MARL Technologies

Multi Agent Reinforcement Learning

Domini Davide, Folin Veronika

Dipartimento di Informatica – Scienza e Ingegneria (DISI)
Alma Mater Studiorum – Università di Bologna

Sistemi Distribuiti

November 15, 2022

Motivazioni ed obiettivi

Motivazioni:

- Rapido sviluppo in ambito *Pervasive Computing* e *Artificial Intelligence*
- Siamo circondati da sistemi intelligenti
- Si vuole aumentare l'autonomia di questi sistemi

Obiettivi:

- Effettuare una panoramica teorica del paradigma MARL
- Selezionare alcune delle tecnologie più promettenti ed effettuare degli esperimenti
- Valutare le potenzialità e i limiti delle tecnologie

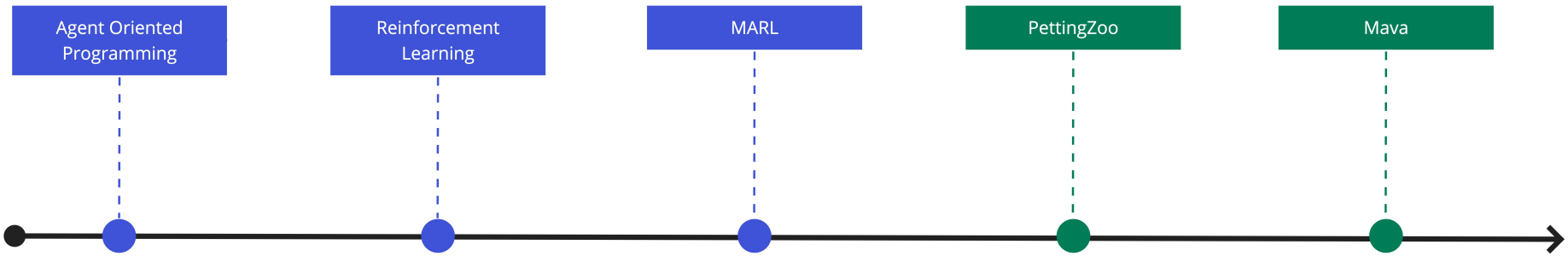


Figure: Roadmap del progetto

Paradigma ad agenti

- Caratteristiche dei moderni **sistemi distribuiti**:
  - Diffusi
  - Sempre connessi
  - Sempre attivi
  - Sempre più autonomi
- Ne derivano alcune **proprietà** del paradigma ad agenti:
  - Situatedness
  - Openness
  - Locality in control
  - Locality in interaction
- Il paradigma **Object-Oriented** ha invece alcuni limiti:
  - Il flusso di controllo è centralizzato a design-time
  - I sistemi costruiti sono predicibili
  - Non hanno una corrispondenza uno ad uno con elementi della natura

Cos'è un agente?

"Un agente è un'entità computazionale autonoma che incapsula il suo flusso di controllo e un criterio per auto-governarsi."

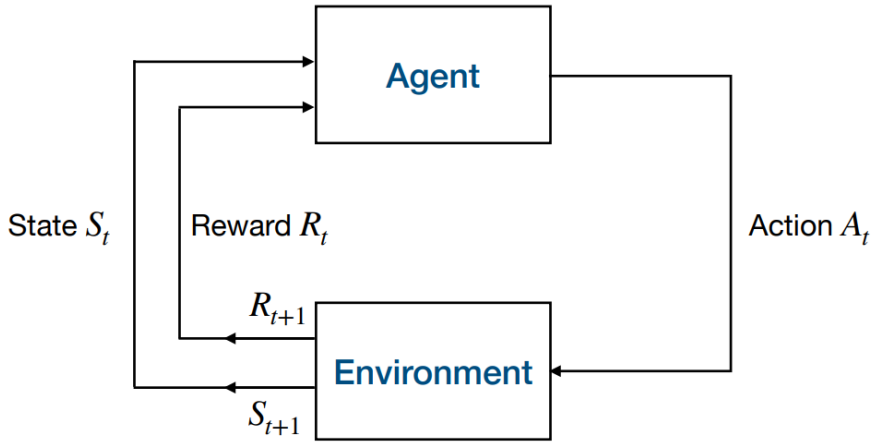
Caratteristiche:

