Published in the proceedings of the Workshop on "Structure & Priors in Reinforcement Learning" at ICLR 2019

FEUDAL MULTI-AGENT HIERARCHIES FOR COOPERATIVE REINFORCEMENT LEARNING

Sanjeevan Ahilan
Gatsby Computational Neuroscience Unit
University College London
London, United Kingdom
ahilan@gatsby.ucl.ac.uk

Peter Dayan
Max Planck Institute for Biological Cybernetics
72076 Tübingen, Germany
dayan@tuebingen.mpg.de

ABSTRACT

We investigate how reinforcement learning agents can learn to cooperate. Drawing inspiration from human societies, in which successful coordination of many individuals is often facilitated by hierarchical organisation, we introduce Feudal Multi-agent Hierarchies (FMH). In this framework, a 'manager' agent, which is tasked with maximising the environmentally-determined reward function, learns to communicate subgoals to multiple, simultaneously-operating, 'worker' agents. Workers, which are rewarded for achieving managerial subgoals, take concurrent actions in the world. We outline the structure of FMH and demonstrate its potential for decentralised learning and control. We find that, given an adequate set of subgoals from which to choose, FMH performs, and particularly scales, substantially better than cooperative approaches that use a shared reward function.

1 INTRODUCTION

In cooperative multi-agent reinforcement learning, simultaneously-acting agents must learn to work together to achieve a shared set of goals. A straightforward approach is for each agent to optimise a global objective using single-agent reinforcement learning (RL) methods such as Q-Learning (Watkins & Dayan, 1992) or policy gradients (Williams, 1992; Sutton et al., 2000). Unfortunately this suffers various problems in general.

First, from the perspective of any one agent, the environment is non-stationary. This is because as other agents learn, their policies change, creating a partially unobservable influence on the effect of the first agent's actions. Second, agents must learn to coordinate effectively, which can be difficult in environments in which many optimal equilibria exist (Matignon et al., 2012). Third, multi-agent methods scale poorly – the effective state space grows exponentially with the number of agents. Optimising a global objective also becomes challenging at scale, as it becomes difficult to assign credit to each agent (Wolpert & Tumer, 2002).

A clue for meeting this myriad of challenges may lie in the way in which human and animal societies are hierarchically structured. In particular, even in broadly cooperative groups, it is frequently the case that different individuals agree to be *assigned* different objectives which they work towards for the benefit of the collective. For example, members of a company typically have different roles and responsibilities. They will likely report to managers who define their objectives, and they may in turn be able to set objectives to more junior members. At the highest level, the CEO is responsible for the company's overall performance.

Inspired by this idea, and prior work in the single-agent domain (Dayan & Hinton, 1993; Barto & Mahadevan, 2003; Vezhnevets et al., 2017), we propose a novel approach for the multi-agent domain, which organises many, simultaneously acting agents into a managerial hierarchy. Whilst typically all agents in a cooperative task seek to optimise a shared reward, in Feudal Multi-agent Hierarchies (FMH) we instead only expose the highest-level manager to this 'task' reward. The manager must therefore learn to solve the principal-agent problem (Jensen & Meckling, 1976) of communicating subgoals, which define a reward function, to the worker agents under its control. Workers learn to satisfy these subgoals by taking actions in the world and/or by setting their own subgoals for workers immediately below them in the hierarchy.

We outline a method to implement FMH for concurrently acting, communicating agents. Our training is fully decentralised (multi-agent observations and actions are not provided to a centralised system) in contrast to methods which rely on centralised training of decentralised policies (Lowe et al., 2017; Foerster et al., 2018). Our approach pre-specifies appropriate positions in the hierarchy as well as a mapping from the manager's choice of communication to the workers' reward functions. We conduct a range of experiments that highlight the advantages of FMH. In particular, we show its ability to address non-stationarity during training, as managerial reward renders the behaviour of workers more predictable. We also demonstrate FMH's ability to scale to many agents and many possible goals, as well as to enable effective coordination amongst workers. It performs substantially better than both decentralised and centralised approaches.

