arXiv:2304.08769v1 [cs.LG] 18 Apr 2023

面向库存管理的协作式多智能体强化学习

Madhav Khirwar[1],Karthik S. Gurumoorthy[1], Ankit Ajit Jain[1], 和 Shantala Manchenahally[1]

[1]Walmart Global Tech，班加罗尔，印度{ madhav.khirwar, karthik.gurumoorthy, ankit.ajit.jain, shantala.manchenahally}@walmart.com

摘要

由于库存管理（IM）中的强化学习（RL）仍是一个新兴研究领域，现有方法往往局限于简单的线性环境，其实现也只是对现成强化学习算法进行少量修改。要将这些简化的环境扩展到现实世界的供应链，仍面临一些挑战，例如：降低环境的计算需求；设定能够反映现实世界门店和仓库动态的智能体配置；以及制定能够鼓励整个供应链实现理想行为的奖励框架。在这项工作中，我们提出了一个系统，其中包含一个自定义的 GPU 并行化环境，该环境由一个仓库和多个门店组成；一种融入增强型状态和动作空间的全新智能体—环境动态架构；以及一套旨在优化大型零售商供应链需求的共享奖励规范。供应链图中的每个顶点都是一个独立智能体，可根据自身库存向上游顶点下达补货订单。仓库智能体除了向供应商下单之外，还具有一项特殊能力：能够限制向下游门店补货，因此需要学习一个额外的分配子策略。对于单一产品，我们构建的系统优于基库存策略等标准库存控制策略以及其他基于强化学习的设定，并提出了面向多种产品的未来研究方向。

关键词——多智能体强化学习、共享奖励、库存管理、分配策略。

1 引言

库存管理（IM）是监督和控制货物流动的过程，涵盖从采购点到销售点的整个流程。库存管理的目标是在最大限度降低运营成本的同时，确保组织能够以正确的数量，在正确的时间和正确的地点提供正确的产品，以满足客户需求。一家大型零售企业可能由数百家分布在广阔地理区域内的门店组成，每家门店均提供数千种产品。对于每家门店而言，这些产品的库存由供应链拓扑结构中与其对应的仓库供应（参见图 1a）。仓库则从专属供应商处获得补货。补货订单由拖车定期配送。门店和仓库负责维持足够的产品库存，以应对补货延迟数个时间段这一意外情况。

库存管理涉及多种权衡，包括维持门店的库存水平，以及尽量降低与持有库存相关的成本。库存管理中的一些关键概念包括：缺货、持有成本和提前期。如果门店或仓库中某种特定产品的库存降至 0，就会发生缺货。这种情况并不可取，因为它不仅会导致销售损失，还会造成糟糕的客户

体验。门店和仓库需要承担持有成本，以维持产品的现有库存。这些成本包括门店的电力成本、食品的冷藏成本以及储存区域维护成本。由于这些成本会随现有库存量增加而上升，因此门店或仓库将现有库存维持在可能的最高水平并非最优做法。提前期是指供应商或仓库将补货订单分别运送至仓库或门店所需的时间。

在这项工作中，我们证明了所提出的系统能够在模拟供应链中成功管理单一产品的库存，而该供应链仿照了现实供应链的复杂性。我们通过纳入 10 种产品的结果，探讨了产品规模扩展，并提出了未来的研究方向，以应对同时管理数千种产品库存所面临的挑战。本文将库存管理问题定义为：已知每种产品在每家门店的需求分布，求出每种产品在每家门店和仓库的最优补货数量，使系统在指定数量的时间段内实现系统整体利润最大化（等价于成本最小化）。

1.1 库存管理的强化学习

以往文献论证了，相较于传统优化方法，基于强化学习的方法更适合构建库存管理解决方案，因为强化学习系统能够考虑未来的长期发展轨迹，而日常启发式优化系统则不具备这种能力 [7, 23, 18, 16]。事实上，这使强化学习系统无需额外的需求预测模型即可运行，因为它们会在训练过程中隐式地学习预测客户需求。我们的实验展示了这一特性：强化学习智能体学习到的库存策略优于基库存策略（BSP）等优化方法 [1], ，且无需额外的需求预测模型。

多智能体强化学习（MARL）是一种利用多个智能体进行学习和适应的机器学习框架。与基于启发式的优化方法相比，MARL 可能是供应链库存管理更好的选择，因为这类优化方法通常假设静态供应链属性，例如需求恒定或采购提前期固定。在供应链中，产品需求可能会波动，供应商延误和运输问题也会影响提前期。MARL 算法能够在实时环境变化中学习和适应，而无需依赖这些假设（在使用历史数据训练时） [22]。因此，它们能更好地应对供应链的不确定性和波动性。与传统优化方法不同，MARL 算法可以采用去中心化方式实施，由每个供应链智能体根据局部信息做出决策。这提高了供应链灵活性，因为供应链拓扑结构的细微变化无需整个系统从头重新训练。

1.2 贡献

本文的主要贡献包括：(i) 提出一种用于库存管理的新型多智能体架构，其中仓库智能体拥有扩展的状态和动作空间，使其能够有效学习分配策略，尤其是在库存不足以满足所有门店需求时；(ii) 提出一种新型奖励设定，以促进系统整体协作，使供应链中的所有智能体共享相同的奖励，该奖励根据整个供应链的动态情况按每个时间段计算；(iii) 提出一种经 CuPy 并行化的环境，可在恒定时间内处理所有产品，但受图形处理器限制。下文将这种结合了增强型仓库、共享奖励和多智能体强化学习的系统称为协作式多智能体强化学习（CMARL）。

2 相关工作

强化学习技术将库存管理问题建模为马尔可夫决策过程，其中状态空间为当前库存水平，动作空间则为每个时间步针对每件待订购商品确定的补货量。近年来，这种方法在小规模概念验证环境中日益流行。例如，[16] 将深度 Q 网络（DQN）扩展至解决分散式啤酒游戏问题

问题 [2] ，并发现，当供应链中的其他参与者采用基库存策略时，DQN智能体能够学习到近乎最优的策略。对于单一产品， [4]研究了一种基于半马尔可夫平均奖励的方法，用于管理供应链中的库存决策。 [8] 提出使用Q学习 [10] ，以达到最小化运营成本的目标，并将其用于管理单一产品的库存，且将结果与基于库存量和基于货龄的策略进行了比较。对于线性四级供应链中的单一产品， [7] 的作者比较了不同的运筹学方法与近端策略优化（PPO）算法在有积压订单和无积压订单环境中的表现。他们的实验表明，在这两种环境中，PPO的表现都优于基库存策略。 [18]讨论了如何应用普通策略梯度（VPG）算法，处理具有随机性和季节性需求的两级供应链。产品的发货量根据仓库中的库存确定。在所有实验中，VPG的表现均优于作为基线的（s, Q）策略。 [22] 的作者讨论了一种单智能体强化学习（SARL）方法，用于优化两级供应链中的库存流。他们将神经动态策略与订货至策略进行比较，并将PPO与VPG和A3C算法进行基准比较，最终得出结论：PPO在库存管理方面表现最佳。

