# 简介

# Collaborative Multi Agent Reinforcement Learning

在协作的多智能体系统（multiagent system）中，每个agent决策的目标是选择出对整体系统最优的action

# Coordination Graphs and Variable Elimination

在协作式的多智能体系统中，每个agent的动作选择会对其他agent产生潜在的影响，即系统中各个agent之间存在依赖关系，一个agent动作的选择会取决于其他agent的决定。比如：TODO ：an example。所以保证各个agent每个时刻选择的动作都是针对整个系统的最优决策，对提高系统的整体收益具有重要的意义。通常这种问题被定义为协调问题 (Coordination Problem)。本章节，我们首先回顾由Guestrin et al. (2002a)提出的问题，计算对由个agents组成的协作式多智能体系统整体最优的动作组合。系统中每个agent 从各自的动作集合 中选择一个action 整体组成一个动作向量（联合动作） ，进一步系统得到环境提供的一个收益 。协调问题的目标是选择一个动作向量 以最大化系统的整体收益 ，即 。

针对这个问题，可以遍历所有可能的动作向量，并且选择可以最大化 的动作向量。但是，很快发现这个思路是不现实的，因为问题的解空间 的规模，随着系统中agent的数量 成指数增长。幸运的是，现实的很多问题中，每个agent的决策只依赖于与其非常相关的一少部分。

由Guestrin et al., 2002a提出的协调图(coordination graphs ,CGs)架构是解决此类策略相互依赖问题的一种方式。此架构假设对一个agent ，其动作的选择只依赖与与其相关的agent  集合。系统整体的收益 由系统中每个agent 的收益 之和组成，即

每个agent 的收益 取决于与其密切相关（有依赖关系）的所有agent 的动作选择，， ，这种相互依赖关系可以通过无向图 表示，其中每个节点 表示agent，每条边 表示相关的agents 需要协调各自动作的选择， 并且 。于是整个系统的协调问题，被拆分为一定数量的局部协调问题，并且减小了问题的规模。

# Payoff Propagation and Max-Plus Algorithm

# Coordination Set Selection

# 实验 Accelerating Norm Emergency

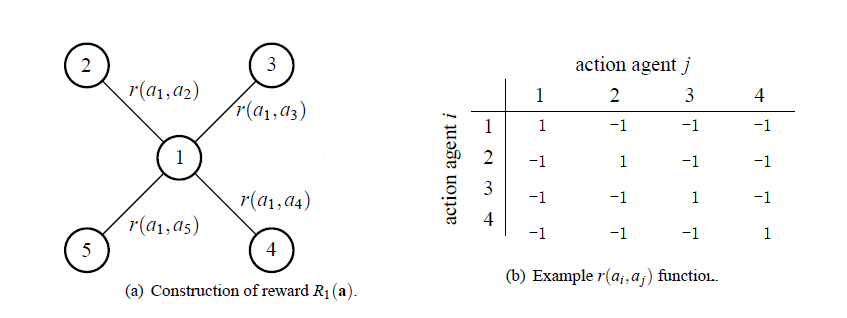
在这一章，针对四五章讨论的方法，用实验进行验证。实验基于单状态(single state)的多agent、多action 协调game

* 1. 基于单状态的协调问题

定义，实验环境是由 个agent组成的协调系统，每个agent独立决策，并且通过对环境的探测与学习，选择对整体最优的动作，以最大化系统整体的收益。系统中每个agent 根据自己的策略，选择出动作action ，随机地与邻居进行交互。随即，当动作执行后，一轮游戏结束，并且每个agent 各自收到一个回报 。每个agent 的目标是选择出各自最优的动作 以最大化系统整体收益 。

每个agent 在每一轮收到的回报 取决于与其交互的邻居agent 。依赖关系可以通过无向图 进行表示，其中每一条边 对应于相邻节点agent 选择各自动作 后的收益 ，如图xxx所示。收益函数 由系统提前设定(但每个agent不能直接获取其准确信息，需要通过学习对收益函数建模)。例如，对每个agent ，, ，回报函数 定义如下：