2 Methods

2.1 Hierarchies

Agents in FMH belong to a hierarchy corresponding to a rooted directed tree, with the highest level manager as the root node and each worker reporting to only a single manager. For simplicity, we focus on two-level hierarchies. Appropriate assignment of agents to positions in a hierarchy depend on the varied observation and action capabilities of the agents. Agents exposed to key information associated with global task performance are likely to be suited to the role of manager, whereas agents with more narrow information, but which can act to achieve subgoals are more naturally suited to being workers. In our experiments, we assign the positions of agents directly, to focus on demonstrating the resulting effectiveness of the hierarchy. However, extensions in which the hierarchy is itself learned would be interesting.

2.2 Goal-Setting

In FMH, managers set goals for workers by defining their rewards. To arrange this, managers communicate a special kind of message to workers that *defines the workers' reward functions* according to a specified mapping. For example, if there are three objects in a room, we might allow the manager to select from three different messages, each defining a worker reward function proportional to the negative distance from a corresponding object. Our experiments explore variants of this scheme, imposing the constraint that the structure of the reward function remain fixed whilst the target is allowed to change. In this context and task, messages therefore correspond to 'goals' (or equivalently 'subgoals') requiring approach to different objects. The manager must learn to communicate these goals judiciously in order to solve complex tasks. In turn, the workers must learn how to act in the light of the resulting managerial reward, in addition to immediate rewards (or costs) they might also experience.

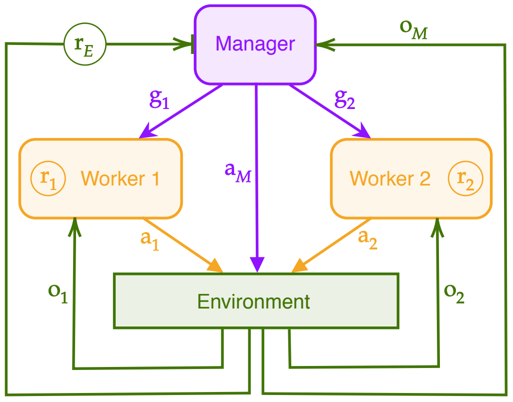

Figure 1: An example of a Feudal Multi-agent Hierarchy with one manager agent and two worker agents. Worker rewards are goal-dependent and computed locally, the manager's reward is provided by the environment.

Since a communicated goal is simply another type of action for the manager, our approach is consistent with the formalism of a (partially-observable) Markov game, for which the reward for agent i is defined as $r_i : \mathcal{S} \times \mathcal{A}_1 \times \ldots \times \mathcal{A}_n \to \mathbb{R}$. However, the worker's reward need not be treated as part of the environment. For instance, each worker agent i could compute its own reward locally $r_i : \mathcal{O}_i \times \mathcal{A}_i \to \mathbb{R}$, where $\mathcal{O}_i$ includes the observed goal from the manager. We illustrate this 'worker-computed' interpretation in Fig. 1.

2.3 Pretraining and Temporally Extended Subgoals

In FMH we aim to resolve the issue of non-stationarity by allowing the manager to determine the reward functions of workers, compelling them towards more predictable behaviour. However, at the starting point of learning workers will not yet have learned how to satisfy subgoals. We would therefore expect a manager to underestimate the value of the subgoals it selects early in training, potentially leading it sub-optimally to discard subgoals which are harder for the workers to learn.

We address this practically in two steps. First, we introduce a bottom-up ‘pretraining’ procedure, in which we initially train the workers before training the manager. Although the manager is not trained during this period, it still acts in the multi-agent environment and sets subgoals for the worker agents. If the set of possible subgoals is sufficiently compact, this will enable workers to gain experience of each potential subgoal. Secondly, to prevent the manager from vacillating between subgoals, we enforce a communication-repeat heuristic for the manager such that goal-setting is temporally extended. This ensures that the manager not only tries a variety of subgoals but also *commits* to them long enough that they have any hope of being at least partially achieved.