与我们提出的 CMARL 最为接近、并为包含 3 家门店和 1 个仓库的两级供应链实现了库存管理多智能体强化学习系统的工作是 [23]。然而，两种方法在强化学习智能体的环境动态、动作空间、奖励结构和训练结构方面存在根本差异。 [23] 假设仓库与门店之间的提前期为零，但这并不现实，因为订单需要运输和处理；而我们的 CMARL 则在环境动态中明确编码了提前期这一概念。与 [23], 一样，所有产品共用单一动作策略且共享全部权重参数，会导致训练时间随产品数量线性增长。由于灾难性遗忘，这可能降低多种产品的整体性能 [5]。我们的实验结果（参见图 4b）表明，为每种产品设置独立的动作空间能够获得更高的奖励值，同时还能对所有产品进行同步训练。与 [23] 不同，CMARL 中的仓库智能体和门店智能体都共享系统级合作奖励；我们证明这种方式更优，因为它避免了为了局部奖励而牺牲供应链整体的全局最优智能体策略，从而避免智能体之间相互竞争。

在 CMARL 中，我们进一步增强了仓库智能体，扩展其观测空间，使其也包含过去的门店动作；同时扩展其动作空间，使其能够在库存有限、无法满足所有门店请求时，明确学习子分配策略。这样一来，所有智能体都可以同时接受训练，这与 [23], 不同： [23],采用分阶段训练方法，并假设仓库容量不受限制，因此仓库始终备有所有产品所需的补货数量；而在第一阶段， [23],只训练门店智能体。待每家门店都收敛到局部最优策略后，才将仓库的货架容量设为有限，并以已收敛的门店智能体为条件训练仓库。然而，由于门店智能体的局部最优行为未必能使整个供应链达到全局最优，这种分阶段训练流程会导致仓库智能体以一组次优策略为条件进行训练。此外，在 [23], 中，仓库智能体虽然可以获取门店智能体当前时间段的补货请求，却无法获取过去的补货请求。由于缺少这些信息，仓库只能对是否向门店补货作出二元决策，无法像我们的系统那样，智能地将有限库存分配给各门店。我们的 CMARL 系统能够成功模拟发散型供应链，超越了大多数其他运筹学文献所描述的线性供应链。

3 库存管理的多智能体强化学习

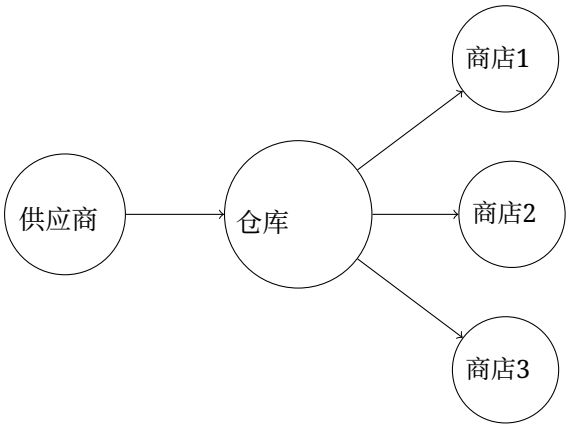
马尔可夫决策过程（MDP）定义为一个元组 $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma)$，其中 $\mathcal{S}$ 是状态空间， $\mathcal{A}$ 是动作空间， $\mathcal{T}$ 是状态之间转移概率的集合， $\mathcal{R}$ 是与每个状态相关的奖励集合， γ 是折扣因子。在每个时间步，MDP 都可以由其状态$s \in \mathcal{S}$完整描述，智能体利用该状态选择动作 $a \in \mathcal{A}$。根据转移概率集合，MDP 将在下一个时间段到达一个新状态 $s' \in \mathcal{S}$： $\mathcal{T}(s' \in \mathcal{S} | s \in \mathcal{S}, a \in \mathcal{A}) : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$。一个强化学习

智能体学习一种随机策略 π，该策略规定了在状态s下可采取的每个动作 a 的概率，即 $\pi(a|s): \mathcal{S} \times \mathcal{A} \to [0,1]$。

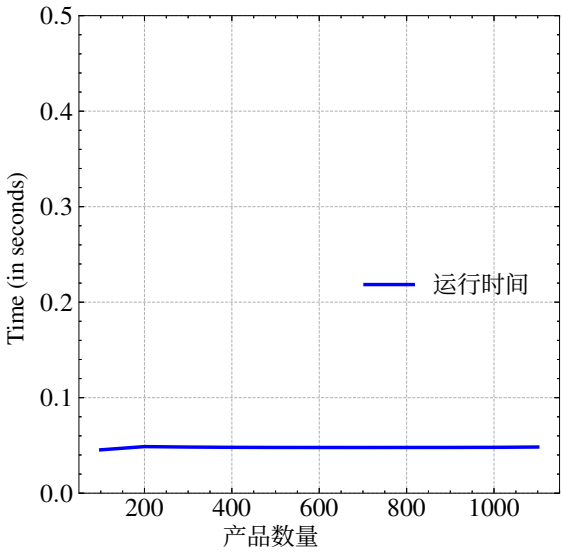
我们将 MDP 推广到多智能体设置，定义为元组 ($\mathcal{U}$, S, A, T, R γ)。其中，$\mathcal{U} = \{u|1 \le u \le |\mathcal{U}|\}$ 是环境中的智能体集合，S 是所有智能体状态空间的笛卡尔积 $u \in \mathcal{U}$: S $= \mathcal{S}_1 \times \mathcal{S}_2 \times \cdots \times \mathcal{S}_{|\mathcal{U}|}$，A 是所有智能体动作空间的笛卡尔积 $u \in \mathcal{V}$: A $= \mathcal{A}_1 \times \mathcal{A}_2 \times \cdots \times \mathcal{A}_{|\mathcal{U}|}$，T 表示 S 和 A 之间的转移概率：T(s′ ∈ S|s ∈ S, a ∈ A)，R 是各智能体奖励函数的笛卡尔积：R $= \mathcal{R}_1 \times \mathcal{R}_2 \times \cdots \times \mathcal{R}_{|\mathcal{U}|}$，而 γ是一个标量折扣因子。在每个时间步 t，所有智能体同步采取动作 a ∈ A。环境中每个智能体的目标是通过找到自身的最优策略 $\pi_u: \mathcal{S}_u \to \mathcal{A}_u$ [24]，使其长期奖励最大化。

3.1 多智能体强化学习实现

在我们的供应链中，$\mathcal{U} = \mathcal{V} \cup \{wh\}$ 是由各个顶点组成的集合，每个顶点要么是仓库智能体（wh），要么是商店智能体 $v \in \mathcal{V}$，如图1a所示。尽管这条供应链中存在层级结构，但所有智能体都是同步执行的。每个智能体都有自己的策略、状态空间和动作空间，且任何智能体都无法直接访问其他智能体的状态空间。