如果agent 同时选择在对角线上的动作组合 ，其中 ,则双方各自收到reward +1，否则协调失败，收到reward -1, 如图xxx所示。



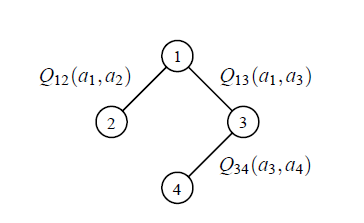
* 1. 实验定义

由于实验环境中，每个agent不能直接获取系统预设的回报函数（或reward table），因此需要通过学习不断与环境进行交互、探测，进而对自己动作集合 中的每个action的优劣进行评估。这里使用Q-learning 来对相邻agent的学习行为进行建模。

针对实验，做出以下定义：

* 是系统中agents的数量。
* 每个agent只有一个状态。
* 是 agent 的动作空间，即agent 有*k*个可选动作。 是系统agents的联合状态空间。其中 ， ，表示当前所有agents的动作选择。
* 是系统预设的reward table，是环境中相邻agents *i,j*选择的action，当 ，否则，。系统整体收益。
* 假定每个agent 可以观察到与其交互的agent 的action选择，并且可以统计最近时间段内，对手选择各个action 的频率。
* 用来记录相邻agent *i,j* 之间的学习经验，以对agent每个action的优劣进行评估。
* 是agent *i* 选择action的策略，。，是系统的整体策略（global policy）。

于是，对于此问题，各agent之间的依赖关系，可以通过协调图 表示，其中每个节点 表示每个agent，每条边 表示相关的agents 的局部Q函数 ，如下图xxx所示。



我们的目标是，找到一个策略，以最大化系统的整体收益。对于一个包含多状态的MDP问题，可以简单的对整体使用single Q-learning：

但是，由于系统整体的策略空间随agents的数量 n，并且往往无法观察到其他agent的所有信息，因此进一步把整体的**Q**函数拆分成各个agent Q函数的线性组合，即：

于是，等式(1) 可以被重新表示为：

上式中，因为 取决于对整体最优的联合action ，因此不能直接拆分为各个agent 局部最优Q值之和。但是我们可以通过VE或Max-Plus等方式，通过使每个agent *i*选择出对整体最优的action 来计算出对整体最优的联合action 。其中。于是对于每一个agent对，有：

对于单状态的协调问题，下一个状态的Q函数没有定义，因此在本实验中，每个agent i在每一轮中，选择自己的action时，直接考虑选择对当前系统整体最优的action ，并且以一定的探索率 随机对动作空间中的action进行探索。

* 1. Coordination action selection

如上节所示，实验中各agents的协调图CG（Coordination Graph）如图xxx所示。于是，每个agent *i* (CG中的节点)，向它的邻居 agent 发送的消息定义为：

其中 表示agent *i*除了 *j* 以外的所有邻居，参数 是为了标准化消息数值的取值范围，防止某些agents 组成的网络中，存在环状结构，进而导致由 *i* 发送出去的消息，一定时间后，又发送到 *i*，进而导致消息值的无限增大。这个消息 是对给定一个目标agent *j* 的动作 ，agent *i* 所能实现的最大收益值的近似。通过最大化与目标agent *j* 之间的平均回报 以及agent *i* 的所有邻居（*j*以外的）向其发送的消息数值总和来计算当前消息 。每个agent 不断向邻居发送消息直到消息的值不再变化，或者到达指定的发送轮数。当网络中所有消息值都达到稳定时，每个消息中都包含了网络中所有边（i,j）上的收益，所以最大化当前消息值即最大化了系统的整体收益**Q**,因此对每个agent *i*，即找到了能最大化整体收益的action 。

整个算法计算过程如Algorithm XXX所示，其中 。在很多情况下，随着消息值的抖动，各个agent的最优action 也在不断变化，因此进一步拓展，只有当agent收到的收益 提高时，才对其最优action 进行更新。

|  |
| --- |
| **Algorithm 1** **runDCOP**(centralized max-plus algorithm for CG(V,E)) |
| 1. initialize for ,m=,fixed\_point=false 2. **while** fixed\_point = false and deadline to send action has not yet arrived **do** 3. // run one iteration 4. fixed\_point = true 5. **for** every agent *i* **do** 6. **for** all neighbors **do** 7. send j messages 8. **if** differs from previous message by a small threshold **then** 9. fixed\_point = false 10. determine and 11. **if** use anytime extension **then** 12. **if** **then** 13. and 14. **else** 16. set best action for agent *i* = 17. **end for** 18. **end for** |

