_j \subseteq \{1, 2, \dots, T\}$ 为台架 j 上所有的任务集合， $i \neq k$ 。

3) 特殊台架需求约束、优先安排需求约束等其他特殊性约束。

3.3 模型求解过程

模型构建完成后，我们进入了求解阶段，利用 OR-Tools 中的 CP-SAT 算法引擎来解算模型。大致步骤如下。

1) 模型导入与转换。首先，将设计好的调度模型通过 OR-Tools 的 API 转化为求解器可识别的格式。这一步骤涉及定义所有决策变量（任务分配变量、设置目标函数，添加所有约束条件等）。

2) 自动约束处理。CP-SAT 算法的核心优势在于其高效的约束传播机制。在求解开始之前，求解器自动进行一系列预处理，包括传播约束条件，通过逻辑推理减少变量的可能取值范围，这一步有助于缩小搜索空间，加速后续的求解过程。

3) 搜索策略。尽管没有手动调整搜索策略，CP-SAT 算法内建有一套默认的智能搜索机制，它会根据当前搜索树的状态动态选择最有潜力的分支进行探索。这包括基于变量选择的启发式方法，如最小剩余值、影响最大的变量等，以及深度优先、宽度优先等探索策略。

4) 解的发现与验证。随着求解过程的进行，求解器尝试为模型找到一组满足所有约束的解。一旦找到可行解，它会立即报告，同时，CP-SAT 算法还会继续搜索是否有更优解，直到达到预设的终止条件。

5) 结果收集与评估。求解结束后，我们从求解器获取解的信息，包括但不限于任务分配详情、最大完成时间、台架分配情况等。这些数据用于评估解的质量，并与预期目标或基准进行比较，以验证模型的有效性和求解过程的成果。

尽管求解过程相对直接，通过 OR-Tools CP-SAT 算法的强大功能，我们能够有效应对复杂调度问题，自动发现并验证满足所有约束的解决方案。

3.4 实验平台与实证分析

本次实验依托于一套较为先进的计算平台，该平台搭载双 Intel Xeon Silver 4210R 处理器，2.4GHz，共 20 核心 40 线程，配备 32GB RAM，存储组合为 500GB SSD 与 2TB HDD，运行 64 位 x64 系统。

CP-SAT 模型的一个显著优势在于其内置的并行计算能力，这为解决大规模调度问题提供了强大动力。我们通过调整模型参数，如 `solver.parameters.num_search_workers = 4`，轻松激活了 4 个并行工作线程，显著加速了求解过程。这种灵活性意味着，无论是在高端服务器还是在更为常见的计算设备上，只需根据实际可用资源和需求，配置相应参数，即可发挥出 CP-SAT 模型的并行优势。认识到并非所有任务都能在任意台架上执行，我们采取了一项创新性的任务分组策略。在模型运行前，我们根据各台架的具体能力和任务对台架及附属设备的需求对任务进行了精细分组，这不仅有效减少了求解过程中的无效计算，还显著降低了问题的规模，进而大幅提升了求解效率。通过以上策略，CP-SAT 模型即使在普通甚至较为基础的计算资源上预期上也能展现出卓越的性能。

在接下来的实际案例中，我们假设已经按照台架能力对任务进行了分组，案例中所有的台架适配所有的任务。

1) 案例 1。本案例具体涉及 5 个任务需分配至 3 个

台架上，相较于基础模型，本案例特别强调了任务优先级（任务 1 优先）、特殊台架需求（任务 3 必须安排在台架 2）等实际调度中的常见约束。目标仍然是最小化最大完成时间，但在实际任务和台架规模下检验模型的有效性。

模型 1 数学表述为

$$\begin{aligned} & \text{minimize} && Z = \max_{i=1}^T c_i \\ & \text{subject to} && \sum_{j=1}^D \sum_{k=1}^{Shift_j} x_{i,j,k} = 1, \forall i \in \{1, 2, \dots, T\} \\ & && s_{j,k+1} \geq c_{j,k}, \forall j \in \{1, 2, \dots, D\}, \forall k \in \{1, 2, \dots, Shift_j - 1\} \\ & && s_2 = 0 \\ & && \sum_{k=1}^{Shift_1} x_{3,1,k} = 1 \end{aligned}$$

模型 2 数学表述为

$$\begin{aligned} & \text{minimize} && Z = \omega_1 \times \sum_{i=1}^T c_i + \omega_2 \times \max_{i=1}^T c_i \\ & \text{subject to} && \sum_{j=1}^D x_{i,j} = 1, \forall i = \{1, 2, \dots, T\} \\ & && s_i + duration_i \leq s_k \text{ 或 } s_k + duration_k \leq s_i \\ & && s_2 = 0 \\ & && x_{3,1} = 1 \end{aligned}$$