(a) 供应链拓扑结构示例。



(b) 最多 1000 个产品的步进时间。

图1：环境动态。

3.2 门店智能体

令向量 $\mathbf{x}_v(t) = [x_v(t,1) \ldots, x_v(t,K)]$ 表示 K 种产品的现有库存，令 $\mathbf{r}_v(t) = [r_v(t,1) \ldots, r_v(t,K)]$ 表示时间 t 商店智能体 $v \in \mathcal{V}$接受的补货数量（定义如下）。其状态空间定义为：$\mathcal{S}_v = \{\mathbf{x}_v(t)\} \cup \{\mathbf{r}_v(t-i)\}_{i=1}^{l_v}$，其中 l_v 是将产品从仓库运送到门店 v所需的提前期。由于智能体 v 在时间段 t 采取的动作只会在经过相应提前期 l_v 后的时间段 $t + l_v$ 影响奖励，因此，我们通过在观测空间中纳入过去补货订单的历史记录来考虑这种延迟，从而保持马尔可夫性质 [12]（如第 3 节所述）。因此，与 TEXPLORE 类似，[6], 智能体能够捕捉时间t 的动作与时间 $t + l_v$对应奖励之间的相关性。

每个商店智能体都会学习一种策略，执行动作 $\mathcal{A}_v$，即在 t 向仓库 $= [\hat{r}_v(t,1) .., \hat{r}_v(t,K)]$ 为所有 K 产品下达补货订单 $\hat{\mathbf{r}}_v(t)$。我们假设商店和仓库

智能体按批次下单，动作空间被量化为每种产品 n 个可能动作，即 $\hat{r}_v(t,k) \in \{0,1,\ldots,n\}$。数值 n 根据历史数据通过经验推导得出。

3.3 仓库智能体

仓库智能体联合学习两个子策略。首先是其自身的补货策略，用于向供应商下达补货请求 $\hat{\mathbf{r}}_{wh}(t)$。其次是分配策略，用于将其现有库存分配给门店 $v \in \mathcal{V}$；在仓库没有足够库存满足所有门店请求的情况下，所接受的补货数量 $\mathbf{r}_v(t)$ 可能少于请求数量 $\hat{\mathbf{r}}_v(t)$。

仓库状态空间扩展为包含来自门店的补货订单数量 $\hat{\mathbf{r}}_v(t)$，因为这使仓库能够学习分配策略，并在需要时限制实际到达门店的补货量 $\hat{\mathbf{r}}_v(t)$。此外，状态空间还包括其自身的库存 $\mathbf{x}_{wh}(t)$，以及最近 m 个时间段内下达的补货订单 $\mathbf{r}_{wh}(t)$，从而为其自身最优地学习补货策略。其状态和动作空间为：

$$\mathcal{S}_{wh} = \{\mathbf{x}_{wh}(t)\} \cup \{\mathbf{r}_{wh}(t-i)\}_{i=1}^{m} \cup \{\{\hat{\mathbf{r}}_v(t-i)\}_{i=1}^{m} | v \in \mathcal{V}\}, \tag{3.1}$$

$$\mathcal{A}_{wh} = \{\hat{\mathbf{r}}_{wh}(t)\} \cup \{\mathbf{r}_v(t) | v \in \mathcal{V}\}. \tag{3.2}$$

在第 6 节图 2b 中，我们观察到，当仓库状态空间在公式 3.1 中排除门店补货订单数量 $\hat{\mathbf{r}}_v(t)$（标记为 LimWh-ShRwd）时，其为每个商店智能体学习分配策略 $\mathbf{r}_v(t)$ 的能力下降，奖励值也随之降低。通过额外观测 $\hat{\mathbf{r}}_v(t)$（标记为 EnWh-ShRwd），仓库智能体能够优先向相关门店分配补货。据我们所知，我们的工作首次为库存管理引入了这种特定的多智能体架构，并增强了仓库智能体的状态和动作空间。

由于假设供应商的产能无限，需求 $\hat{\mathbf{r}}_{wh}(t)$ 总能得到满足，即 $\hat{\mathbf{r}}_{wh}(t) = \mathbf{r}_{wh}(t)$。理想情况下，仓库应有足够的库存来满足所有门店的请求 $\hat{\mathbf{r}}_v(t)$。当仓库向门店补充的数量较少，即 $\mathbf{r}_v(t) \leq \hat{\mathbf{r}}_v(t)$ 时，系统受到惩罚的机制将在第 5 节中说明。

4 库存管理环境

该环境通过一组表格实现，用于跟踪数量 $\mathbf{x}(t)$、$\mathbf{s}(t)$、$\hat{\mathbf{s}}(t)$、$\mathbf{r}(t)$、$\hat{\mathbf{r}}(t)$，以及用于记录补货订单状态的内部表格。这些表格使用 CuPy [14]实现，后者借助 CUDA 工具包库 [13]（如 cuBLAS、cuRAND 和 cuSOLVER）在图形处理器上执行矩阵运算。因此，如图 1b 所示，环境表格可在常数时间内更新（受图形处理器限制）。附录 A 对环境实现进行了详细说明。虽然该环境能够处理任意数量的产品，但如第 6 节所述，我们实现的多智能体强化学习算法在训练过程中最多可在 $K = 10$ 个产品的情况下收敛。我们坚信，未来在尝试实现用于管理数千种产品库存的多智能体强化学习时，仍可使用同一环境。

在我们的环境中，一个回合从仓库和门店顶点处每种产品的初始库存水平开始，并持续 $T = 30$ 个时间段。每个时间段可以类比为一天的销售活动，在此期间，每个智能体都会下达各自的补货订单，并在相应的提前期后收到货物。此外，每个时间段内，门店都会根据从每种产品对应的需求分布中抽样得到的客户需求，售出一定数量的库存。

4.1 门店动态

在每个时间段 t，所有产品的客户需求 $\hat{\mathbf{s}}_v(t) = [\hat{s}_v(t,1) \ldots, \hat{s}_v(t,K)]$ 均从各产品特定的需求分布中抽样得到。受 $\mathbf{x}_v(t)$ 的限制，实际售出的库存量为：$\mathbf{s}_v(t) = \min(\hat{\mathbf{s}}_v(t), \mathbf{x}_v(t))$，其中 min() 函数按元素逐个作用于每种产品。

在每个时间段结束时，门店节点 v 收到若干时间段前下达的已接受补货订单 $\mathrm{r}_v(t-l_v)$ 后，其库存会更新为：$\mathrm{x}_v(t+1) = \mathrm{x}_v(t) - \mathrm{s}_v(t) + \mathrm{r}_v(t-\mathrm{l}_v)$。根据 $\mathrm{x}_v(t)$ 和补货历史 $\{\mathrm{r}_v(t-i)\}_{i=1}^{l_v}$，门店 v 向仓库下达补货订单 $\hat{}\mathrm{r}_v(t)$。我们无需显式建模未来需求，因为智能体会根据学习到的策略和过去的需求隐式地对其进行预测。我们在第 6 节图 2b 中展示了这一特性：能够提前一个提前期 l_v 看到真实客户需求 $\hat{}\mathrm{s}_v(t+l_v)$ 的预言机实现，其表现并不优于 CMARL。环境通过逐元素设置 $\mathrm{x}_v(t) = \min(\mathrm{x}_v(t), \mathrm{c}_v)$，确保库存 $\mathrm{x}_v(t)$ 始终不超过相应产品的最大货架容量 c_v。无法放在货架上的库存会被丢弃，不会转化为销售。不过，这些库存仍需采购并存放在仓库，以便分配给门店。任何被丢弃的库存都会按比例降低共享奖励（后文公式 (5.5) 对其进行定义），因为它会产生仓库的采购成本和库存持有成本，却不会带来任何销售收入。因此，门店会隐式地避免提出会导致库存超出货架容量的补货请求。