- **Autonomia**
- **Reattività ai cambiamenti**
- **Proattività**
- **Flusso di controllo incapsulato**
- **Socialità**

  "Gli strong-agents hanno dei componenti aggiuntivi come: desideri, goal, piani, conoscenza, mobilità, cooperazione, apprendimento, adattamento, . . . "

Reinforcement Learning

- Tecnica di apprendimento automatico, per rinforzo
- Concetti chiave: **policy**, **reward**, **funzione valore**, **modello dell'ambiente**
- L'ambiente può essere descritto formalmente: **Markov Decision Process**
- Algoritmi **model-based** Vs **model-free**
- Ulteriori misure: **Q-function** ed **equazione di Bellman**



Multi Agent Reinforcement Learning

"Un gruppo di agenti **impara insieme** *la miglior policy per massimizzare una funzione di reward a lungo termine."*

- Molteplici entità che, con un certo grado di autonomia, prendono decisioni ed interagiscono fra loro all'interno di un ambiente
- Ad ogni agente viene assegnato un compito da perseguire (**cooperativo**, **competitivo** o **misto**)
- Esprimere in modo programmatico le skills di un determinato agente sarebbe troppo complesso
- Il sistema viene addestrato mediante Reinforcement Learning

MARL: Environment

Comunemente si usa l'astrazione dei **Markov Games**, anche detti *Stochastic Game* $\mathcal{S}$ [1]

- $\mathcal{S}$ è una tupla $< N, S, \{A^i\}, P, \{R^i\} >$ con $i \in 1..N$
- N è il numero di agenti
- S è lo stato globale dell'environment
- A^i è lo spazio delle azioni del i-esimo agente
  - Lo spazio globale viene definito come $\mathbb{A} = A^1 \times A^2 \times .... \times A^N$
- $P : S \times \mathbb{A} \to \mathcal{P}(S)$ è la funzione che descrive i passaggi fra uno stato e l'altro
  - $P(S)$ è una distribuzione di probabilità discreta che associa ad ogni stato $s \in S$ una probabilità
- $R^i : S \times \mathbb{A} \times S \to \mathbb{R}$ è la funzione di reward

MARL: Esempio

Stochastic Game $\mathcal{S}$

- $N = 2$
- $A^1 = A^2 = \{\text{Paper, Rock, Scissor}\}$
- $S = \{\ \}$
- $R = \begin{array}{c} \\ \textit{Rock} \\ \textit{Paper} \\ \textit{Scissor} \end{array} \begin{array}{c} \begin{array}{ccc} \textit{Rock} & \textit{Paper} & \textit{Scissor} \end{array} \\ \begin{bmatrix} 0,0 & -1,1 & 1,-1 \\ 1,-1 & 0,0 & -1,1 \\ -1,1 & 1,-1 & 0,0 \end{bmatrix} \end{array}$

MARL: Challenges

In letteratura sono presenti ancora numerose challenge riguardo ai sistemi MARL [2]

- Environment parzialmente osservabili
- Non-stazionarità
- Comunicazione
- Coordinazione
- Credit assignment problem
- Scalabilità

MARL: Esempi di successo

- **OpenAI - "Hide and Seek"** [3]
  - Due tipologie di agenti: *hiders* and *seekers*
  - Oggetti ed ostacoli posizionati randomicamente con cui gli agenti interagiscono
  - Autocurriculum: gli agenti apprendono behaviour e competono tra loro
  - domain-specific intelligent tests
- **DeepMind - "Capture the flag"** [4]
  - Due team di agenti che competono in una mappa di gioco e hanno l'obiettivo di rubare la bandiera dell'avversario e di proteggere la propria
  - Elementi di successo:
    - attivazione di neuroni che codificano gli stati più significativi del gioco
    - utilizzo della memoria
    - visual attention
  - Confronto con le capacità umane

Tecnologie studiate: PettingZoo (1/3)