3 Experiments and Results

We used the multi-agent particle environment[1] as a framework for conducting experiments to test the potential of FMH to address the issues of non-stationarity, scalability and coordination, comparing against alternatives. Our RL agents have both an actor and a critic, each corresponding to a feedforward neural network.

3.1 Cooperative Communication

We first experiment with an environment implemented in Lowe et al. (2017) called ‘Cooperative Communication’ (Figure 2a), in which there are two cooperative agents, one called the ‘speaker’ and the other called the ‘listener’, placed in an environment with many landmarks of differing colours. On each episode, the listener must navigate to a randomly selected landmark; and in the original problem, both listener and speaker obtain reward proportional to the negative distance of the listener from the correct target. However, whilst the listener observes its relative position from each of the differently coloured landmarks, it does not know to which landmark it must navigate. Instead, the colour of the target landmark can be seen by the speaker, which cannot observe the listener and is unable to move. The speaker can however communicate to the listener at every time step, and so successful performance on the task corresponds to the speaker enabling the listener to reach the correct target. We also note that, although reward is provided during the episode, it is used only for

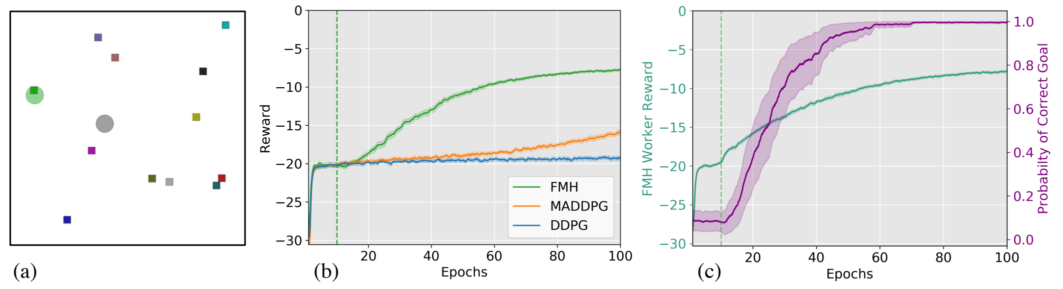

Figure 2: Cooperative Communication with 1 listener and 12 landmarks. (a) The speaker (grey circle) sees the colour of the listener (green circle) , which indicates the target landmark (green square). Here there are 12 landmarks and the agent trained using FMH has successfully navigated to the correct landmark. (b) FMH substantially outperforms MADDPG and DDPG. The dotted green line indicates the end of pretraining for FMH. (c) FMH worker reward and the probability of the manager correctly assigning the correct target to the worker.

[1]https://github.com/openai/multiagent-particle-envs

Number of		Final Reward		
Listeners	Landmarks	FMH	MADDPG	DDPG
1	3	$\mathbf{-6.63 \pm 0.05}$	$\mathbf{-6.58 \pm 0.03}$	-14.26 ± 0.07
1	6	-6.91 ± 0.07	$\mathbf{-6.69 \pm 0.06}$	-18.10 ± 0.07
1	12	$\mathbf{-7.79 \pm 0.06}$	-15.96 ± 0.09	-19.32 ± 0.11
3	6	$\mathbf{-7.10 \pm 0.04}$	-11.13 ± 0.03	-18.90 ± 0.05
5	6	$\mathbf{-7.17 \pm 0.03}$	-18.47 ± 0.04	-19.73 ± 0.06
10	6	$\mathbf{-8.96 \pm 0.03}$	-19.80 ± 0.06	-21.19 ± 0.03

Table 1: Performance of FMH, MADDPG and DDPG for versions of Cooperative Communication with different numbers of listeners and landmarks. Final reward is determined by training for 100 epochs and evaluating the mean reward per episode in the final epoch.

training agents and not directly observed, which means that agents cannot simply learn to follow the gradient of reward.

