discussion汇总

基本方法概述

本次可以参考的agent相关比赛参考

agent策略

（看到relic node point tile location）

Code

Relicbound (most voted)是一个好的起步notebook，但是只是汇总了简单的寻路策略

NuralBrain v0.5 model | Train And Win DQN 强化学习框架， RL-agent的应用（但是仍然是手动进行对抗决策，单一agent进行训练）改进训练方法就可以铜牌

一些思路

> ### 行动顺序
>
> 在比赛的每个时间步，我们按以下顺序执行以下步骤：
>
> 1. 移动所有有足够能量进行移动的单位。
> 2. 执行所有有足够能量进行汲取动作的单位的汲取操作。
> 3. 处理碰撞并应用能量空洞场的影响。
> 4. 根据单位所在位置（如能量场和星云地块）更新所有单位的能量。
> 5. **为所有团队生成新单位**，并移除能量小于 0 的单位。
> 6. 确定所有团队的视野/传感器掩码，并据此屏蔽观察结果。
> 7. 让环境中的物体（如小行星、星云地块、能量节点）在空间中移动。
> 8. 计算新的团队积分。
>
> 需要注意的是，每场比赛运行 `params.max_steps_in_match` 步，您可以执行相应数量的动作来影响游戏。然而，您实际上会收到 `params.max_steps_in_match + 1` 帧的观察结果，因为第一帧要么是空的，要么是上一场比赛的最终观察结果（基于这些观察结果执行的动作不会产生任何效果）。

首先，因为每一回合都会生成一个新的单位，所以是一个muti-agnet的过程（需要不同agent的配合）

方向

经典的Centralized muti-agent算法

PRIMAL，MADDPG，GDQ，

MAPPO（2022，将基础的单agent算法扩展到muti-agent上，2022推荐，业界标杆star高，代码多）

https://paperswithcode.com/paper/the-surprising-effectiveness-of-mappo-in

经典的Decentralized muti-agent算法

> 检查agent的运动策略，因为要和比赛规则相同（只能↑、→、↓、←、*）
>
> 并且一把游戏有100个steps，所以max-agent最好要在100以上

~~MAPPER（2020）~~

> MAPPER 为代理添加了四个额外的对角线移动（↗、↘、↙、↖），Max-agent：150没代码

G2RL （2020）

> Max-agent：128

使用DDQN和A*规划器，优点是由于代理之间不需要通信，因此该方法不需要考虑通信延迟，并且可以扩展到任何规模。

https://github.com/Tushar-ml/G2RL-Path-Planning（github issues：有人反馈和论文结果差别较大）

PRIMAL2（2021）

> Max-agent：2048

离散方法，够用了

https://github.com/marmotlab/PRIMAL2

~~PIPO（2022）~~

> Max-agent：512没代码