- Libreria[5] che fornisce un insieme di **environment** a cui possono interfacciarsi sistemi **multi-agente**
- Ispirata alla libreria *Gym*[6], standard per il paradigma *single-agent*
- Ha introdotto un nuovo modello matematico di gioco **Agent Environment Cycle (AEC)** che sfrutta l'elemento *Environment Agent* e apporta notevoli vantaggi

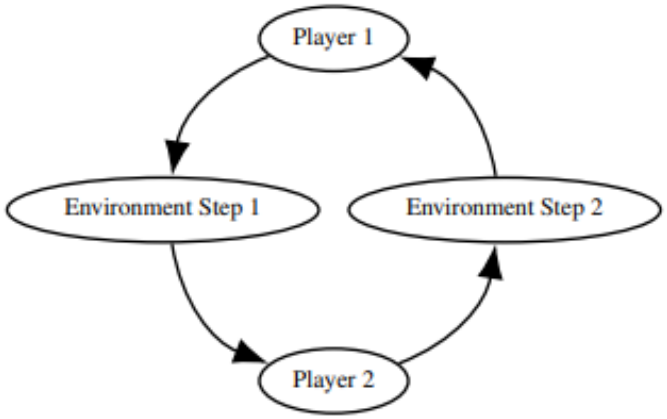


Figure: Diagramma del modello AEC per il gioco degli scacchi

Tecnologie studiate: PettingZoo (2/3)

Mette a disposizione la possibilità di sviluppare un environment personalizzato, che rappresenta il proprio dominio di interesse.

API Custom Environment

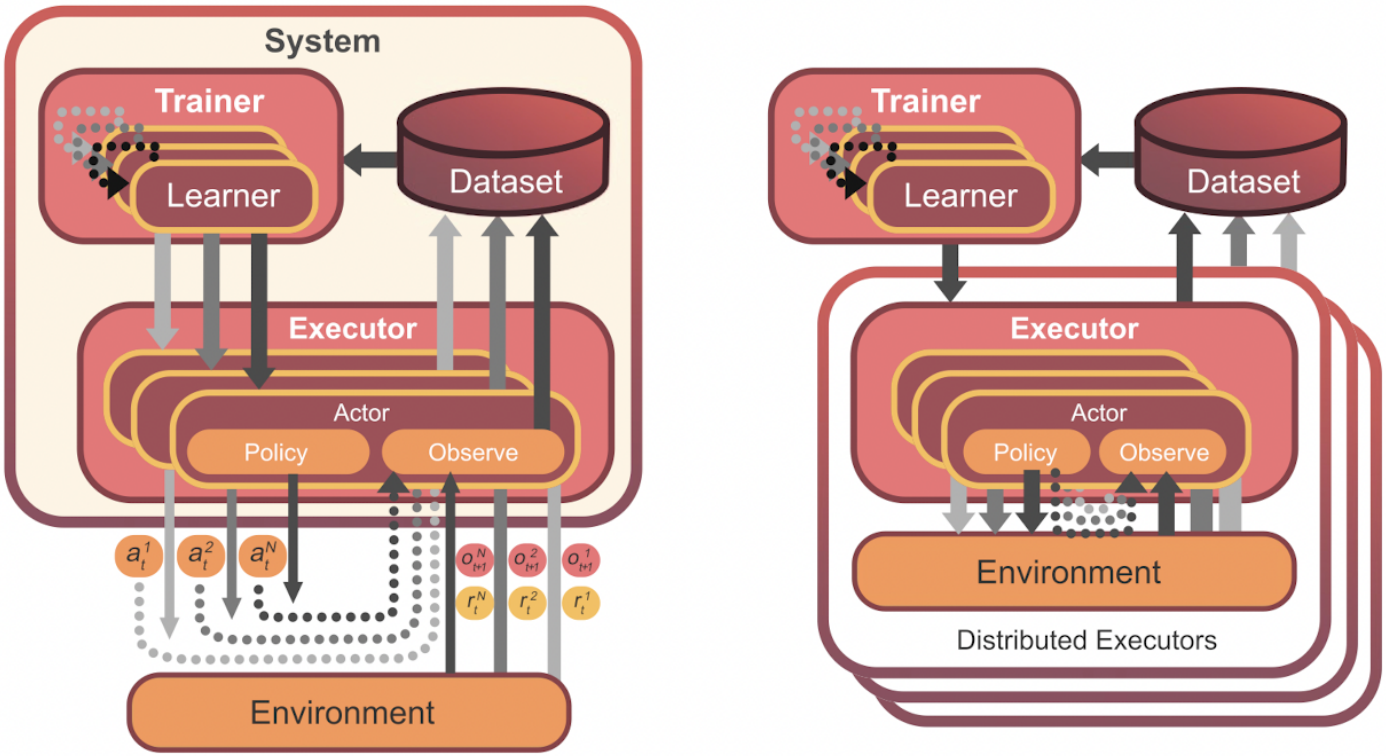
```
def step()
def render()
def close()
def reset()
def agent_iter()
def last()
def agents()
def rewards()
def observation_space()
def action_space()
def observe(agent)
def state()
def agent_selection()
```