4.2 仓库动态

根据门店的补货请求 $\hat{}\mathrm{r}_v(t)$，仓库使用其分配子策略来决定接受的补货订单 $\mathrm{r}_v(t)$。在时间段 t 结束时，当仓库收到其在 l_{wh} 个提前期之前下达的补货 $\mathrm{r}_{wh}(t-l_{wh})$ 后，其库存更新如下：

$$\mathbf{x}_{wh}(t+1) = \mathbf{x}_{wh}(t) - \left(\sum_{v\in\mathcal{V}} \mathbf{r}_v(t)\right) + \mathbf{r}_{wh}\left(t-l_{wh}\right).$$

根据可用库存，仓库向供应商下达补货订单 $\hat{}\mathrm{r}_{wh}(t)$，该订单总会被接受，因为供应商没有库存约束。环境再次通过逐元素设置 $\mathrm{x}_{wh}(t) = \min(\mathrm{x}_{wh}(t), \mathrm{c}_{wh})$ 来强制执行最大货架容量，其中向量 c_{wh} 表示仓库中每种产品的货架容量。由于货架容量约束而被丢弃的任何库存，虽然是仓库下达补货请求 $\hat{}\mathbf{r}_{wh}(t-l_{wh})$ 时采购的，但不会用于向门店分配 $\mathrm{r}_v(t)$。式（5.5）中定义的系统级奖励会按比例受到惩罚，因为采购这些额外库存的成本无法通过门店销售收回。因此，仓库智能体会隐式地尽量减少下达无法存放在货架上的多余补货请求。

5 奖励结构

由于目标是最大化系统级奖励（整个供应链的利润），智能体之间为争夺奖励而竞争的情况并不可取。为确保智能体充分协作，我们采用共享奖励结构，奖励每个智能体选择有利于整个系统的动作。这与 [23] 中为门店智能体和仓库智能体分别规定的局部奖励公式不同。共享奖励有助于加快学习过程的收敛，并找到更优的解决方案 [17], ，同时降低出现次优行为的风险，例如为了追求个体目标而牺牲系统级目标 [15]。图 2b 中的结果证实了共享奖励结构优于个体奖励，这一点将在第 6 节讨论。我们的共享奖励函数由以下部分构成：

5.0.1 销售收入

如第 4.1 节所述，每家门店根据顾客需求销售一定数量的库存 $\mathrm{s}_v(t)$ $\mathrm{d}\,\hat{}\mathrm{s}_v(t)$。总收入 所有门店所创造的收入为：

$$P_r(t) = \sum_{k=1}^{K}\sum_{v\in\mathcal{V}} \left[s_v(t,k) \times \theta_{\mathrm{SP}}(k,v)\right], \tag{5.1}$$

其中， $\theta_{SP}(k, v)$ 是 v 处 k^{th} 产品的单件销售价格。在大多数大型零售商的供应链中，库存从仓库转运至门店时不会发生中间销售。因此，我们不会明确对仓库的销售进行建模，并将其销售收入设为零。

5.0.2 库存持有成本

门店和仓库中的每单位现有库存通常都会产生与制冷、清洁、电力等相关的维护成本。如果没有这项成本，仓库和门店始终通过下达最大允许数量的补货订单，将库存保持在接近满容量的状态，对双方而言通常都是最优选择。为抑制这种行为，我们对持有库存引入惩罚，其定义如下：

$$P_h(t) = \sum_{k=1}^{K} \sum_{v \in \mathcal{V} \cup wh} \left[x_v(t, k) \times \theta_h(k, v) \right], \tag{5.2}$$

其中， $\theta_h(k, v)$ 表示在 v 持有一单位 k^{th} 产品的成本。

5.0.3 采购成本

如前所述，仓库采购的是提前 l_{wh} 个提前期向供应商下达的补货订单 $\mathbf{r}_{wh}(t - l_{wh})$。采购总成本为：

$$P_p(t) = \sum_{k=1}^{K} \left[r_{wh}(t - l_{wh}, k) \times \theta_{CP}(k, wh) \right], \tag{5.3}$$

其中， $\theta_{\mathrm{CP}}(k, wh)$ 表示采购一单位产品 k 的成本。在大多数零售商的供应链中，库存从仓库转移到门店不会产生中间销售，因为这种转移属于组织内部行为。因此，我们不对门店计入采购成本。

5.0.4 未履约惩罚

当仓库的库存不足以满足门店补货请求总量，或门店库存不足以满足客户需求时，我们会施加未履约订单惩罚。该惩罚定义如下：

$$P_u(t) = \theta_u \sum_{k=1}^{K} \mathrm{ReLU}\left(\sum_{v \in \mathcal{V}} \hat{r}_v(t, k) - x_{wh}(t, k) \right) + \theta_u \sum_{k=1}^{K} \sum_{v \in \mathcal{V}} \left[\hat{s}_v(t, k) - s_v(t, k) \right], \tag{5.4}$$

其中， θ_u 是该惩罚的超参数，$\mathrm{ReLU}()$ 是修正线性单元函数，定义如下：$\mathrm{ReLU}(x) = \max(0, x)$。

