3 基于分层强化学习的弹药保障作业动态调度方法

本章针对持续出动模式下舰载机弹药保障实时调度问题展开研究。首先介绍了本章所涉及的时间窗、分层强化学习、策略网络等相关概念；接着给出了弹药保障动态实时调度问题的定义，阐述了弹药保障动态实时调度框架，对时间窗划分层、规则调度层以及模型训练过程进行了详细描述；最后针对持续出动模式下对模型和算法进行了实验验证和分析。

3.1 本章引言

上一章研究了舰载机弹药静态预调度的相关问题，针对小规模、特定任务下的静态调度场景，设计了基于群体智能优化算法的弹药预调度策略。然而，当场景问题规模较大时，基于群体智能的进化方法会因为大量的迭代搜索导致较大的计算耗时，因此很难扩展到在线、实时优化调度问题中，因此有必要针对弹药保障作业开展动态实时调度研究。

针对弹药动态实时调度问题，学者们也进行了一些研究。强化学习以智能体不断试错的方式在环境中进行探索，从而获得奖励来指导智能体行为，能够很好地从当前执行结果及环境的反馈中学习并不断优化决策质量，适用于解决复杂动态的作业序列决策问题[110]。在原有调度任务事先设定规划方法的基础上增加动态飞行甲板环境下的舰载机保障与调度方法，对调度过程中出现的意外中断实时做出更优的决策，根据作战态势的实时动态性获得更可靠的舰载机弹药保障与调度指导。

尽管上述研究取得了一定成果，但仍然面临以下挑战：1）对弹药保障作业中动态扰动等不确定性情况考虑不足，特别是在动态场景下，出现任务变更或设备故障等特情影响，基于最优化和群体智能优化的算法难以在短时间内给出最优解，使得构建的模型和算法难以满足高动态、强实时的弹药保障作业场景需要；2）现有智能优化调度方法往往存在获取当前保障状态困难、重调度计算效率低、无法通过存储优秀历史经验提升自身调度能力；3）对持续出动模式下大批量弹药保障任务，未明确基于时间窗的任务匹配规则，难以保证“保障任务-保证设备（资源）”匹配的整体质量。

针对以上不足，本文面向航母舰载机弹药保障作业调度场景提出一种端到

端的弹药保障作业实时调度模型，该模型以自适应时间窗的推进方式生成舰载机多波次弹药保障实时调度方案。具体而言，本章的主要工作如下：

（1）根据弹药保障动态场景，将保障过程划分为时间窗划分、弹药转运规则匹配两个阶段，进一步地，将两个阶段建模为马尔科夫决策过程。

（2）基于分层强化学习方法，构建了一种自适应动态弹药保障框架，主要包括时间窗划分层和规则匹配层。针对时间窗划分层，智能体通过动态调整时间窗，根据系统当前状态自适应设置灵活窗口。

（3）针对规则匹配层，智能体基于预定规则，自适应选择最优匹配策略，为弹药选择最适合的启发式匹配规则，从而实现弹药保障的动态优化和资源高效利用实现有限资源的合理分配。

（4）使用 PPO 算法联合分层强化学习框架。

通过时间窗划分层和规则匹配层的协同配合，该框架能够实现资源的精准调度，进一步增强了整体调度框架的适应性和效率。本章所构建的解决方案框架如图3.1所示。（蓝色和粉红框中需要标注一些文字，区分是 XX 层和 XX 层，另外 s,a 这些最好写成动作 a, 状态 s 等）