* 1. Coordination Set Selection

Algorithm runDCOP中，消息的数量与系统的CG（Coordination Graph）中，边的条数成正比。对于一个足够大网络结构来说，各个agent的相互依赖关系比较复杂，图中每个节点的度数可能比较大，因而消息发送的次数频繁。但是在现实环境中，每个agent通信的资源数量往往是有限的，并且通信的代价往往比较昂贵，因此我们设计了一种动态调整，选择出当前时刻，对各个agent最有益的最小协调子集（Coordination Set，），以减少在CG中相互依赖的边的数量，进而减少每个agent发送message的数量，进而降低通信的代价。

**定义一：**在稳定状态的Coordination Set（CS）中，对任意agent *i*, 其邻居agent *j*，将无条件的配合agent *i*的行为选择action，以最大化其局部的整体最大收益。对于初始网络中agent *i* 的邻居 （ 是网络初始化时，agent *i* 的所有邻居组成的集合），agent *i* 能够根据对agent *k* 行为的观察，统计出当前其选择各个action的概率，进一步可计算其对 *i* 收益的平均影响。

**定义二：**当选定Coordination Set = C时，agent *i* 的预期最大收益（the potential expected utility） ：

其中，，是agent *i* 对agent *k* 最近一段时间选择各个action的可能性的统计概率。

**定义三：**不与NC协调而造成的预期损失（the potential loss in lack of coordination）

其中，。

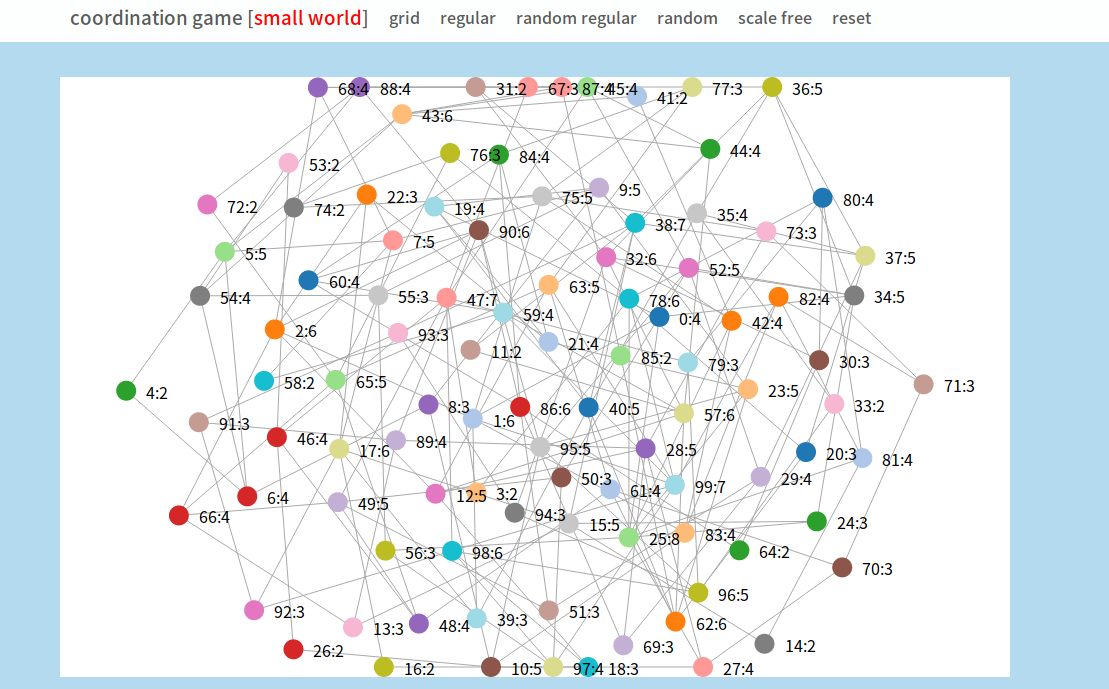
整个算法过程描述如下：其中 代表系统允许的最大损失率，此处设置为0.001

|  |
| --- |
| **Algorithm 2** **computeCoordinationSet**(*i*) |
| 1. initialize 2. find , such that 3. (1) 4. (2) , for all 5. (3) for all 6. return C |