5.0.5 共享奖励规范

公式 5.1-5.4 中的奖励和惩罚用于计算回合中每个时间段的总奖励，定义如下：

$$P(t) = P_r(t) - \left(P_p(t) + P_h(t) + P_u(t) \right). \tag{5.5}$$

这个表达式还可以选择纳入运输和劳动力成本。由于我们的实验涉及的产品数量较少，因此未将这些成本纳入。

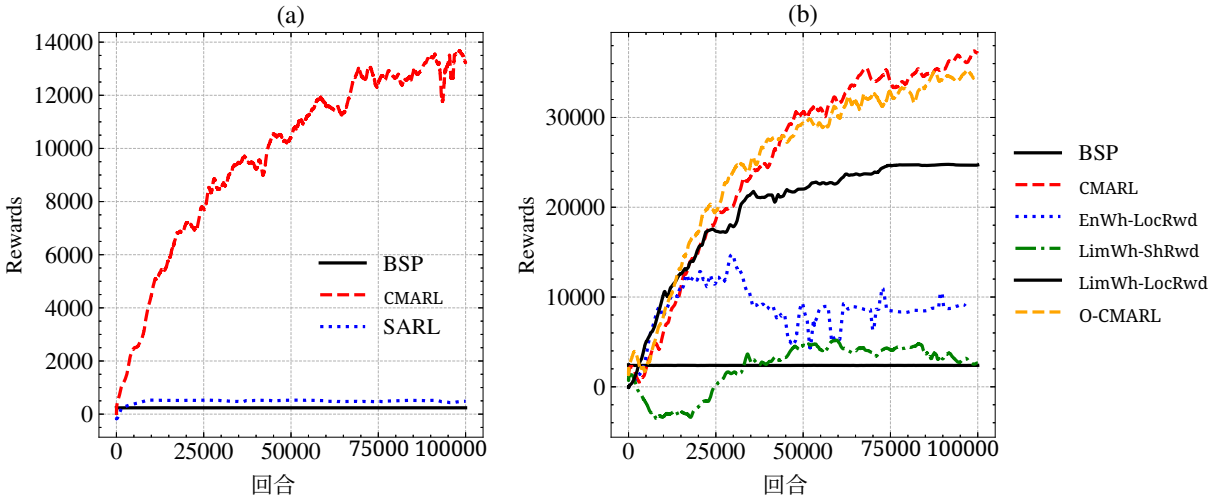


图 2：以下场景的训练奖励：(a) 线性供应链；(b) 包含 10 家门店的发散型供应链；两者均只有 1 种产品。

6 实验结果

我们使用近端策略优化（PPO） [20] 而非普通策略梯度（VPG）来实现所有强化学习智能体，因为已知前者具有更高的样本效率、更好的训练稳定性、更有效的动作空间探索能力，以及对高维状态空间的更强鲁棒性 [9, 19, 24, 21, 10]。假设单一产品服从均值参数为 $10 \leq \mu \leq 1000$ 的泊松分布。每个系统均在一台 8 核虚拟中央处理器和一块 NVIDIA Tesla P4 GPU 上训练 100, 000 个回合。我们将 $\theta_u \gg \theta_h$ 设为该值，因为实验表明，这样既能最大限度地减少缺货，又能保持库存稳定。

6.1 线性供应链

我们首先考虑一个包含 2 个顶点 $\mathcal{U} = \{wh, v\}$ 的线性供应链，其中有一家仓库和一家门店，经营单一产品。我们将 CMARL 与基库存策略（BSP） [1] 等库存控制策略以及单智能体强化学习（SARL）进行比较。在 BSP 和 SARL 的实现中，仓库与门店共用全局的动作与观测空间。动作空间是在时间 t所有顶点请求的补货数量集合： $\mathcal{A} = \{\hat{\mathrm{r}}_{wh}(t), \hat{\mathrm{r}}_{v}(t)\}$。同样，观测空间是所有顶点的现有库存及其过去已接受的补货订单集合：$\mathcal{S} = \{\mathrm{x}_{wh}(t)\} \cup \{\mathrm{r}_{wh}(t-i)\}_{i=1}^{l_{wh}}\} \cup \{\mathrm{x}_{v}(t)\} \cup \{\mathrm{r}_{v}(t-i)\}_{i=1}^{l_{v}}$。BSP 的实现遵循 [7]中所述的方法。

对于这 3 种方法，我们使用公式 5.5 计算奖励，并将其绘制在图 2a 中。我们观察到，SARL 的奖励（超过 450）高于基库存策略，而后者不超过 300。这一结果得到了 [7, 23, 18, 16] 的支持：经过充分训练后，SARL 是优于基于优化的方法的选择。然而，我们提出的 CMARL 框架远远优于 SARL，其奖励超过后者的 30 倍。所提出的多智能体强化学习系统还在远短于 SARL 达到最大值所需时间的情况下，达到了并超过了 SARL 框架获得的奖励。这可能是因为在多智能体环境中，智能体可以承担不同的角色并表现出不同的行为，与单个智能体相比，这会带来更加多样且高效的状态空间探索。正如 [11, 26], 中所观察到的，多个智能体的存在能够增加有效样本量，从而实现更快的收敛。

6.2 发散型供应链

6.2.1 单一产品

现在，我们将环境扩展为具有一个仓库顶点和 10 个商店顶点的发散型拓扑结构，并首先讨论单一产品的实验。基库存策略的奖励上限可设为 $10 \times 360 = 3$，即 600。这是因为，一个包含 $|\mathcal{V}|$ 个门店的发散型供应链可以建模为 $|\mathcal{V}|$ 条带有仓库顶点约束的线性供应链。移除这些约束后，每条 $|\mathcal{V}|$ 线性供应链中可能的行为集合都会扩大，从而提高潜在奖励。在训练过程中，平均奖励超过 35,000，约为基库存策略预计奖励的 10 倍，如图 2b 所示。

我们通过以下方式研究多种多智能体强化学习系统变体：（i）将公式（3.1）中定义的仓库智能体的观测空间限制为仅包含其过去的补货操作，并排除门店的补货请求（LimWh）；（ii）为每个智能体设置独立的局部奖励，而不是系统级共享奖励（LocRwd）。具体而言，我们研究了 5 种不同的多智能体强化学习配置：（i）CMARL，即 EnWh-ShRwd；（ii）EnWh-LocRwd；（iii）LimWh-ShRwd；（iv）LimWh-LocRwd；以及（v）CMARL 的预言机实现（O-CMARL），其中时间 t 的门店智能体可以看到实际客户需求 $\hat{\mathrm{s}}_v(t+l_v)$，并提前 l_v 个提前期，根据此需求提出补货请求 $\hat{\mathrm{r}}_v(t)$。如图 2b 所示，同时采用增强型仓库智能体观测空间和共享奖励结构的 CMARL 系统具有最高的奖励值。其表现也不逊于预言机实现，这表明我们并不需要显式的需求预测模型来预见未来需求。智能体能够根据其学习到的策略和过去的需求对未来需求进行隐式预测。表现最差的配置是未采用增强型仓库智能体的系统级共享奖励配置（LimWh-ShRwd）。该配置平均分配奖励，无法向系统提供关于如何改进行为的明确信号。因此，当仓库无法满足所有补货请求时，分配决策基本上是随机作出的，仓库也无法学会高效的分配策略。同样地，为增强型仓库智能体的每个智能体设置个体奖励（EnWh-LocRwd），会导致仓库智能体比门店智能体更快地收敛到局部有利的策略，因为仓库智能体能够获得比门店智能体更多的环境信息。

MARL 系统既不采用增强型仓库智能体，也不使用共享奖励（LimWh-LocRwd），因此能够避免这些问题。尽管智能体会争夺奖励，但它们彼此之间的策略收敛速度大致相近。因此，作为一个系统，其表现排名第二。不过，这仍会导致次优行为，如图 3b 所示：在训练阶段智能体的策略收敛后，运行一个测试回合来观察各时间段的库存水平。图 3b 所示的 LimWh-LocRwd 配置中，仓库智能体收敛到一种策略，即不断向供应商提出最大量的补货请求；而门店智能体的补货请求则反复无常，导致频繁缺货（因为其库存经常降至零）。相比之下，所提出的 CMARL 系统能够更好地管理库存。如图 3a 所示，CMARL 系统中的仓库库存相对稳定，不会持续增加或减少。这意味着持有成本降低，因为销售后剩余的库存得以最小化，同时仍能满足所有订单，且库存始终不会降至 0，从而避免缺货。