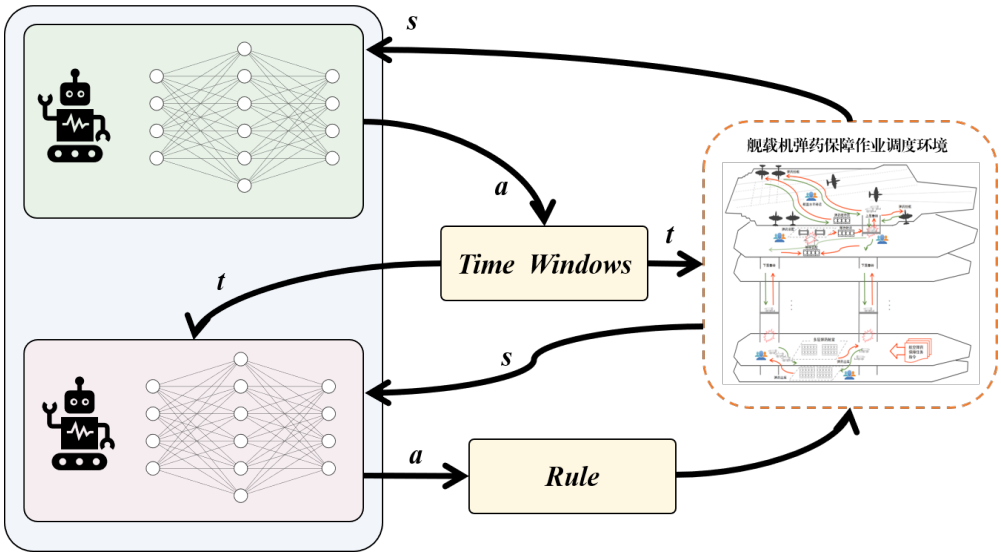


图 3.1 基于分层强化学习的两阶段弹药保障动态调度框架

本章的组织结构如下：第 3.2 节介绍了本算法的相关理论基础，包括分层强化学习和 PPO 网络；第 3.4 节首先概述了本算法的整体模型，接着详细介绍了各个模块的实现过程；第 3.5 节详细描述了本章所涉及的实验设置、对比实验与

消融实验；最后，第 3.6 节总结了本章的研究内容与主要贡献。

3.2 相关概念

本节介绍相关概念，包括分层强化学习、PPO 网络。

3.2.1 分层强化学习

分层强化学习。

3.2.2 策略网络

策略网络是强化学习中的核心组成部分之一，负责直接生成智能体在特定状态下应采取的动作。它通过神经网络或其他函数逼近方法，将环境的状态输入映射到动作空间，输出动作的概率分布或具体的动作选择。策略网络广泛应用于需要连续或离散动作决策的问题中，例如机器人控制、游戏策略优化和资源分配等。在这些场景中，策略网络能够根据当前环境动态生成实时决策，使智能体在复杂的、可能具有不确定性或动态变化的环境中表现出较高的适应性和决策能力。目前还未体现 DQN 的介绍，可以在介绍完策略网络之后，说本章节主要聚焦在 DQN 和 PPO 网络作为我们的策略网络，下面具体介绍下这两种网络：1）DQN：XXXX; 2）PPO：XXXX

近端策略优化算法（Proximal Policy Optimization, PPO）是一种基于策略梯度的强化学习方法，广泛应用于需要处理复杂和高维动作空间的场景。PPO 的核心在于通过约束策略的更新幅度来保证策略优化过程的稳定性。与传统的策略梯度方法相比，PPO 引入了一个剪切损失函数，通过限制策略变化的范围来避免策略更新过快导致的失稳问题。在 PPO 中，策略的更新目标是最大化以下损失函数：

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \tag{3.1}$$

$$L^{\text{CLIP}}(\theta) = \mathbb{E}_t\left[\min\left(r_t(\theta)\hat{A}_t, \text{clip}\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_t\right)\right] \tag{3.2}$$

其中，$\pi_\theta(a_t|s_t)$ 表示新策略的概率分布，$\pi_{\theta_{\text{old}}}(a_t|s_t)$ 表示旧策略的概率分布，$\hat{A}_t$ 为优势函数，用于度量当前动作相对于其他动作的优劣性，ϵ 为超参数，用于控制剪切范围。PPO 通过限制新旧策略之间的变化幅度，使得策略更新更加平稳，避免了较大更新步长可能带来的性能波动。该算法在训练过程中不断调整策略

以最大化累积奖励，并且由于剪切机制的引入，使得 PPO 算法的收敛速度和稳定性得到了显著提升。

PPO 的优化目标还包括联合优化价值函数，其整体损失函数为:

$$L(\theta) = L^{\text{CLIP}}(\theta) - c_1 \cdot L^{\text{VF}}(\theta) + c_2 \cdot H(\pi_\theta), \tag{3.3}$$