Tecnologie studiate: PettingZoo (3/3)

- Per testare questa funzionalità è stato implementato un environment di tipo competitivo
- Sono presenti due agenti: *Thief* e *Policeman*
- Questi agenti sono posizionati in uno spazio 2D
- Entrambi hanno a disposizione cinque azioni possibili: UP, DOWN, LEFT, RIGHT e NONE
- Il goal dell'agente Thief è incrementare la sua distanza dall'agente Policeman, mentre il goal dell'agente Policeman è ridurre la sua distanza dall'agente Thief

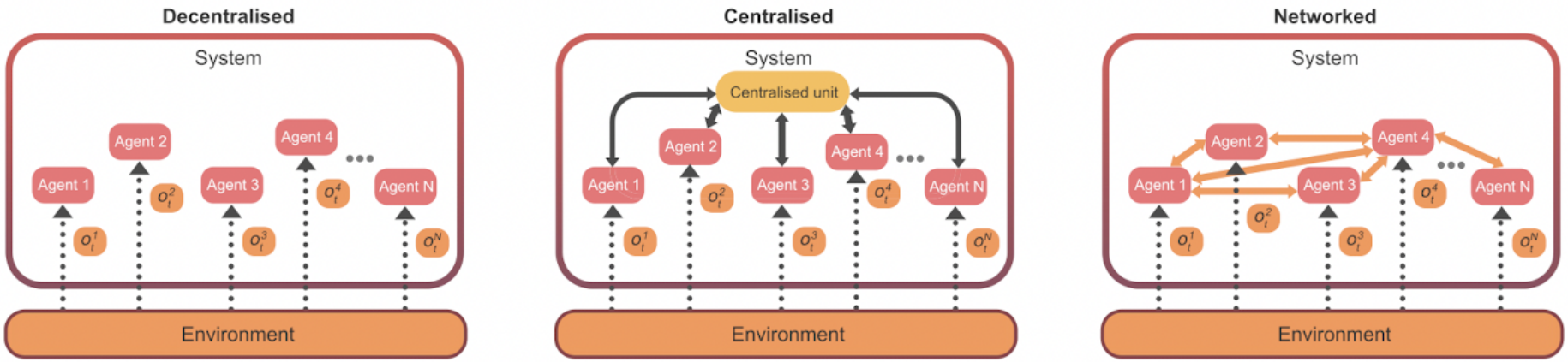
Tecnologie studiate: Mava (1/2)

- Framework di ricerca [7] sviluppato per permettere di **costruire sistemi MARL** in modo scalabile
- Tre componenti di base: **Executor**, **Trainer** e **Dataset**



Tecnologie studiate: Mava (2/2)

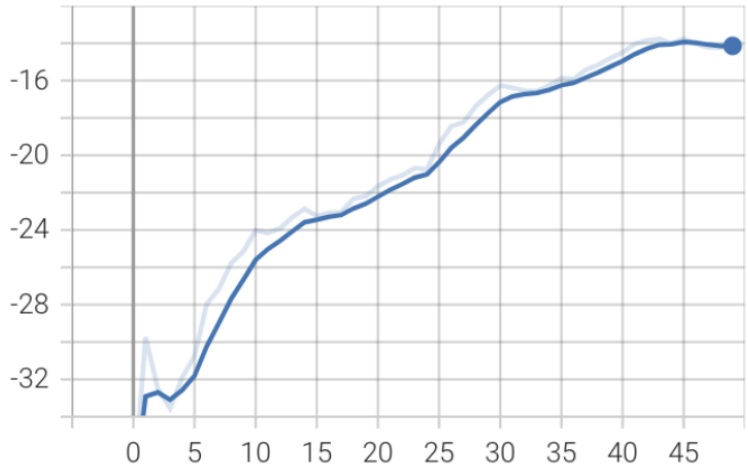
Un System può essere implementato seguendo diverse architetture:



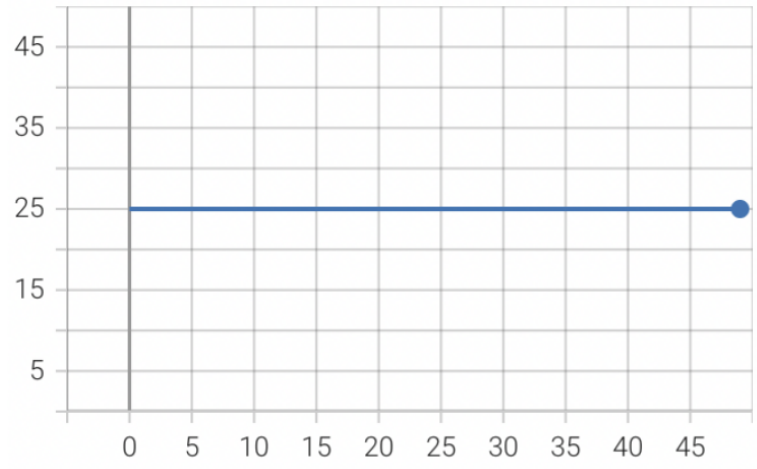
Esperimenti (1/2)

- Sono state utilizzate le tecnologie:
  - **Mava** per definire il sistema multi agente e la fase di training
  - **PettingZoo** per la selezione dell'environtment
- Metriche utilizzate:
  - MeanEpisodeReturn
  - MeanEpisodeLength
  - StdEpisodeReturn
  - MeanStepsPerSecond
- Sono stati utilizzati diversi environment provenienti dalla famiglia Multi Particle Environments (MPE) [8] [9]
  - Sia di tipo cooperativo che competitivo
  - Alcuni esempi: Simple Adversary, Simple Crypto, Simple Spread, Simple Speaker-Listener, Simple World Comm

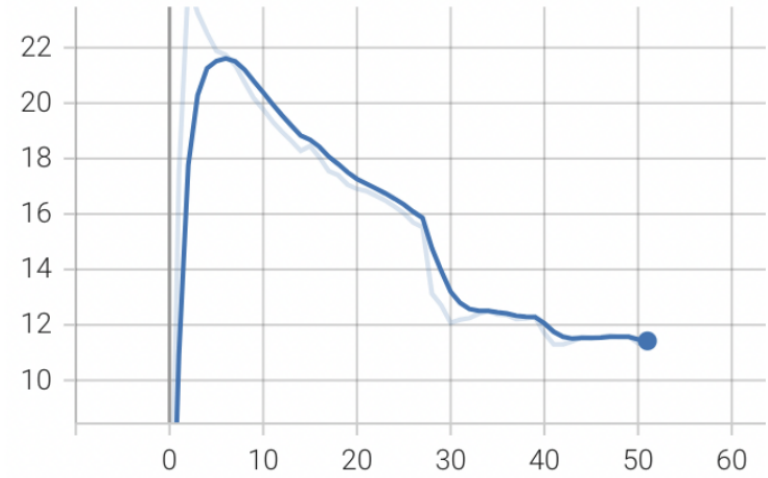
Esperimenti (2/2)