Although this task seems simple, it is challenging for many RL methods. Conventional approaches using decentralised training perform poorly in this task (Lowe et al., 2017), including the single-agent method of Deep Deterministic Policy Gradients (DDPG) (Lillicrap et al., 2015) with a global reward function, which we use as a baseline for the decentralised apprach. We also compare our results to multi-agent DDPG (MADDPG) (Lowe et al., 2017), which combines DDPG with centralised training. MADDPG was found to perform strongly on Cooperative Communication with three landmarks, far exceeding the performance of DDPG.

For our method, FMH, we also utilised DDPG, but reverted to the more generalizable decentralized training that was previously ineffective. Crucially, we assigned the speaker the role of manager and the listener the role of worker. The speaker can therefore communicate messages which correspond to the subgoals of the different coloured landmarks. The listener is not therefore rewarded for going to the correct target but is instead rewarded proportional to the negative distance from the speaker-assigned target. The speaker itself is rewarded according to the original problem definition, the negative distance of the listener from the true target.

We investigated in detail a version of Cooperative Communication with 12 possible landmarks (Figure 2a). FMH performed significantly better than both MADDPG and DDPG (Figure 2b) over a training period of 100 epochs (each epoch corresponds to 1000 episodes). For FMH, we pretrained the worker for 10 epochs and enforced extended communication over 8 time steps (each episode is 25 time steps). In Figure 2c, the left axis shows the reward received by the FMH worker (listener) over training, which increases again immediately after pretraining concludes. This improvement in worker performance enables the manager to assign goals correctly, with the rise in the probability of correct assignment occurring shortly thereafter (right axis), then reaching perfection.

3.1.1 Scaling to many agents

We next scaled Cooperative Communication to include many listener agents. At the beginning of an episode each listener is randomly assigned a target out of all the possible landmarks and the colour of each listener's target is observed by the speaker. The speaker communicates a single message to each listener at every time step. To allow for easy comparison with versions of Cooperative Communication with only one listener, we normalised the reward by the number of listeners.

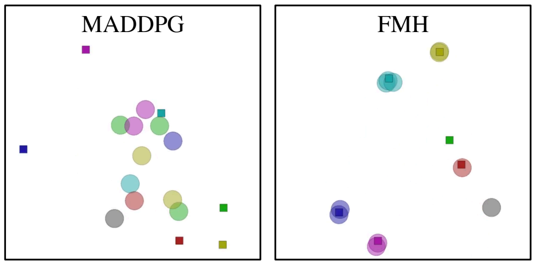

Figure 3: Scaling Cooperative Communication to 10 listeners with 6 landmarks - final time step on example episode.

In Table 1 we show the performance of FMH, MADDPG and DDPG for variants of Cooperative Communication with different numbers of listener agents and landmarks. Consider the version with 6 landmarks, which we found that

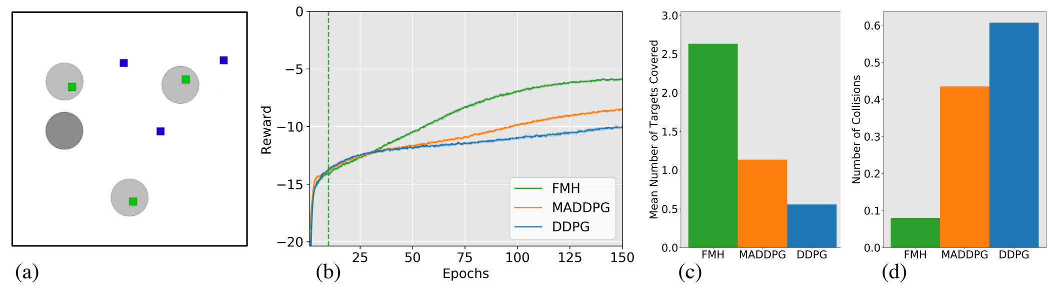

Figure 4: Cooperative Coordination (a) Three listeners (light grey agents) must move to cover the green landmarks whilst ignoring the blue landmarks. However, only the speaker (dark grey agent) can see the landmarks' colours; it communicates with the listeners at every time step. In this example, FMH agents have successfully coordinated to cover the three correct targets. (b) FMH performs significantly better than MADDPG and DDPG. (c) Agents trained using FMH cover on average more targets, by the end of the episode, than MADDPG and DDPG (d) Agents trained using FMH avoid collisions more effectively than MADDPG and DDPG over the duration of an episode.