其中 $L^{\text{VF}}(\theta)$ 表示价值函数的误差，$H(\pi_\theta)$ 表示策略的熵正则化项，c_1 和 c_2 是权重系数，用于平衡目标之间的关系。通过引入熵正则化，PPO 能够鼓励策略的探索行为，避免陷入局部最优。

3.3 问题分析及定义

3.3.1 问题分析

3.3.2 问题定义

还未完成，摘抄自第三章。还需要修改

定义 1（舰载机弹药） 舰载机弹药表示为一个二元组 $m = (m_t, m_w)$，其中 m_t 为弹药类型，m_w 为弹药重量。

针对不同作战任务配备不同类型的舰载机弹药，弹药转运保障时要满足弹药转运车所允许的弹药数量和载重限制。对于弹药的外形尺寸约束，在建模和仿真实验中将其与弹药重量设定为正相关的映射关系。基于已有的研究工作[12]，大部分类型的舰载机弹药可通过使用不同型号的适配器固定在弹药转运车上，因此仅考虑重量因素即可构造各类型弹药遵守保障车辆最大载重限制以及尺寸大小的约束，从而将复杂问题适当简化便于建模和仿真测试。

定义 2（弹药加工设备） 弹药舱表示为一个二元组 $d = (d_l, \boldsymbol{\Phi})$，其中 d_l 为弹药舱的位置，$\boldsymbol{\Phi}$ 为弹药舱贮存舰载机弹药的类型及数量。

弹药舱分布在航母舰艏和舰舯的不同位置上，作战任务所需的弹药分散贮存在不同的弹药舱内，一个弹药舱可贮存多种类型的舰载机弹药。

定义 3（保障环境） 整个弹药转运保障作业环境定义为一个加权有向图 $\boldsymbol{G} = (\boldsymbol{Z}, \boldsymbol{E}, \boldsymbol{W})$，其中 $z \in \boldsymbol{Z}$ 为弹药舱/弹药升降机/弹药装配台/停机挂弹位等保障节点，$e_{ij} \in \boldsymbol{E}$ 为连接保障节点 i 和 j 的一条有向边，边 e_{ij} 的权值 $w_{ij} \in \boldsymbol{W}$ 由各保障节点之间的位置距离及弹药转运任务优先级组成。:

定义 4（保障环境）

定义 5（问题定义） 最小化时间

3.4 基于分层强化学习的弹药实时调度算法

3.4.1 框架概述

面对舰载机弹药保障作业中高动态性、强实时性以及多约束条件等复杂环境，本文提出了一种基于分层强化学习的自适应动态弹药保障框架（Hierarchical Reinforcement Learning based Adaptive Dynamic Ammunition Support Framework，HRL-ADAS），其具体结构如图 3.1所示。该方法基于分层强化学习范式，构建双层智能网络以优化弹药实时保障过程：上层智能体通过动态调整时间窗，根据系统当前状态自适应设置灵活窗口；下层智能体基于预定规则，自适应选择最优匹配策略，实现有限资源的合理分配。具体而言，上层智能体首先综合分析实时弹药保障态势和弹药保障机床运行状态，精准选择时间窗大小。通过将该时间窗内空闲弹药的决策信息传递至下层智能体，下层智能体将为空闲弹药选择最适合的启发式匹配规则，从而实现弹药保障的动态优化和资源高效利用。（ADAS 框架名字需要和图 4.1，4.4 章节名对应）

基于上述构建的自适应动态弹药保障方法，本章聚焦特勤状况下的弹药保障调度优化，表 3.1详细阐释了方法中涉及的关键符号及其定义。接下来，将系统性地阐述该方法的具体实现路径和技术细节。