根据以上模型进行计算分析后得到的任务排列甘特图如图 1、图 2 所示。



图 1 案例 1—模型 1 求解结果甘特图

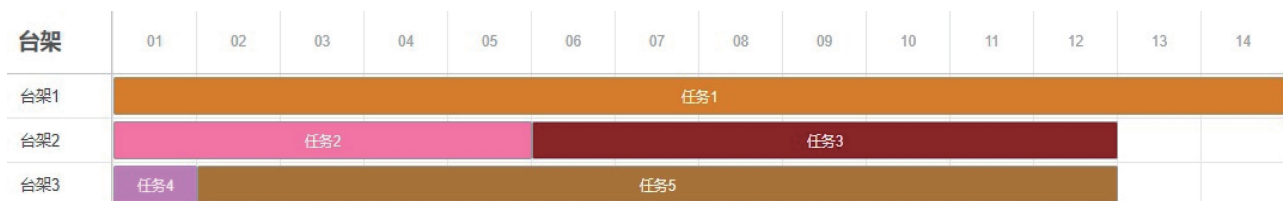


图 2 案例 1—模型 2 求解结果甘特图

根据结果可得，由于目标函数只关注最小化最大完成时间，容易导致某些台架任务过于集中，无法合理利用台架资源，且约束复杂，会造成计算分析耗时过多。

因模型 2 增加了对全体任务完成时间最小化的约束，各台架资源都得到了充分合理的利用。

2) 案例 2。为探究模型在更复杂环境下的性能，案

例 2 将问题规模扩大至 35 个任务和 15 个台架，并加入了更多维度的约束条件，包括但不限于多个任务的优先安排、特定任务到指定台架的硬性要求，以及任务开始时间的限制区间。目标函数进一步发展为最小化各任务结束时间与最大完成时间的加权和，以期达到资源利用与时间效率的更好平衡。具体约束条件：① 每个任务必须且只能分配到一个台架上；② 每个台架上的任务不允许发生冲突；③ 任务 2、3、5 必须优先安排；④ 任务 3、6 必须安排到台架 2 上；⑤ 任务 11、20 必须安排到台架 12 上；⑥ 任务 8 的开始时间不得早于 9 个时间单位；⑦ 任务 9 的开始时间不得晚于 6 个时间单位。

各权重比下模型结构分析表（求解限时 30s）见表 1。

通过对比分析上述不同权重配置的案例，我们观察到权重分配对模型最终解决方案有显著影响，这一发现凸显了在实际应用场景中，平衡各项时间指标的必要性 和挑战。因此，针对特定任务需求定制化的权重配置显得尤为关键。基于全面考量各项性能指标，本研究选定

了一组权重比为 2 : 1 作为优化模型的输入参数。此配置不仅有效控制了总任务结束时间，还兼顾了其他时间敏感性的要求，展现了良好的综合性能。

表 1 各权重比下模型结构分析

编号	权重比		总结束时间 /s	最大结束时间 /s	约束是否全部满足
1	0	1	587	30	是
2	1	0	396	35	是
3	1	1	399	33	是
4	1	2	402	30	是
5	1	4	402	30	是
6	2	1	398	33	是
7	4	1	397	34	是