* 1. Coordinated Learning process

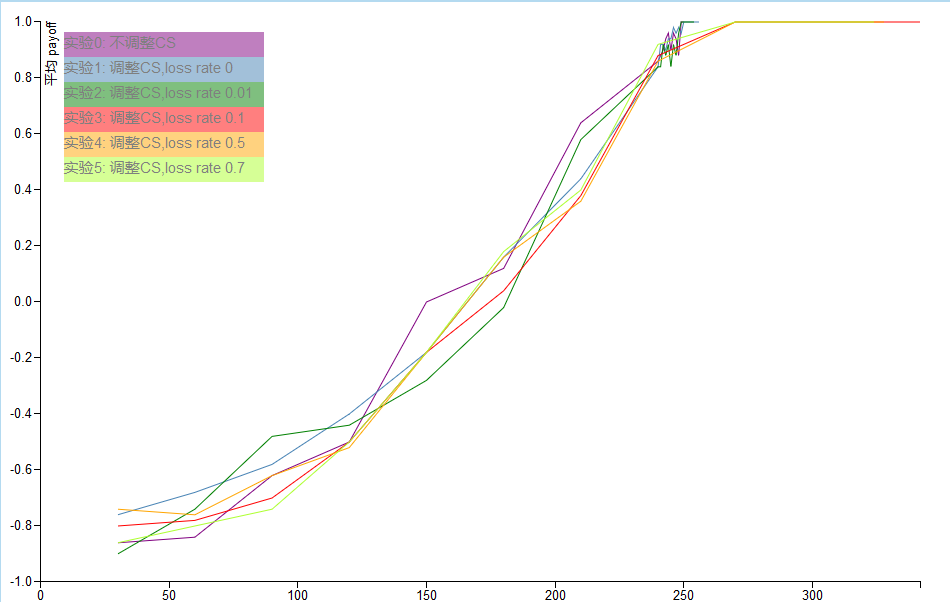
|  |
| --- |
| **Algorithm 3** The coordinated learning process |
| 1. initialize learning rate ,explore rate , loss rate 2. **while** not converge **do** 3. runDCOP() to select the best action for each agent 4. **for** every agent *i* **do** 5. random select a neighbor *j* to interact 6. each agent *i,j* select the its’ action (each select the best action 7. with some explore rate ) 8. each agent observed the reward ,and observed each other’s action (for 9. record ) 10. each agent update its’ Q table 11. agent *i* update its’ learning rate and explore rate with some decay 12. computeCoordinationSet(*i*) 13. **end for** |

* 1. 实验结果
     1. 实验环境：100 agents,10 action
     2. Small world 网络下



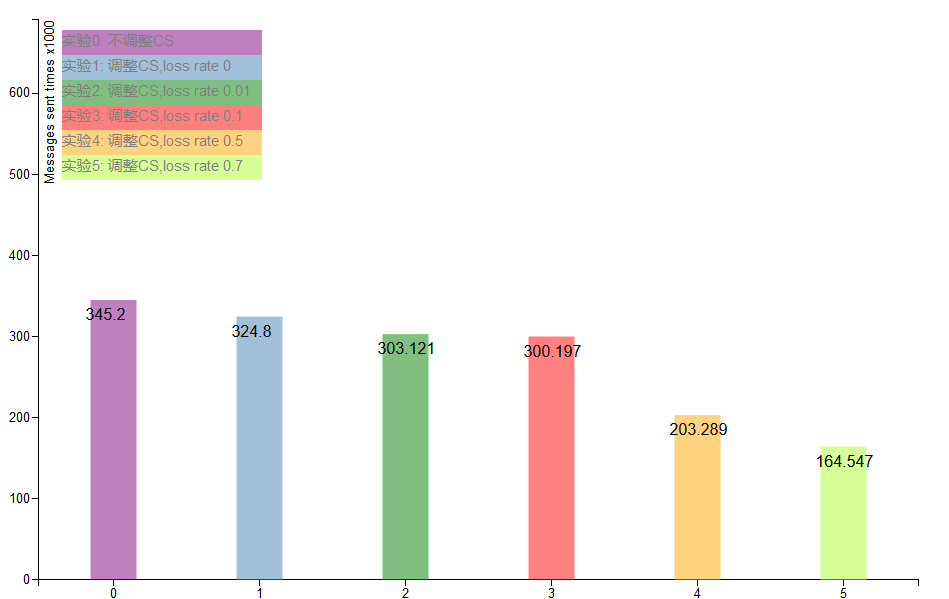
网络结构示意图 small-world

1）收敛情况及轮数对比图



收敛情况及轮数对比图

2）消息（Message）发送次数对比图



收敛时各实验Message发送次数