表 3.1 符号定义

符号	定义	符号	定义
Q-Learning[124]	离散	值	
DQN[125]	离散	值	

3.4.2 马尔可夫决策建模

针对舰载机弹药保障的复杂动态环境，本文引入分层强化学习策略，将整个弹药保障系统建模为两个马尔可夫决策过程（Hierarchical Markov Decision Process）。具体而言，系统定义为 (MDP^l, MDP^h)，其中高层智能体 $MDP^h = (\mathcal{S}^h, \mathcal{A}^h, \mathcal{R}^h, \mathcal{P}^h, \gamma^h)$，低层智能体 $MDP^l = (\mathcal{S}^l, \mathcal{A}^l, \mathcal{R}^l, \mathcal{P}^l, \gamma^l)$ 均由状态空间 $\mathcal{S}$、动作空间 $\mathcal{A}$、奖励函数 $\mathcal{R}$、状态转移函数 $\mathcal{P}$ 和奖励折扣因子 γ 五元组构成。具体定义如下：

3.4.2.1 时间窗划分层 MDP

$\mathcal{S}^h$（高层状态）：状态 $s_i^h \in \mathcal{S}^h$ 表示第 i 个决策步下高层智能体的全局状态，状态 s_i^h 可以使用三元组来表示，即 $s_i^h = (D, D^w, O)$，其中 D 表示所有弹药的基本状态，使用一组二维向量组表示为 $[[x_1^1, \ldots, x_n^1], \ldots, [x_1^m, \ldots, x_n^m]]$，其中 $, x_n^m$ 表示第 m 层，第 n 类弹药的数量，D^w 表示未进行保障弹药的状态，也使用一组二维向量组表示为 $[[x_1^1, \ldots, x_n^1], \ldots, [x_1^m, \ldots, x_n^m]]$，其中 $, x_n^m$ 表示第 m 层，第 n 类弹药的数量，O 表示所有保障设备的状态，使用一组二维向量组 $[[y_1^1, \ldots, y_k^1], \ldots, [y_1^m, \ldots, y_z^m]]$ 表示，其中 y_k^m 表示第 m 层，第 m 台机器的状态，且 $y_k^m \in \{0, 1\}$，其中 0 表示设备空闲，1 表示设备忙碌。

L 表示各个阶段当前等待调度的车次弹药的数量，R 表示各个阶段空闲的机器数量，A 表示各个阶段下一个车次弹药的到达时间，h 表示距离上次划分时间窗的时间，

$\mathcal{A}^h$（高层动作）：高层动作 $a_i^h \in \mathcal{A}^h$ 表示第 i 个决策步骤中高层智能体的动作，其动作空间表示为 $\mathcal{A}^h = [t_{min}, t_{max}]$，其中 t_{min} 和 t_{max} 分别表示时间窗的下界和上界，高层智能体的动作限定为该时间窗范围内的整数值。

$\mathcal{R}^h$（高层奖励）：高层奖励 $r_i^h \in \mathcal{R}^h$ 用于评估第 i 个决策步中高层动作 a_i^h 对全局效益的影响。该奖励通过聚合低层奖励来量化高层决策的整体效果，体现其对全局目标的优化能力。本文设计了一种独特的奖励层次机制，，该机制融合下层智能体信息，具体设计将在后文详细阐述。

$\mathcal{P}^h$（状态转移函数）：$\mathcal{P}(S_{t+1}|S_t, A_t)$ 描述了在状态 S_t 下执行动作 A_t 后转移到下一个状态 S_{t+1} 的概率分布。

γ^h（折扣因子）：折扣因子 $\gamma \in [0, 1]$ 表示智能体对未来奖励的关注程度。

3.4.2.2 规则匹配层 MDP

$\mathcal{S}^l$（低层状态）：状态 $s_i^l \in \mathcal{S}^l$ 表示第 i 个决策步下低层智能体的状态，状态 s_i^h 可以使用四元组来表示，即 $s_i^l = (D, D, O, a^h)$。其中，D, D, O 和高层状态中的设置相同，a^h 表示高层智能体的在示第 i 个决策步下的动作。

$\mathcal{A}^l$（低层动作）：低层动作 $a_i^l \in \mathcal{A}^l$ 表示第 i 个决策步骤中低层智能体的动作，具体为选择某种启发式规则进行匹配，

$\mathcal{R}^l$（低层奖励）：低层奖励 $r^l \in \mathcal{R}^l$ 用于衡量第 i 个决策步骤中低层智能体动作 a_i^l 的效果，其计算方法为上个决策步中的最长完成时间减去当前最长完整

