Multi-agent Democratic Gridworld

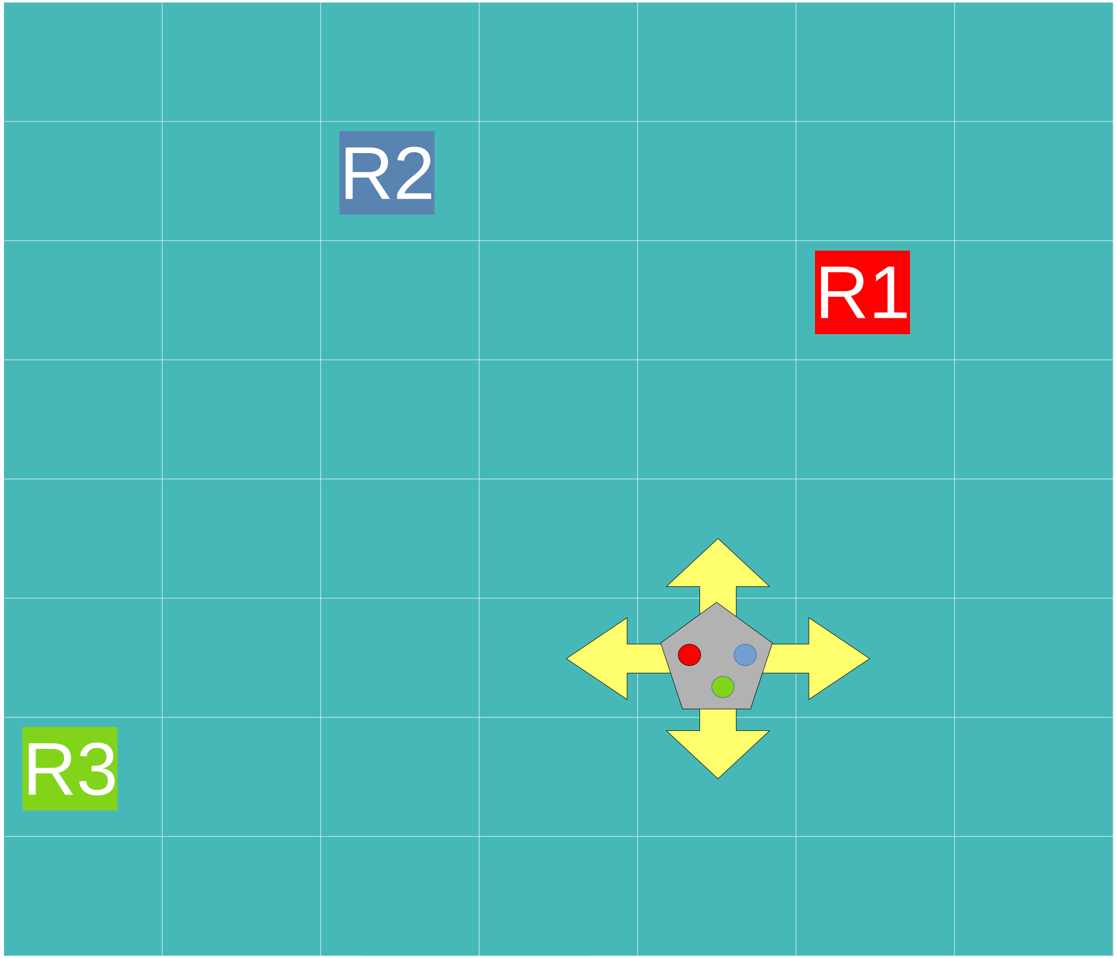
An anthological perspective

Andrea Longo
Reinforcement learning

“

In a grid world a ship is guided by the democratic vote of some agents within it, each agent is intent on reaching a specific cell

”


R2
R1
R3

Getting off

When an agent reach its goal it will not gain further rewards and cannot vote, so its like if it will get off the ship

Playoff

In case multiple directions gains the same number of votes the direction taken will be chosen randomly among those

Win away

The first cell for the ship cannot contain any reward

Goal

Learn a policy for each agent to make it reach its goal (hopefully quickly)

Evaluation

For each episode the number of total steps (or steps of the latest agent) will be the key measurement

Setting alternative

Tests are made either with random initial position or fixed at the middle

Restriction to spice it up

Use of single-agent reinforcement learning techniques readjusted to the multi-agent context

Keeping track

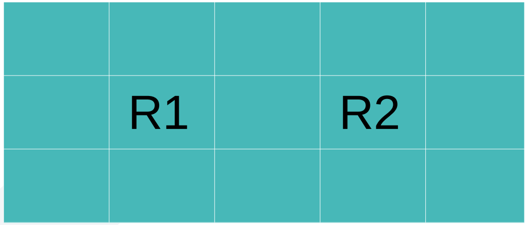
The way in which cooperative or competitive dynamics are established