MADDPG could solve with a single listener within 100 epochs (unlike for 12 landmarks). On increasing the number of listeners up to a maximum of 10, we found that FMH scales much better than MADDPG; FMH was able to learn an effective policy with 5 listener agents whereas MADDPG could not. Further, FMH even scaled to 10 listeners (Figure 3).

3.2 Cooperative Coordination

We then designed a task to test coordination called 'Cooperative Coordination' (Figure 4a). In this task, there are 6 landmarks. At the beginning of each episode 3 landmarks are randomly selected to be green targets and 3 blue decoys. A team of 3 agents must navigate to cover the green targets whilst ignoring the blue decoys, but they are unable to see the colours of the landmarks. A fourth agent, the speaker, can see the colours of the landmarks and can send messages to the other agents (but cannot move). All agents can see each others' positions and velocities, are large in size and face penalties if they collide with each other. The task shares aspects with 'Cooperative Navigation' from Lowe et al. (2017) but is considerably more complex due to the greater number of potential targets and the hidden information.

We apply FMH to this problem, assigning the speaker agent the role of manager. One consideration is whether the manager, the worker, or both should receive the negative penalties due to worker collisions. Here we focus on the case in which the manager only concerns itself with the 'task' reward function. Penalties associated with collisions are therefore experienced only by the workers themselves, which seek to avoid these whilst still optimising the managerial objective.

In Figure 4b we compare the performance of FMH to MADDPG and DDPG. As with Cooperative Communication, FMH does considerably better than both after training for 150 epochs. This is further demonstrated when we evaluate the trained policies over a period of 10 epochs: Figure 4c shows the mean number of targets covered by the final time step of the episode, for which FMH more than doubles MADDPG. Figure 4d shows the mean number of collisions (which are negatively rewarded) during an episode. FMH collides very rarely whereas MADDPG and DDPG collide over 4 times more frequently.

4 Conclusion

We have shown how cooperative multi-agent problems can be solved efficiently by defining a hierarchy of agents. Our hierarchy was reward-based, with a manager agent optimising the overall task objective whilst setting goals for multiple simultaneously-operating worker agents. Our method, called FMH, was trained in a decentralised fashion and showed considerably better scaling and coordination than both centralised and decentralised methods that used a shared reward function.

Our work has much scope for further exploration. In particular, specifying goals using a learned hidden representation, as in Vezhnevets et al. (2017) would likely be of value for problems in which defining a set of subgoals is challenging, such as learning directly from pixels. More generally, addressing how agents can learn and agree upon a language for goal-setting would be most important. Finally, we did not explore how complex hierarchical structures should be formed. Given their apparent effectiveness, this would provide a promising direction for further research.

5 ACKNOWLEDGEMENTS

We would like to thank Heishiro Kanagawa, Jorge A. Menendez and Danijar Hafner for helpful comments on a draft version of the manuscript. Sanjeevan Ahilan received funding from the Gatsby Computational Neuroscience Unit and the Medical Research Council. Peter Dayan received funding from the Max Planck Society.

REFERENCES

Andrew G Barto and Sridhar Mahadevan. Recent advances in hierarchical reinforcement learning. *Discrete event dynamic systems*, 13(1-2):41–77, 2003.

Peter Dayan and Geoffrey E Hinton. Feudal reinforcement learning. In *Advances in neural information processing systems*, pp. 271–278, 1993.

Jakob N Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Michael C Jensen and William H Meckling. Theory of the firm: Managerial behavior, agency costs and ownership structure. *Journal of financial economics*, 3(4):305–360, 1976.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in Neural Information Processing Systems*, pp. 6379–6390, 2017.

Laetitia Matignon, Guillaume J Laurent, and Nadine Le Fort-Piat. Independent reinforcement learners in cooperative markov games: a survey regarding coordination problems. *The Knowledge Engineering Review*, 27(1):1–31, 2012.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pp. 1057–1063, 2000.