完成时间：

$$r_i^l = max_j(C_j)_{i-1} - max_j(C_j)_i \tag{3.4}$$

$\mathcal{P}^l$（状态转移函数）：$\mathcal{P}(S_{t+1}|S_t, A_t)$ 描述了在状态 S_t 下执行动作 A_t 后转移到下一个状态 S_{t+1} 的概率分布。

γ^l（折扣因子）：折扣因子 $\gamma \in [0, 1]$ 表示智能体对未来奖励的关注程度。

3.4.3 时间窗划分层

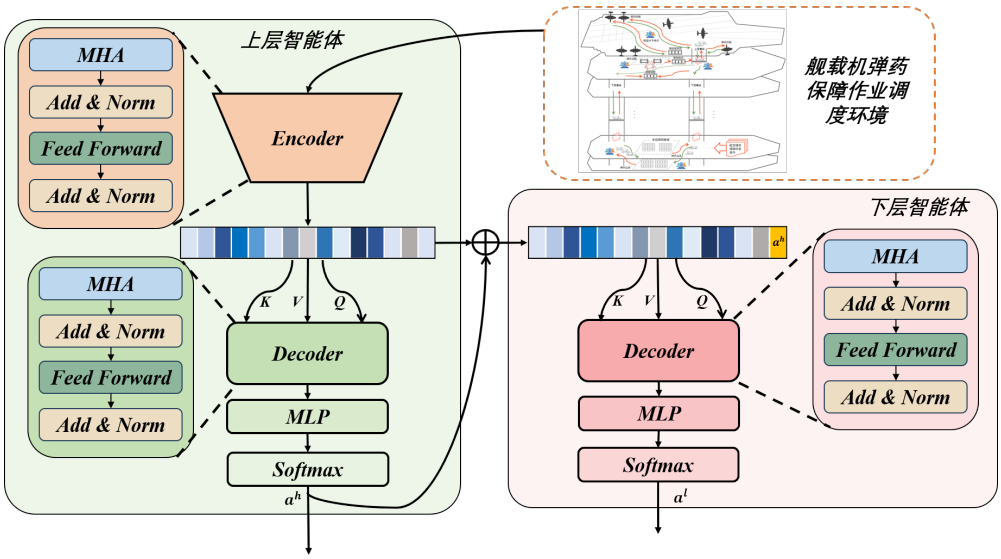


图 3.2 网络结构

本小节主要讲述上层智能体的主要构建方式，如图3.2所示，上层智能体参考 Transformer，由编码器和解码器的的组件构成，其中，编码器的主要功能是处理弹药保障信息，通过对舰载机弹药种类、数量以及保障状态的综合分析，将这些原始数据转化为可用于后续计算的隐变量嵌入（embedding）。解码器的主要功能是根据输入的隐变量嵌入（embedding），通过对其进行处理和解码，输出相应的时间窗大小。通过这种方式，解码器则负责根据编码器生成的隐变量嵌入，对其进行进一步处理和解码，最终输出与任务时间窗大小相关的决策信息，以优化调度过程。编码器和解码器在结构上具有较高的一致性，主要由多头注意力机制（Multi-Head Attention, MHA），残差连接及归一化（Add & Norm），前馈神经网络（Feed Forward）组成，我们以编码器为示例介绍嵌入传递过程，编码器由 L_1 个相同的注意力层的堆叠组成, 每一个注意力层由多头注意力层和前

馈神经网络层 (Feed Forward) 组成, 每个操作后都会进行残差连接和归一化 (add & norm), 嵌入在模型中更新遵循两步过程。

首先是多头注意力机制嵌入。更新输入的嵌入 $x_i^{\ell-1}$ 通过 MHA 提取上下文关系，输出结果与原始嵌入相加后，经过归一化操作得到中间结构 $\hat{x}_i$。通过对所有输入嵌入建模，MHA 能够高效捕捉当前弹药转运作业环境的状态属性的交互关系：

$$\hat{x}_i = BN^{\ell}(x_i^{\ell-1} + MHA_i^{\ell}(x_1^{\ell-1}, x_2^{\ell-1}, \ldots, x_{|p|}^{\ell-}1)) \tag{3.5}$$