6.2.2 多种产品

为将 CMARL 扩展到经营多种产品的零售商供应链，我们进行了最多包含 10 种产品的实验。如第 3 节所述，每个智能体的动作与观测空间会随供应链中的产品数量线性增加。结合第 4 节讨论的特定 GPU 并行化环境实现，这使我们能够同时训练智能体针对多种产品的策略，从而避免训练时间开销，并能够分别捕捉每种产品需求分布的变化。图 4a 显示，对于每个顶点管理 10 种产品库存的供应链，训练奖励持续上升，这表明智能体最终实现了全系统收敛并获得最优策略。

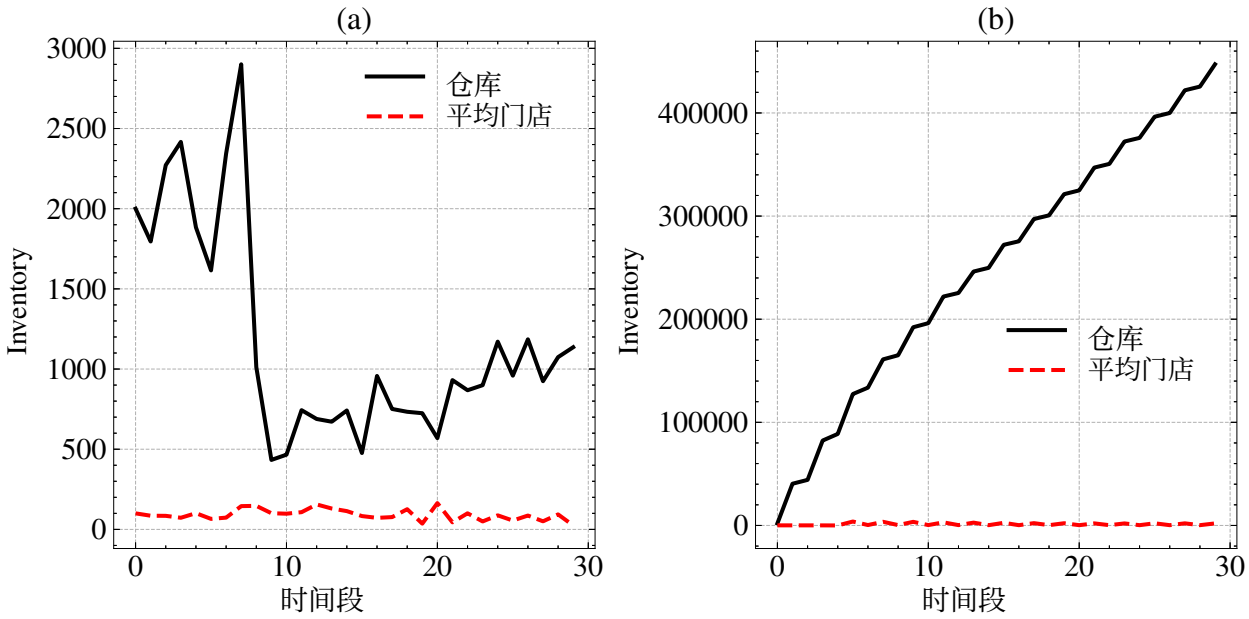


图 3：包含 10 家门店、1 种产品的供应链中，一个回合（30 个时间段）的仓库库存和门店平均库存，其中 (a) 为 CMARL，(b) 为 LimWh-LocRwd。

相比之下，像 [23] 中那样在不同产品间共享策略参数的实现，必须针对每种产品分别训练，因此训练时间会增加 K 倍。此外，这种做法也较难捕捉不同产品类型之间的显著差异，从而降低整体表现。举例来说，水果这类易腐商品与电视机这类电子设备在交易量、保质期和客户需求方面存在根本差异，因此为两者共享库存管理策略的参数通常并非理想选择。为突出这一特点，我们实现了 CMARL 的一个变体（ShPol-CMARL）：每个智能体的动作与观测空间每次只能接收一种产品作为输入，并通过在不同产品间共享策略参数，依次针对 10 种产品进行训练。图 4b 比较了单个回合的奖励：一方是第 3 节和第 4 节所述、针对 10 种产品分别具有独立动作和状态维度的 CMARL 系统，另一方是共享策略变体（ShPol-CMARL）。我们发现，CMARL 获得的奖励是 ShPol-CMARL 的两倍。在图 4 的实验中，决定产品需求的每个泊松分布的 μ 值都不相同。由于下一节所述的原因，我们当前的 CMARL 实现最多只能处理 10 种产品。

7 结论与未来工作

我们提出了一个库存管理问题的多智能体强化学习机制系统，该系统通过GPU并行化环境实现，环境由一个仓库和多个门店组成。针对仓库智能体，我们设计了具有增强观测空间和动作空间的智能体—环境动态，使其能够有效学习分配子策略。此外，CMARL中的共享奖励机制鼓励智能体之间开展协作，共同优化零售商的供应链需求。我们证明，在管理单一产品的库存时，CMARL优于优化BSP方法、单智能体强化学习，以及其他采用个体奖励结构和有限仓库观测空间的多智能体强化学习配置。我们的实验结果表明，与 [23], 中对所有产品共享单一策略的做法相比，为每种产品设置对应的个体动作空间和观测空间维度，不仅能够获得更高的奖励值，还能