Alexander Sasha Vezhnevets, Simon Osindero, Tom Schaul, Nicolas Heess, Max Jaderberg, David Silver, and Koray Kavukcuoglu. Feudal networks for hierarchical reinforcement learning. In *International Conference on Machine Learning*, pp. 3540–3549, 2017.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

David H Wolpert and Kagan Tumer. Optimal payoff functions for members of collectives. In *Modeling complexity in economic and social systems*, pp. 355–369. World Scientific, 2002.

SUPPLEMENTARY MATERIALS

A EXPERIMENTAL RESULTS

A.1 PARAMETER SETTINGS

In all of our experiments, we used the Adam optimizer with a learning rate of 0.001 and $\tau = 0.01$ for updating the target networks. γ was 0.75. The size of the replay buffer was 10^7 and we updated the network parameters after every 100 samples added to the replay buffer. We used a batch size of 1024 episodes before making an update. For our feedforward networks we used two hidden layers with 256 neurons per layer. We trained with 10 random seeds (except otherwise stated).

Hyperparameters were optimised using a line search centred on the experimental parameters used in Lowe et al. (2017). Our optimised parameters were found to be identical except for a lower value of γ (0.75) and of the learning rate (0.001), and a larger replay buffer (10^7). We found these values gave the best performance for both MADDPG and FMH on a version of Cooperative Communication with 6 landmarks evaluated after 50 epochs (an epoch is defined to be 1000 episodes).

A.2 PARAMETER SHARING

We implemented parameter sharing for the decentralised algorithms DDPG and FMH when there were multiple agents of the same type (e.g. identical listeners). To make training results approximately similar to implementations which do not use parameter sharing we restrict updates to a single agent and add experience only from a single agent to the shared replay buffer (amongst those sharing parameters). We find in practice that both approaches give very similar results for FMH, whereas parameter sharing slightly improves the performance of DDPG – we show this for a version of Cooperative Communication with 3 listeners and 6 targets (Figure S1). In general sharing parameters reduces training time considerably, particularly as the number of agents scales.

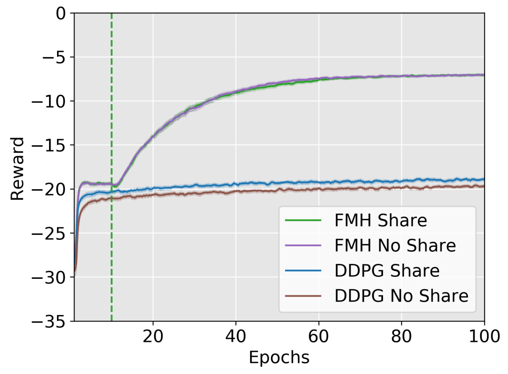

Figure S1: Parameter sharing does not affect performance for FMH but slightly improves DDPG.

A.3 ENTROPY OF COMMUNICATION

We show the change in entropy of communication over the duration of an episode for the various algorithms (Figure S2). Agents are trained on Cooperative Communication with 12 landmarks and a single listener. As the target does not change during the middle of an episode, we expect the entropy to decrease as agents learn.

For FMH, during pretraining, entropy is high as all goals are sampled approximately uniformly (but with enforced extended communication over 8 time steps). However, shortly after pretraining ends the entropy of managerial communication rapidly decreases (which in turn allows the worker to more easily optimise the managerial objective). We also show the entropy for MADDPG which declines more slowly relative to FMH and DDPG.

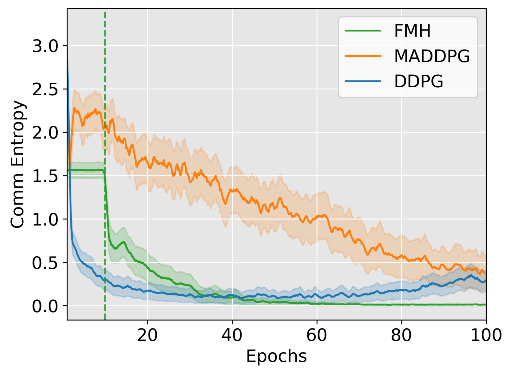

Figure S2: Entropy of managerial communication over the duration of an episode at different stages in training

A.4 Pretraining

Agents trained using FMH were pretrained for 10 epochs (10,000 episodes) across all experiments. During pretraining all agents act in the multi-agent environment. Although all experiences are added appropriately into the replay buffers, the manager does not update its parameters during pretraining whereas the workers do.