接着是前馈神经网络嵌入更新，上一步得到的中间结果 $\hat{x}_i$，通过前馈神经网络层进一步提取高维特征，与 $\hat{x}_i$ 相加并归一化后得到最终输出 x_i^{ℓ}，进一步丰富其语义信息：

$$x_i^{\ell} = BN^{\ell}(\widehat{x_i} + FF^{\ell}(\hat{x}_i)) \tag{3.6}$$

上述式子中的核心是多头注意力层, 其具体过程如下。编码器中的 MHA 将输入嵌入 x_i 通过权重矩阵生成查询（Query）、键（Key）和值（Value）向量：

$$Q_i^h, K_i^h, V_i^h = W_Q^h \cdot x_i, W_K^h \cdot x_i, W_V^h \cdot x_i \tag{3.7}$$

Q 和 K 向量通过点积计算相似性，并结合 V 向量生成加权输出

$$A_i^h = softmax(\frac{Q_i^h \cdot K_i^h}{\sqrt{d_k}}) \cdot V_i^h \tag{3.8}$$

最终，所有注意力头的输出拼接后通过线性变换得到多头注意力的输出：

$$MHA_i = concat(A_i^1, A_i^2, \ldots) \cdot W_o \tag{3.9}$$

其中 $h = 1, 2, 3, \ldots, H$, H 是 MHA 中注意力的总头数, W_Q^h, W_K^h, W_V^h 是 Q, K, V 的权重, W_o 是 MHA 的输出权重, d_k 是单头的特征维度，可以通过以下公式得出: $d_k = d_x/H$.

上层智能体解码器的任务是将编码器生成的隐变量嵌入转化为时间窗划分的具体决策，其结构和编码器类似，由 MHA、FF、残差连接和归一化组成。解码器在处理嵌入时，采用与编码器类似的公式更新嵌入表示，并通过 MLP 层和

softmax 层输出各时间窗选择的概率：

$$TimeWindows = argmax(softmax(MLP(x_i))) \tag{3.10}$$

最终通过编码器和解码器的协作实现时间窗划分任务的高效建模。编码器负责提取弹药保障信息的上下文特征，解码器根据提取的信息生成时间窗划分的最优决策。结合多头注意力机制的高效建模能力，上层智能体为弹药保障调度提供了智能化的时间窗划分方案，显著提升了任务的执行效率和响应能力。

3.4.4 规则匹配层

3.4.4.1 网络结构

规则调度层和时间窗层有相同的网络结构，如图3.2所示。

3.4.4.2 调度规则

在规则调度层智能体中，动作空间由一组预定义的调度规则组成，每条规则对应一种调度策略。规则调度层智能体通过选择这些规则，动态调整任务的执行顺序，以优化系统性能指标。具体而言，我们定义以下几种常用的调度规则作为动作空间的候选项：