采用此策略获得的任务分配方案的甘特图如图 3 所示，直观呈现了优化后的任务调度情况，进一步验证了所选权重配置的有效性与实用性。

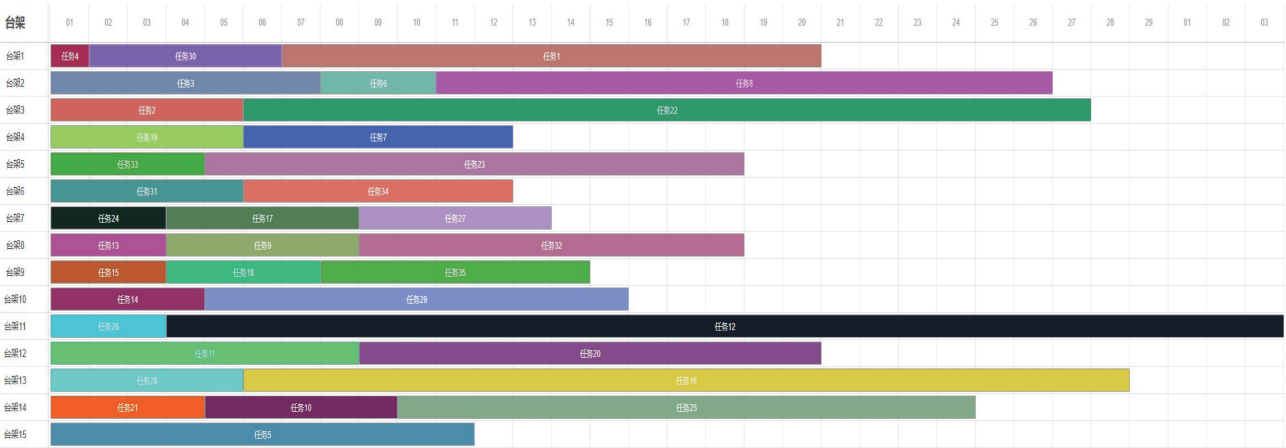


图 3 优化权重配置下的任务分配甘特图

本研究通过实证分析与性能比较，不仅揭示了模型设计中权重配置的关键作用，还展示了通过精细化调整求解参数以适应复杂任务调度场景的能力。模型 2 及其优化策略的成功应用，不仅证明了其在理论层面的先进性，也为工业界提供了强大的工具，有望在提升生产调度效率、降低成本方面产生深远影响。

4 结束语

本文聚焦于任务调度领域中的关键挑战，旨在优化任务完成时间以提升调度效率，同时满足多样的约束条

件。研究通过模型迭代优化，理论探索与实践应用，取得了以下成果。

- 1) 模型创新：提出并评估了两套模型，即详尽调度模型（模型 1）与高效调度模型（模型 2）。模型 1 全面但计算复杂度高，模型 2 通过精简变量显著提升求解效率，展现广泛适用性，尤其在复杂场景中的高效与鲁棒性。
- 2) 性能验证：实证研究表明，模型 2 在各类规模任务集上表现优异，尤其在复杂约束下仍保持高效，验证其实用性和可靠性。
- 3) 理论与实践结合：结合 OR-Tools CP-SAT 求解器，

实现了模型求解的算法细节展示，并通过实践案例凸显其在任务调度中的直接效用，有效缩短排程，提升效率。

4) 研究影响与应用前景：本研究为任务调度提供实用模型，对优化理论与算法设计有启示，尤其是在制造、物流、项目管理及数据中心等领域的应用潜力显著。

5) 未来展望：提出了四个方向的深入探索，包括算法参数优化、混合策略（机器学习与传统调度结合）、动态调度适应实时变化环境的策略，以及跨行业案例验证与模型优化，以增强特定行业适应性。

综上所述，本文通过模型设计与优化迭代，为任务资源预调度提供了有效策略，并指明了未来研究的多个维度，预期在理论与实践应用中取得更广泛进展。IM

参考文献

- [1] 董肇君. 系统工程与运筹学 [M]. 2 版. 北京: 国防工业出版社, 2007.
- [2] 杨燕霞, 伍岳庆, 姚宇, 等. 带时间窗车辆调度问题的启发式算法研究与应用 [J]. 计算机应用, 2013, 33 (S1): 59-61.
- [3] 寿涌毅. 项目调度的数学模型与启发式算法 [M]. 杭州: 浙江大学出版社, 2019.
- [4] 葛继科, 邱玉辉, 吴春明, 等. 遗传算法研究综述 [J]. 计算机应用研究, 2008 (10): 2911-2916.
- [5] 张丹. 基于遗传算法的排课系统 [D]. 长春: 吉林大学, 2009.
- [6] 李冰, 杨鹏, 孙元康, 等. 人工智能文本生成的进展与挑战 (英文) [J]. Frontiers of Information Technology & Electronic Engineering, 2024, 25 (1): 64-84.
- [7] 骆迪, 王宏斌, 蔡峰, 等. 深度学习技术在地震储层预测中的应用及挑战 [J]. 石油地球物理勘探, 2024, 59 (3): 640-651.
- [8] 李明阳, 许可儿, 宋志强, 等. 多智能体强化学习算法研究综述 [J]. 计算机科学与探索, 2024, 18 (8): 1979-1997.
- [9] DECHTER R. Constraint Processing [M]. San Francisco: Morgan Kaufmann Publishers, 2003.
- [10] SCHRIJVER A. The Constraint Satisfaction Problem: Complexity and Algorithms [M]. Berlin: Springer, 2009.
- [11] MARQUES-SILVA J, LYNCE I. Satisfiability, Theory and Practice: From an Abstract Problem to Efficient Algorithms [M]. Cham: Springer International Publishing, 2018.
- [12] 曾清华, 杨志斌, 周勇. 基于约束规划的航空电子系统任务分配与调度方法 [J/OL]. 小型微型计算机系统, 1-11 [2024-10-28]. <http://kns.cnki.net/kcms/detail/21.1106.TP.20230915.1116.022.html>.
- [13] 邹鑫, 王仁锋, 张立辉, 等. 计及软逻辑的重复性项目离散时间费用权衡及其约束规划模型研究 [J]. 中国管理科学, 2022, 30 (10): 109-118.

收稿日期: 2024-06-07