We show the benefits of pretraining on Cooperative Communication with 12 landmarks and a single listener agent in Figure S3. We varied the number of epochs allocated for pretraining for a fixed extended communication of 8 time steps. Although it is clear that pretraining helps in this task, performance is still substantially better than MADDPG and DDPG even without it.

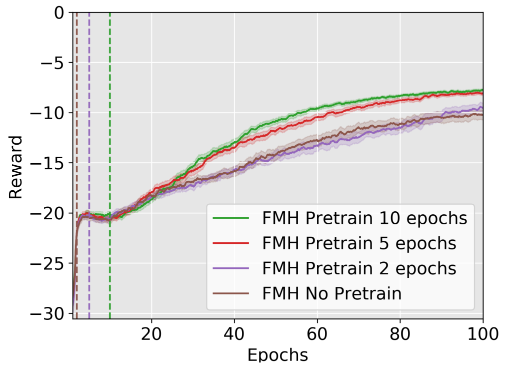

Figure S3: Pretraining improves FMH, although agents can still learn effectively without it. We use extended communication (goal-setting) for 8 time steps.

A.5 Extended Communication

In the main text we introduced the extended goal-setting for FMH. Here we show its benefits, which are substantial even with a single repetition of the manager's chosen communication (Figure S4). We also show that this benefit does not apply to MADDPG, for which communication does not influence the reward functions of other agents.

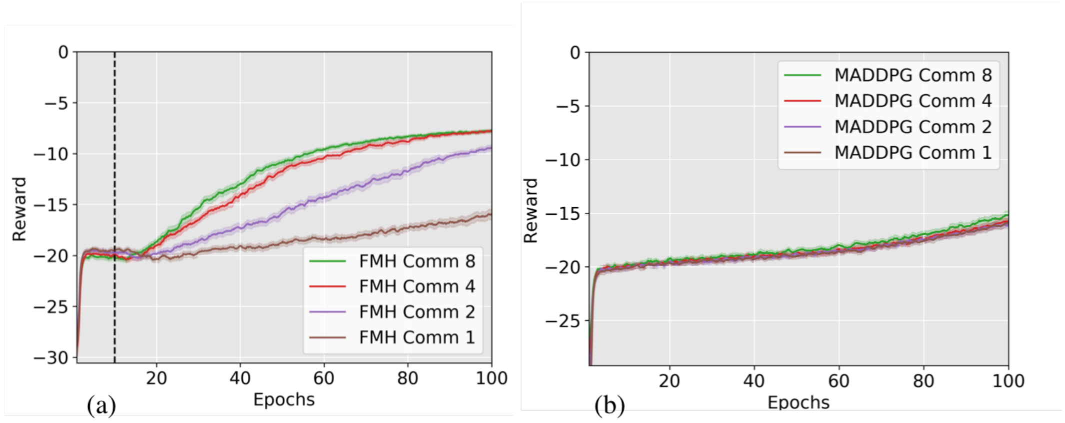

Figure S4: (a) Extended communication improves the performance of DDPG (even for a single repetition) (b) Extended communication does not significantly improve the performance of MADDPG.

A.6 Cooperative Communication with 3 landmarks

For reference we show performance of the various algorithms on Cooperative Communication with 3 landmarks. Both MADDPG and FMH perform well on this task, although MADDPG reaches convergence more rapidly (Figure S5).