1. 最短加工时间优先（Shortest Processing Time, SPT）优先选择加工时间最短的任务进行调度。

2. 最早到期时间优先（Earliest Due Date, EDD）优先选择任务到期时间最早的任务进行处理。

3. 最大剩余工作量优先（Maximum Remaining Work, MRW）优先选择剩余工作量最大的任务进行调度。

4. 最小开始时间优先（Minimum Start Time, MST）优先选择可以最早开始的任务。

3.4.5 模型训练

3.4.5.1 奖励链接

本文对上下两层智能体构建了独特的奖励链接机制，其中下层奖励为上个决策步中的最长完成时间减去当前最长完整完成时间，即

$$r_i^l = max_j(C_j)_{i-1} - max_j(C_j)_i \tag{3.11}$$

根据下层智能体选择各个调度规则的可能性对最长完成时间进行加权求和，

Algorithm 1 HRL-ADAS 算法

Input: Environment E, policy network π_θ, value function network V_ϕ, learning rate α, clipping parameter ϵ, discount factor γ, number of epochs K, batch size B, number of timesteps per update T
Output: Optimal scheduling scheme
Initialize policy network parameters θ, value function parameters ϕ
for each iteration $i = 1, 2, \ldots$ **do**
    Initialize empty storage for trajectories: $\mathcal{D} = \emptyset$
    for each episode $j = 1, 2, \ldots, N$ **do**
        Reset environment: $s_0 = E.reset()$
        for each timestep $t = 0, 1, 2, \ldots, T$ **do**
            Sample action $a_t \sim \pi_\theta(a_t|s_t)$
            Execute action a_t in the environment: $s_{t+1}, r_t = E.step(a_t)$
            Store transition: $\mathcal{D} \leftarrow \mathcal{D} \cup (s_t, a_t, r_t, s_{t+1})$
            if done or max timesteps reached **then**
                Break
            end if
        end for
    end for
    Compute advantages $\hat{A}_t$ and returns R_t for each trajectory in $\mathcal{D}$
    Normalize advantages: $\hat{A}_t \leftarrow \frac{\hat{A}_t - \mu}{\sigma}$
    for each epoch $k = 1, 2, \ldots, K$ **do**
        Shuffle $\mathcal{D}$ into mini-batches
        for each mini-batch $b \in \mathcal{D}$ **do**
            Compute the ratio according to Equation (3.1)
            Compute surrogate loss according to Equation (3.2)
            Compute value loss: $L^{VF}(\theta) = \mathbb{E}_t \left[(V_\phi(s_t) - R_t)^2 \right]$
            Update the networks: $\theta \leftarrow \theta - \alpha \nabla_\theta L(\theta, \phi)$
            Optionally update value function network: $\phi \leftarrow \phi - \alpha \nabla_\phi L^{VF}(\phi)$
        end for
    end for
end for

作为上层智能体的奖励，计算方式如下：

$$r_i^h = \sum p(a_j^l) \cdot [max_j(C_j)_{i-1} - max_j(C_j)_i] \tag{3.12}$$

3.4.5.2 模型训练

针对本文提出的分层强化学习框架（框架名称），本小节介绍改框架的训练方法，本文以 PPO 算法为例，阐述（框架名称）的训练过程（没有体现出上下层的交互训练，即联合这个词）

HRL-ADAS 算法的伪代码如算法1所示。

3.5 实验评估与分析

为验证本文所提出调度方法的有效性与鲁棒性，本文在弹药调度环境下进行了大量实验评估，并与多种具代表性的基线算法进行了系统对比。实验主要从调度效率、资源利用率、任务完成时间和算法稳定性等维度进行评估，进一步分析各类方法在静态与动态场景下的表现差异。本方法的代码详见于https://github.com/

3.5.1 对比算法

为全面评估所提出方法的性能，本节选择了多种具有代表性的对比算法，其中包括贪婪算法、启发式算法、时间窗自适应算法、规则自适应选择算法以及遗传算法。

(1) 贪婪算法（Greedy Algorithm，GA）：本方法采用基于贪婪策略的调度机制。每个待调度弹药单元依次选择当前执行时间最短的工作机床，以最小化局部决策时间，从而提升整体保障效率。

(2) 启发式算法（Heuristic-Based Scheduling，HBS）：启发式算法结合任务与资源的先验知识，采用人工设定的优先级函数或规则（如最短加工时间优先、最少剩余任务优先等）进行调度决策。该类方法在部分特定场景中表现稳定，但其通用性与鲁棒性受限，依赖于启发函数设计质量。

(3) 时间窗自适应算法（Time-Window Adaptive Matching，TWAM）：该方法在贪婪算法的基础上引入时间窗机制，通过智能体自动选择合适的匹配窗口长度，在窗口内对资源进行动态匹配，以更好地适应任务密度和资源状态的变化。该算法在权衡调度粒度与计算复杂度方面具备较强灵活性，适用于动态调度环境。

(4) 规则自适应选择算法（Rule-Based Adaptive Selection，RBAS）：该方法基于深度强化学习，通过智能体选择匹配策略，从而实现调度优化。

