A. Experimental Setup

`RoboSumo` Environment

To limit the scope of our study, we restrict agent morphologies to only 4-leg robots. During the game, observations of each agent were represented by a 120-dimensional vector comprised of positions and velocities of its own body and positions of the opponent's body; agent's actions were 8-dimensional vectors that represented torques applied to the corresponding joints.

Network Architecture

Agent policies are parameterized as multi-layer perceptrons (MLPs) with 2 hidden layers of 90 units each. For the embedding network, we used another MLP network with 2 hidden layers of 100 units each to give an embedding of size 100. For the conditioned policy network we also reduce the hidden layer size to 64 units each.

Policy Optimization

For learning the population of agents, we use the distributed version of PPO algorithm as described in (Al-Shedivat et al., 2018) with 2×10^{-3} learning rate, $\epsilon = 0.2$, 16,000 time steps per update with 6 epochs 4,000 time steps per batch.

Training

For our analysis, we train a diverse collection of 25 agents, some of which are trained via self-play and others are trained in pairs concurrently, forming a clique agent-interaction graph.

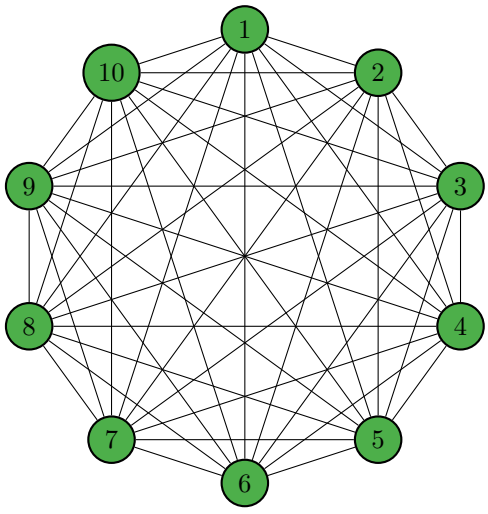

Figure 7: An example clique agent interaction graph with 10 agents.

`ParticleWorld` Environment

The overall continuous observation and discrete action space for the speaker agents are 3 and 7 dimensions respectively. For the listener agents, the observation and action spaces are 15 and 5 dimensions respectively.

Network Architecture

Agent policies and shared critic (*i.e.*, a value function) are parameterized as multi-layer perceptrons (MLPs) with 2 hidden layers of 64 units each. The observation space for the speaker is small (3 dimensions), and a small embedding of size 5 for the listener policy gives good performance. For the embedding network, we again used an MLP with 2 hidden layers of 100 units each.

Policy Optimization

For learning the initial population of listener and agent policies, we use multiagent deep deterministic policy gradients (MADDPG) as the base algorithm (Lowe et al., 2017). Adam optimizer (Kingma & Ba, 2015) with a learning rate of 4×10^{-3} was used for optimization. Replay buffer size was set to 10^6 timesteps.

Training

We first train 28 speaker-listener pairs using the MADDPG algorithm. From this collection of 28 speakers, we train another set of 28 listeners, each trained to work with a speaker pair, forming a bipartite agent-interaction graph. We choose the best 14 listeners for later experiments.

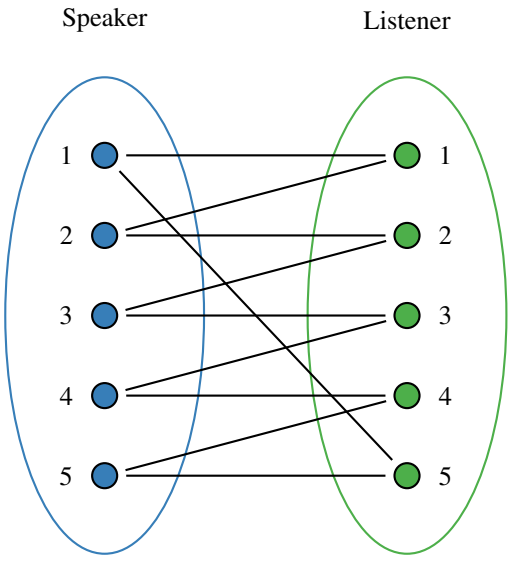

Figure 8: An example bipartite agent interaction graph with 5 speakers and 5 listeners.