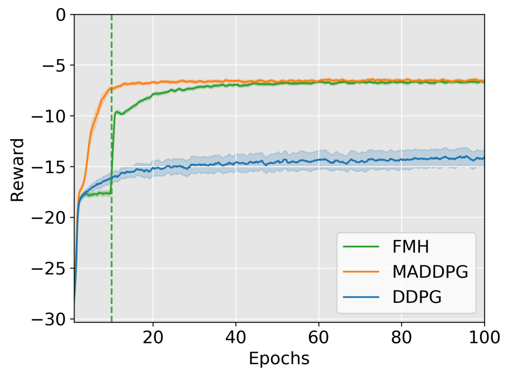

Figure S5: Cooperative Communication with 1 listener and 3 landmarks.

B Environments

B.1 Cooperative Communication

We provide further details on our version of Cooperative Communication (see main text for original description). In general, we keep environment details the same as Lowe et al., including the fact that the manager only has access to the target colour. However, we use negative distance as a reward function rather than negative square distance, as we find this gives better performance for all algorithms. We also scale up the number of coloured landmarks, which we do by taking the RGB values provided in the multi-agent particle environment, $[0.65, 0.15, 0.15]$, $[0.15, 0.65, 0.15]$, $[0.15, 0.15, 0.65]$, and adding 9 more by placing them on the remaining vertices of the corresponding cube and at the centre-point of four of the faces (in RGB space). The particular colour values used for the landmarks influences the performance of RL algorithms as landmarks which have similar colours are harder for the speaker to learn to distinguish.

B.2 Cooperative Coordination

We provide further details on our version of Cooperative Coordination (see main text for original description). The task provides a negative reward of -1 to each agent involved in a collisions. For DDPG and MADDPG this penalty is shared across agents, whereas in FMH only the agents involved in the collision experience this penalty.

We also evaluated performance of trained policies in Figures 4c and 4d with slight modifications to the overall task. In the case of Figure 4c, to ensure that targets were never impossible to achieve by overlapping with the immobile manager, we moved the manager off-screen. For Figure 4d we ensured that agent positions were never initialised in a way such that they would automatically collide (such cases are rare).

C Solving a Coordination Game

We show how our approach might address a popular form of multi-agent task, namely coordination games. These games have multiple good and bad Nash equilibria, and it is necessary to find the former. It is intuitively obvious that appointing one of the agents as a manager might resolve the symmetries inherent in a cooperative coordination game in which agents need to take different actions to receive reward:

		Player Y	
		A	B
Player X	A	$(0, 0)$	$(1, 1)$
	B	$(1, 1)$	$(0, 0)$

This game has two pure strategy Nash equilibria and one mixed strategy Nash equilibrium which is Pareto dominated by the pure strategies. The challenge of this game is for both agents to choose a single Pareto optimal Nash equilibrium, either (A, B) or (B, A).

For a matrix game, we define the feudal approach as allowing Player X, the manager, to specify the reward player Y will receive for its actions. This is a simplification when compared to the more general setting of a Markov game in which the feudal manager can reward not only actions but also achievement of certain states.[2]. In order to specify the reward, we assume that Player X communicates a goal, either g_A or g_B, prior to both players taking their actions. If Player X sends g_A it means that action A is now rewarded for Player Y and action B is not. Player X's rewards are unchanged, and so together this induces the following matrix game:

		Player Y	
		A	B
Player X	A	$(0, 1)$	$(1, 0)$
	B	$(1, 1)$	$(0, 0)$

Action A for player Y is now strictly dominant and so a rational Player Y will always choose it. By iterated elimination of strictly dominated strategies we therefore find the resulting matrix game:

		Player Y
		A
Player X	A	$(0, 1)$
	B	$(1, 1)$

And so a rational Player X will always choose B, resulting in an overall strategy of (B, A) conditioned on an initial communication of g_A. By symmetry, we can see that conditioned on g_B, rational

[2]Typically we prefer the manager to choose distal states as targets rather than actions as this requires the manager to micromanage less and so supports temporal abstraction

players X and Y will play (A, B). The feudal approach therefore allows the manager to flexibly coordinate the pair of agents to either Nash equilibrium. For games involving N players, coordination can be achieved by the manager sending out N-1 goals.