PPO × Family 第六讲技术问题 Q&A

Q0： 有没有第六节课内容的大白话总结?

A0:

小节	算法要点	代码和实践要点
多智能体协作基础概述	• Dec-POMDP 的定义 • CTDE Framework 和 IGM 条件	• IGM 失效的情形
MAPPO	• Multi-Agent PG 的问题 • MAPPO 如何结合 CTDE • MAPPO 如何设计 global state	• 多粒子运动（MPE） • 一键切换 IPPO 和 MAPPO
HAPPO/HATRPO	• 回忆 TRPO/PPO 的特点 • 多智能体优势函数分解引理 • HAPPO 训练流程	• 多智能体机器人控制协作 （Multi-Agent MuJoCo）
Bags of Tricks in MARL	• 使用Transformer • 共享参数 • 动作掩码和存活掩码 • 策略多样性的平衡与控制	• 星际争霸2微观操作（SMAC） • 谷歌足球博弈（GRF）

Q1: 如何理解第六节课中 Multi-Agent Transformer 的 decoder 对应的动作解码过程“就是”应用了多智能体优势函数分解引理?

A1: 多智能体优势函数分解引理和 Transformer 中自回归（autogressive）向前预测的方式正好对应。分解引理将多智能体之间的优势函数关系分解成一个序列叠加过程，而 Transformer 的设计原理天生适合这样的序列预测任务，具体来说，Transformer 因为使用了 casual mask 所以只能看到之前的 token ，每一步用这些信息向前预测一个新的 token，即对应用分解引理向前迭代一步产生下一个智能体分解后的优势函数。那么当 Transformer 预测完整个序列时，就正好对应自回归地逐步输出了每个智能体的分解后的优势函数，使得网络的运行机理和实际多智能体优化的逻辑一一对应。

Q2: 第六节课中提到熵平衡（Entropy Balance）操作需要根据动作空间的大小相应调整超参数，或者设置自适应衰减的熵权重，请问具体是如何实现的?

A2:

这里类似 SAC 中的 target entropy 的设置，根据动作空间来调节 PPO 中的 entropy weight。在 SAC 算法中，通常将目标熵设置为动作空间的维度，例如对于一个连续动作空间，设动作空间的维度为 d，则目标熵可以设置为-d。

此外，entropy weight 可以在训练中设置为动态衰减，一般采用指数下降衰减的形式可以得到不错的结果。但总体设计都是在前期给较大的 weight，逐渐衰减到后期给较小的 weight，因此初始 entropy weight 可以稍微比常规值大一点。衰减有两种方式，第一种是手动写一个指数下降衰减，类似 eps greedy 中那种，另一种是类似 SAC，设计一个额外的优化项来优化当前 entropy 和 target entropy 之间的差异，其中 target entropy 根据动作空间大小来设计。

Q3: 第六节课中提到 HAPPO 和 HATRPO 算法中，多个智能体的策略模型是独立（每个智能体都有自己的策略模型）或是共享的（所有智能体共享同一个策略模型），相应有什么好处？

A3:

这节课提到的 HAPPO 和 HATRPO 算法适用于协作的多智能体决策环境，在这两个算法的实现中，它们共享多智能体的价值模型，但其各自的策略模型是独立维护的。这样做的好处在于，对于某些环境（如分工合作类游戏），需要合作的智能体执行不同的动作，才有更好的奖励。

Q4: 第六节课作业第一题中，在多智能体优势函数分解引理的执行过程中，为什么需要引入不同的分解顺序，这个操作是必须的吗？

A4:

多智能体优势函数分解的分解顺序在 HAPPO 和 HATRPO 算法的执行层面可以设计为固定顺序或随机顺序。两种情况下，都可以实现对多智能体策略的近端优化，因此，并不是必须要多智能体优势函数分解公式在执行过程中引入不同的分解顺序，但某些论文中有相关实验结论证明，在某些环境中，随机顺序可能会比固定顺序有更好的效果。

Q5: 第六节课作业第二题中，为什么 QMIX 算法已经实现了一种非线性的多智能体价值函数的建模，但却还是不能完美建模某些多智能体的动作价值函数？

A5:

多智能体价值函数分解算法 QMIX 虽然实现了非线性的多智能体价值函数的建模，但它要求被建模的全局价值函数和局部单智能体价值函数之间的关系必须满足正相关关系，这样，全局动作价值函数的最优的动作取值，恰好对应每一个智能体在该状态下的最优动作取值，即 IGM 条件。这个数学上的客观要求较为苛刻，不适用于一些情形（第六讲课程 PPT 中也讲到了相应的反例）。