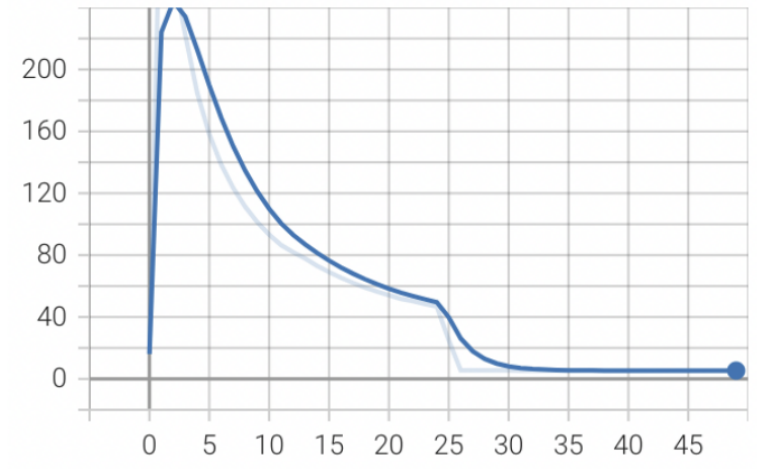
(a) Mean Episode Return



(b) Mean Episode Length



(c) Std Episode Return



(d) Mean Step Per Second

Figure: Simple speaker-listener environment

Conclusioni

- **PettingZoo** è un framework flessibile e stabile
  - Molto ricco di environment pre-implementati
  - API molto semplice per implementare un proprio environment
- **Mava** ha notevoli potenzialità ma presenta dei limiti
  - La documentazione presente è abbastanza scarsa
  - Negli ultimi mesi è stato effettuato un passaggio da TensorFlow a Jax, questo ha portato alcune feature a non funzionare correttamente
- I **sistemi MARL** hanno notevoli potenzialità ma sono usati ancora come black-box
  - Le tecniche di *explainability* e *interpretability* sono ancora immature e non molto utilizzate
  - Potrebbe essere interessante vertere verso un mix di tecniche di machine learning con modelli come il BDI

Bibliografia

[1] Michael L. Littman.
Markov games as a framework for multi-agent reinforcement learning.
In William W. Cohen and Haym Hirsh, editors, *Machine Learning Proceedings 1994*, pages 157–163. Morgan Kaufmann, San Francisco (CA), 1994.

[2] Sven Gronauer and Klaus Diepold.
Multi-agent deep reinforcement learning: a survey.
Artificial Intelligence Review, 55(2):895–943, 2022.

[3] Bowen Baker, Ingmar Kanitscheider, Todor Markov, Yi Wu, Glenn Powell, Bob McGrew, and Igor Mordatch.
Emergent tool use from multi-agent autocurricula, 2019.

[4] Max Jaderberg, Wojciech M. Czarnecki, Iain Dunning, Luke Marris, Guy Lever, Antonio Garcia Castañeda, Charles Beattie, Neil C. Rabinowitz, Ari S. Morcos, Avraham Ruderman, Nicolas Sonnerat, Tim Green, Louise Deason, Joel Z. Leibo, David Silver, Demis Hassabis, Koray Kavukcuoglu, and Thore Graepel.
Human-level performance in 3d multiplayer games with population-based reinforcement learning.
Science, 364(6443):859–865, 2019.

[5] Justin K. Terry, Benjamin Black, Ananth Hari, Luis S. Santos, Clemens Dieffendahl, Niall L. Williams, Yashas Lokesh, Caroline Horsch, and Praveen Ravi.
Pettingzoo: Gym for multi-agent reinforcement learning.
CoRR, abs/2009.14471, 2020.

[6] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba.
Openai gym.
CoRR, abs/1606.01540, 2016.

[7] Arnu Pretorius, Kale ab Tessera, Andries P. Smit, Kevin Eloff, Claude Formanek, St John Grimbly, Siphelele Danisa, Lawrence Francis, Jonathan Shock, Herman Kamper, Willie Brink, Herman Engelbrecht, Alexandre Laterre, and Karim Beguir.
Mava: A research framework for distributed multi-agent reinforcement learning.
arXiv preprint arXiv:2107.01460, 2021.

[8] Igor Mordatch and Pieter Abbeel.
Emergence of grounded compositional language in multi-agent populations.
arXiv preprint arXiv:1703.04908, 2017.

[9] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch.
Multi-agent actor-critic for mixed cooperative-competitive environments.
Neural Information Processing Systems (NIPS), 2017.