Setting extensions

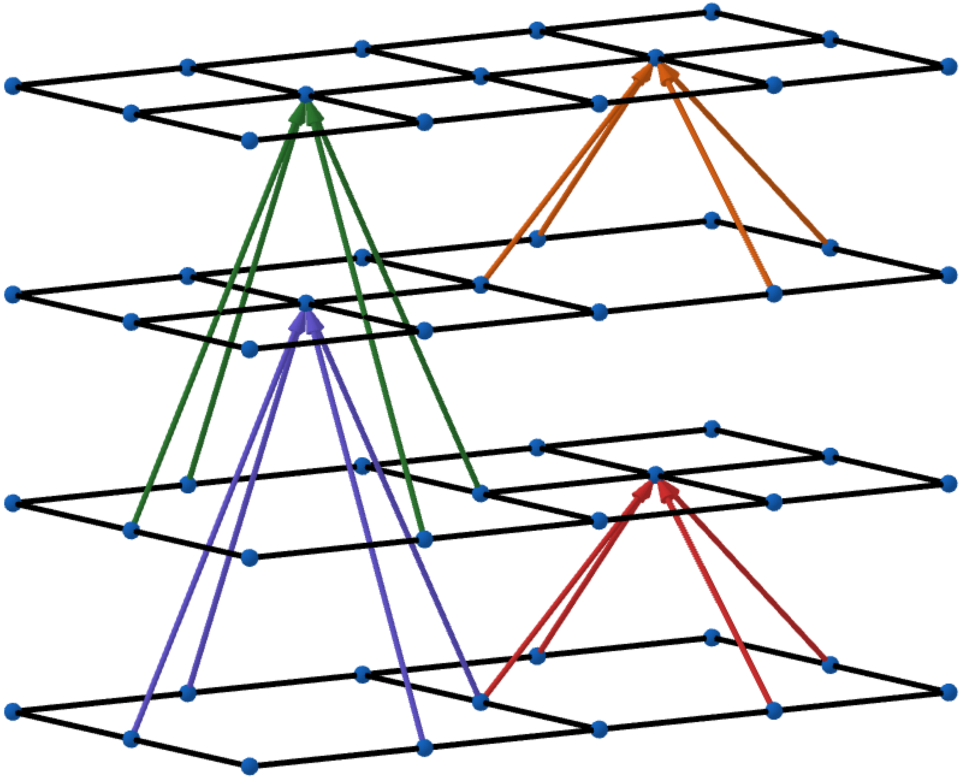
- Multiple reward cells for each agent
- Obstacles inside the grid
- Competition between different agents