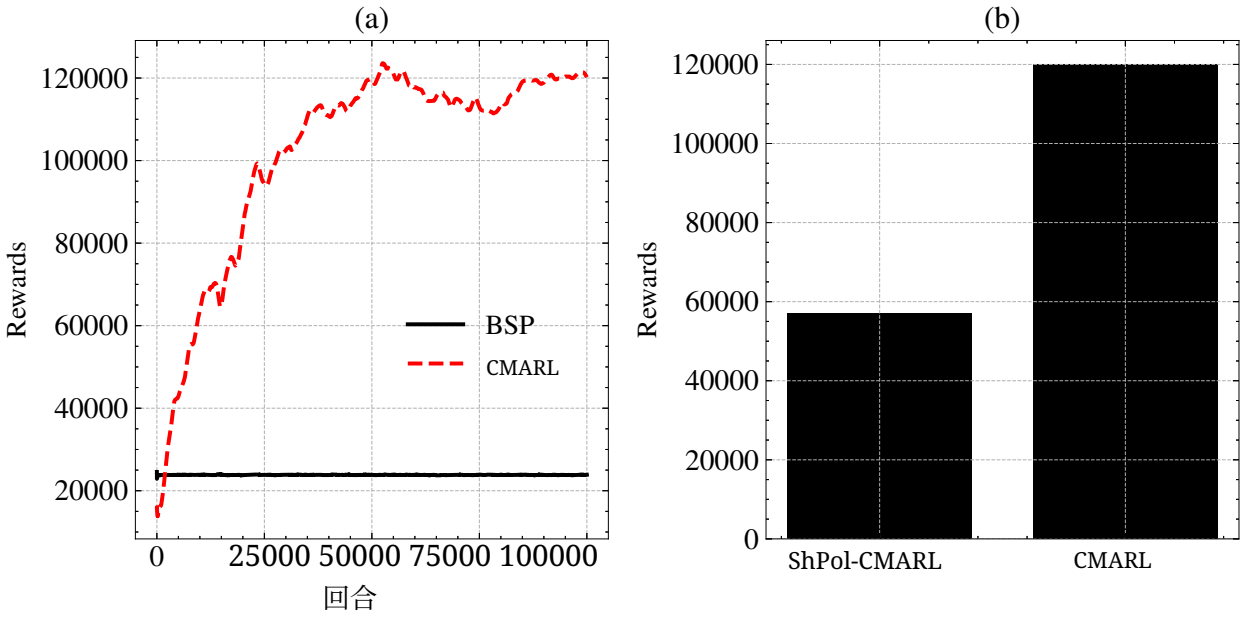


图4：(a) 对于一个包含10家门店、10种产品的发散型供应链，单个回合的训练奖励；(b) 测试奖励。

同时针对所有产品进行训练。我们对仓库智能体和门店智能体进行同步训练，无需像阶段 [23]中那样，在仅训练门店智能体时假设仓库容量不受限制这一或许并不现实的条件。

对于我们的系统，若要提高奖励值，就需要为每种产品纳入各自独立的观测和动作维度；随着产品数量增加，智能体的动作与观测空间会因此发生组合爆炸。这反过来限制了当前 CMARL 框架处理超过 10 种产品的能力，而大型零售商的库存管理系统则需要处理数千种产品。未来的研究方向包括探索能够处理大型动作空间的扩展方法，例如 [3]，中提出的方法，以及分支深度Q网络等替代方案 [25]。

参考文献

[1] Anbazhagan, N.、Wang, J.、Gomathi, D.：《带重试需求的基库存策略》。《应用数学建模》37(6)，4464–4473（2013）[2] D'Atri, A.、Spagnoletti, P.、Banzato, A.、Bonelli, C.、D'Atri, E.、Traversi, V.、Zenobi, P.：《从供应链到供应网络：啤酒游戏的演变》。《IFAC会议论文集》42(4)，1316–1321（2009）[3] Farquhar, G.、Gustafson, L.、Lin, Z.、Whiteson, S.、Usunier, N.、Synnaeve, G.：《扩展动作空间》。载于：《国际机器学习大会》论文集。第119卷，第3040–3051页。机器学习研究论文集（2020）[4] Giannoccaro, I.、Pontrandolfo, P.：《供应链中的库存管理：一种强化学习方法》。《国际生产经济学期刊》78(2)，153–161（2002）[5] Goodfellow, I.J.、Mirza, M.、Xiao, D.、Courville, A.、Bengio, Y.：《基于梯度的神经网络中灾难性遗忘的实证研究》。arXiv预印本，arXiv:1312.6211（2013）

[6] Hester, T.、Stone, P.：《TEXPLORE：面向机器人的实时高样本效率强化学习》。《机器学习》90，385–429（2013）[7] Hubbs, C.D.、Perez, H.D.、Sarwar, O.、Sahinidis, N.V.、Grossmann, I.E.、Wassick, J.M.：《Or-gym：面向运筹学问题的强化学习库》。CoRR abs/2008.06319（2020）[8] Kara, A.、Dogan, I.：《用于制定易腐品库存系统订货策略的强化学习方法》。《专家系统与应用》91，150–158（2018）[9] Konda, V.、Tsitsiklis, J.：《演员-评论家算法》。《神经信息处理系统进展》12（1999）[10] Lillicrap, T.P.、Hunt, J.J.、Pritzel, A.、Heess, N.、Erez, T.、Tassa, Y.、Silver, D.、Wierstra, D.：《基于深度强化学习的连续控制》。arXiv预印本，arXiv:1509.02971（2015）[11] Lowe, R.、Wu, Y.I.、Tamar, A.、Harb, J.、Pieter Abbeel, O.、Mordatch, I.：《混合合作竞争环境中的多智能体演员-评论家》。《神经信息处理系统进展》 30 （ 2017 ） [12] Miller, D.R.：《马尔可夫过程》，第 486–490 页。施普林格美国，纽约（2001）[13]NVIDIA、Vingelmann, P.、Fitzek, F.H.：《CUDA，版本：10.2.89》（2020），https://developer.nvidia.com/cuda-toolkit[14] Okuta, R.、Unno, Y.、Nishino, D.、Hido, S.、Loomis, C.：《CuPy：兼容NumPy的英伟达图形处理器计算库》。载于：机器学习系统研讨会（LearningSys）论文集，第三十一届神经信息处理系统年会（NIPS）（2017）[15] Omidshafiei, S.、Pazis, J.、Amato, C.、How, J.P.、Vian, J.：《部分可观测性下的深度分散式多任务多智能体强化学习》。载于：《国际机器学习大会》。第2681–2690页。机器学习研究论文集（2017）[16] Oroojlooyjadid, A.、Nazari, M.、Snyder, L.V.、Tak´aˇc, M.：《啤酒游戏的深度Q网络：用于库存优化的深度强化学习》。《制造与服务运营管理》24(1)，285–304（2022）[17] Panait, L.、Luke, S.：《合作式多智能体学习：现状》。《自主智能体与多智能体系统》11，387–434（2005）[18] Peng, Z.、Zhang, Y.、Feng, Y.、Zhang, T.、Wu, Z.、Su, H.：《需求不确定性下容量约束供应链优化的深度强化学习方法》。载于：2019年中国自动化大会（CAC）。第3512–3517页。电气电子工程师学会（2019）[19] Schulman, J.、Levine, S.、Abbeel, P.、Jordan, M.、Moritz, P.：《信赖域策略优化》。载于：《国际机器学习大会》论文集。第37卷，第1889–1897页。机器学习研究论文集（2015）[20] Schulman, J.、Wolski, F.、Dhariwal, P.、Radford, A.、Klimov, O.：《近端策略优化算法》。arXiv预印本，arXiv:1707.06347（2017）[21] Silver, D.、Lever, G.、Heess, N.、Degris, T.、Wierstra, D.、Riedmiller, M.：《确定性策略梯度算法》。载于：《国际机器学习大会》论文集。第32卷，第387–395页。机器学习研究论文集（2014）[22] Stranieri, F.、Stella, F.：《供应链库存管理的深度强化学习方法》。CoRR abs/2204.09603（2022）

[23]Sultana, N.N., Meisheri, H., Baniwal, V., Nath, S., Ravindran, B., Khadilkar, H.：用于供应链多产品多节点库存管理的强化学习。CoRR abs/2006.04037 (2020)

[24] Sutton, R.S., Barto, A.G.：强化学习：入门。麻省理工学院出版社 (201 8)

[25] Tavakoli, A., Pardo, F., Kormushev, P.：用于深度强化学习的动作分支架构。载于：AAAI人工智能大会论文集。第 32 卷 (2018)