(5) 遗传算法（Hybrid Genetic Algorithm, HGA）：基于经典遗传算法，本文采用一种混合进化机制，引入邻域搜索与交叉变异算子优化策略。通过对解空间的深入探索与局部精细优化，HGA 在求弹药调度中表现出更快的收敛速度与更优的调度质量，优于传统遗传算法在大规模任务场景下的表现。

3.5.2 实验环境

为确保实验结果的可信度与代表性，本文构建了一个舰载机弹药转运作业仿真环境，用以模拟多任务、多资源受限、调度冲突频发的典型军用场景。仿真系统支持任务动态注入、资源状态变更等复杂情境，以满足对调度方法通用性与鲁棒性的评估需求。

本节基于参考文献中的弹药转运作业环境，构建实验仿真环境。系统涉及?种类型的弹药，每种弹药均需经过?个保障工序，对应类保障节点，各类节点的数量详见表。?个保障工序的执行时间列于表。

表 3.2

保障节点	数量
下层升降机	4
弹药装配台	8
上层升降机	4
弹药转运小组	6

3.5.3 实验设置

实验设置覆盖以下两类场景：**标准作业调度场景**（对应静态任务模式）：模拟舰载机处于正常出动状态，弹药种类与数量已知，调度目标为在最短时间内完成全量弹药装配与转运任务。**突发特情响应场景**：模拟典型任务中突发的异常情况，为进一步评估调度算法在不确定性条件下的稳定性与响应能力，本文设计了三类典型的突发特情场景：

- **机床故障场景**：在任务执行过程中，随机指定一台关键机床发生故障并退出运行，系统需重新分配任务，考验算法的任务重调度能力。
- **新增保障任务场景**：调度过程进行中动态注入若干新弹药任务，模拟战术调整或新增任务需求情形，重点评估算法的在线调度性能与响应延迟。
- **高复杂度混合特情场景**：综合上述两种情况，模拟多个设备故障、任务冲突与动态变化同时发生的复杂场景，全面考察算法的鲁棒性与适应性。

表 3.3 Caption

表 3.4 Caption

上述特情设置使得实验更贴近真实舰载作业中的不确定性与高动态特性，能够更全面地检验所提方法的工程可用性。

本节所用算法和实验均基于 python3.8 实现，所有实验均在 Intel(R) Core(TM)i9-13900K CPU，内存 64GB，NVIDIA GeForce RTX4080SUPER GPU，Windows 10 操作系统的机器上运行。

3.5.4 评估指标说明

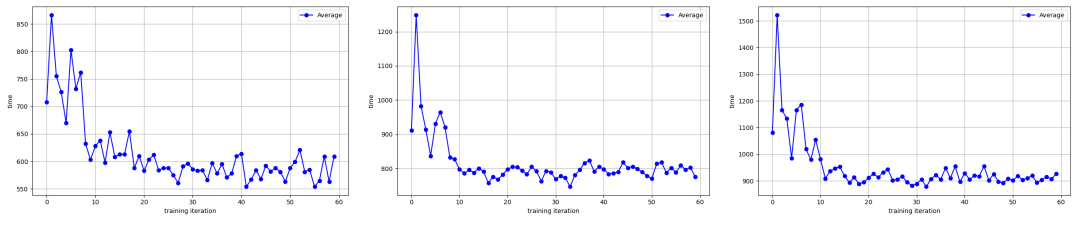
为全面评估各类调度算法的性能表现，本文选取以下两个关键指标进行量化对比：

- **最大完工时间（Makespan）**：该指标定义为所有弹药转运任务中最晚完成时间，反映了整个保障流程的总时长。作为本章优化目标，最大完工时间越小，说明调度方案整体效率越高，任务执行更加紧凑有序。
- **算法执行时间（Time）**：指调度算法在为单次弹药转运任务生成调度方案过程中的运行耗时，用于衡量算法的计算复杂度与实际部署时的响应能力。该指标越小，表示算法在相同算力资源下具有更高的运行效率。

上述两个指标分别从调度效果与计算开销两个维度出发，综合反映了不同方法在性能与实用性上的差异。

3.5.5 模型训练

用三组不同规模的测试训练的模型



(a) 小规模 (b) 中规模 (c) 大规模

图 3.3 training