Intuition on state space

$|S| = X \cdot Y$



$|S| = (X \cdot Y) \cdot 2^N$



Approaches

Collaborative

Planning + Value contamination

$|S| = (X \cdot Y) \cdot 2^N$

Parametrized Value Iteration

Naive

Multi-agent as Transition Probability

$|S| = (X \cdot Y) \cdot 2^N$

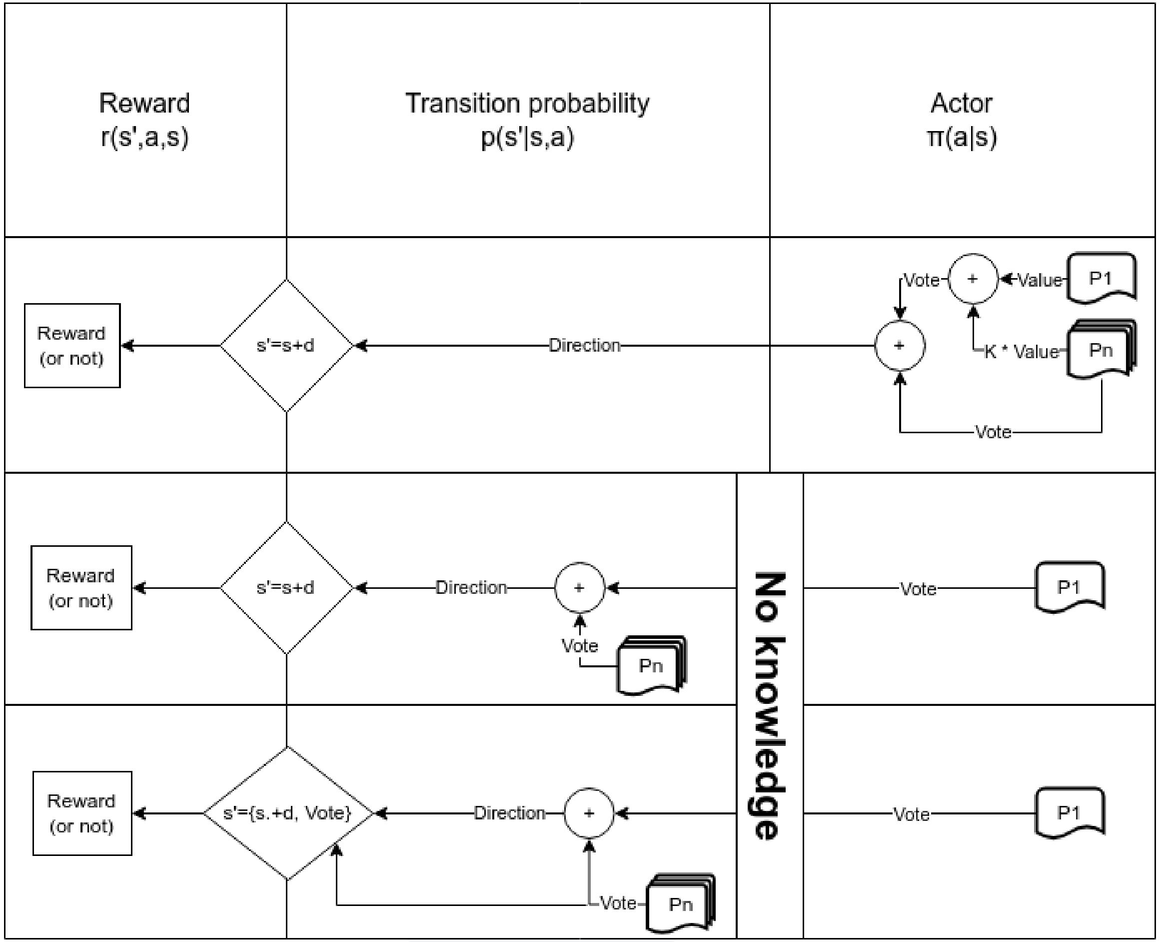
Q-learning

Unstructured

Previous votes into states

$|S| = (X \cdot Y) \cdot (2 \cdot 5)^N$

SARSA VFA with Neural Network



Reward
r(s',a,s)
Transition probability
p(s'|s,a)
Actor
π(a|s)
Reward
(or not)
s'=s+d
Direction
Vote
+
Value
P1
K * Value
Pn
Vote
No knowledge
Reward
(or not)
s'=s+d
Direction
+
Vote
Pn
Vote
P1
Reward
(or not)
s'={s.+d, Vote}
Direction
+
Vote
Pn
Vote
P1

Collaborative approach

Parametrized Value Iteration

- **Individual Value function construction is indipendent**

  simulating actions as direction taken, not votes

- **Contaminated Value function $CV_i = K_i \cdot sum_{j \,!=\, i}(V_j)$**

  with K ~ U(k_min, k_max)

- **Since V expresses only future rewards the adjusted policy is**

$$Policy_i(s) = argmax_a[sum_i[R_i(s,a)] + ^{\gamma} \cdot CV_i(s'(s,a))]$$

Notable attempt

Inside Value Iteration

$$V_{i,t+1}(s) = \max_a[R_i(s,a) + \gamma \cdot V_{i,t}(s'(s,a))]$$

I tried with contamination during evaluation

$$V_{i,t}(s'(s,a)) \mathrel{+}= K_i \cdot \mathrm{sum}_{j\,!=\,i}[V_{j,t}(s'(s,a))]$$

But the underline Bellman Equation rarely converged with K!=0

Best [k_min,k_max]

Random init | 3 agents | 10x10 grid | rewards [goal,empty,wall]=[100,-1,-10]

Naive Approach
Q-Learning

- **Agents constructs their own $Q_i(s,a)$ over the training episodes**

  after training the evaluation's votes are greedy (no ε) and Q does not update

- **Votes are the actions**

  the democratic step is integrated as Transition probability

- **Rewards are individual**

  unlike the previous case

Unstructured Approach

SARSA VFA with Neural Network

- **Features from states, votes, actions**

  $feat_i \rightarrow$ o-h(X) + o-h(Y) + o-h($alives_{j\,!=\,i}$) + lastVotes(actions) [+ o-h(actions)]

  |feat| = width + height + (n_agents – 1) + 4 [+ 4]

- **Votes are the actions**

  the democratic step is integrated as Transition probability

- **Rewards are individual**

  unlike the previous case