[26] Yang, Y., Hao, J., Chen, G., Tang, H., Chen, Y., Hu, Y., Fan, C., Wei, Z.：用于深度多智能体强化学习的Q值路径分解。载于： 37th 国际机器学习大会论文集。第 119 卷，第 10706–10715 页。机器学习研究论文集 (2020)

A 环境实现

除主要数量 x(*t*)、s(*t*)、ˆs(*t*)、r(*t*)、ˆr(*t*) 外，环境还会跟踪每个时间段 t 的“在途”库存，记为 $\mathbf{x}_v^{\text{transit}}(t)$。它是已下达但尚未交付的补货订单总量。对于顶点 $v \in \mathcal{V}$ 处的门店，在途库存 $\mathbf{x}_v^{\text{transit}}(t) = [x_v^{\text{transit}}(t,1) \ldots, x_v^{\text{transit}}(t,K)]$ 会在每个时间段结束时更新如下：

$$\mathbf{x}_v^{\text{transit}}(t+1) = \mathbf{x}_v^{\text{transit}}(t) - \mathbf{r}_v(t-l_v) + \mathbf{r}_v(t) \tag{A.1}$$

对于仓库顶点 wh，其来自供应商的在途库存 $\mathbf{x}_{wh}^{\text{transit}}(t)$ 会在每个时间段结束时更新如下：

$$\mathbf{x}_{wh}^{\text{transit}}(t+1) = \mathbf{x}_{wh}^{\text{transit}}(t) - \mathbf{r}_{wh}(t-\mathbf{l}_{wh}) + \mathbf{r}_{wh}(t) \tag{A.2}$$

以下 CuPy [14] 表（第 4 节中引用）用于跟踪环境动态，其中 $N = |\mathcal{V}|$ 为商店顶点的数量：

- 现有库存 I、在途库存 $I_{\text{在途}}$、请求补货量 R 和已接受补货量 R 的维度为 $(T+1, N, \hat{2}, K)$，满足 $t \in T$：

$$I(t) = \begin{bmatrix} \mathbf{x}_{wh}(t) & \mathbf{x}_{v_1}(t) \\ \vdots & \vdots \\ \mathbf{x}_{wh}(t) & \mathbf{x}_{v_N}(t) \end{bmatrix} \text{ and } I_{\text{Transit}}(t) = \begin{bmatrix} \mathbf{x}_{wh}^{\text{transit}}(t) & \mathbf{x}_{v_1}^{\text{transit}}(t) \\ \vdots & \vdots \\ \mathbf{x}_{wh}^{\text{transit}}(t) & \mathbf{x}_{v_N}^{\text{transit}}(t) \end{bmatrix} \tag{A.3}$$

$$\hat{R}(t) = \begin{bmatrix} \hat{\mathbf{r}}_{wh}(t) & \hat{\mathbf{r}}_{v_1}(t) \\ \vdots & \vdots \\ \hat{\mathbf{r}}_{wh}(t) & \hat{\mathbf{r}}_{v_N}(t) \end{bmatrix} \text{ and } R(t) = \begin{bmatrix} \mathbf{r}_{wh}(t) & \mathbf{r}_{v_1}(t) \\ \vdots & \vdots \\ \mathbf{r}_{wh}(t) & \mathbf{r}_{v_N}(t) \end{bmatrix} \tag{A.4}$$

其中，对于 $v \in \mathcal{V}$，$\mathbf{x}_v(t) = [x_v(t,1) \ldots, x_v(t,K)], \mathbf{x}_v^{\text{transit}}(t) = [x_v^{\text{transit}}(t,1) \ldots, x_v^{\text{transit}}(t,K)], \hat{\mathbf{r}}_v(t) = [\hat{r}_v(t,1) \ldots, \hat{r}_v(t,K)]$ 以及 $\mathbf{r}_v(t) = [r_v(t,1) \ldots, r_v(t,K)]$。

- 最大库存容量 C 的形状为 $(N, 2, K)$，满足：

$$C = \begin{bmatrix} \mathbf{c}_{wh} & \mathbf{c}_{v_1} \\ \vdots & \vdots \\ \mathbf{c}_{wh} & \mathbf{c}_{v_N} \end{bmatrix} \tag{A.5}$$

其中对于 $v \in \mathcal{V}$，$\mathbf{c}_v = [c_v(1) \ldots, c_v(K)]$。

- 形状为（T, N, K）的门店节点需求 D，满足以下条件 $t \in T$:

$$D(t) = \begin{bmatrix} \hat{\mathbf{s}}_{v_1}(t) \\ \vdots \\ \hat{\mathbf{s}}_{v_N}(t) \end{bmatrix} \tag{A.6}$$

其中，对于 $v \in \mathcal{V}$，$\hat{\mathbf{s}}_v(t) = [\hat{s}_v(t,1) \ldots, \hat{s}_v(t,K)]$。

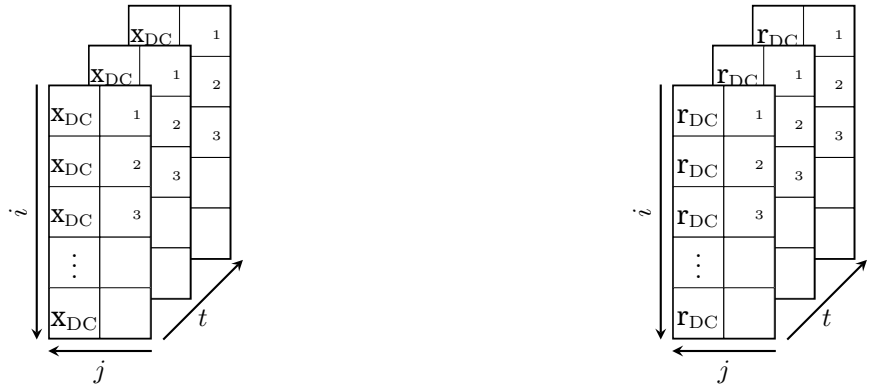
- 形状为 $(T, N, 2, K)$ 的顶点销售额 S，满足 $t \in T$:

$$S(t) = \begin{bmatrix} \mathbf{r}_{v_1}(t) & \mathbf{s}_{v_1}(t) \\ \vdots & \vdots \\ \mathbf{r}_{v_N}(t) & \mathbf{s}_{v_N}(t) \end{bmatrix} \tag{A.7}$$

其中，对于 $v \in \mathcal{V}$，$\mathbf{s}_v(t) = [s_v(t,1) \ldots, s_v(t,K)], \mathbf{r}_v(t) = [r_v(t,1) \ldots, r_v(t,K)]$。

- 奖励 $P \in \mathbb{R}^T$ 满足 $P(t)$（详见正文第 5 节）是时间段 t的奖励。

参照正文图 1(a) 所述的拓扑结构，从仓库到门店的每条图游走都是上述表格中的一行（ P除外）。每一行中，仓库顶点 wh 对应的值会被复制 N 次，使整个环境作为一组 N线性供应链运行。图 5a 和 5b 展示了环境的表格结构，以及一个回合中完整的表格I 和 R 。



(a) 表格 I　　(b) 表 R

图 5：环境表结构示意图，包含 2 个示例。