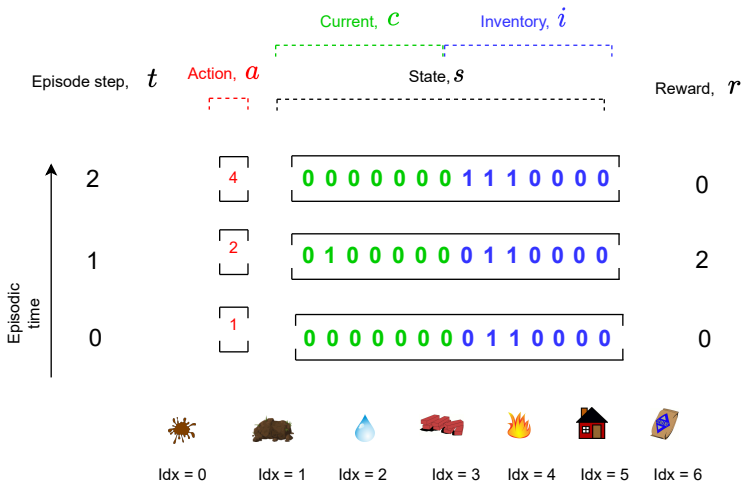

Figure 6: Visualizing actions and states in Wordcraft: we present the first 3 time steps of an episode corresponding to playing the example in Figure 1. This task contains 7 elements, so the action space is a integer with maximum value 7. In the components current c and inventory i, each digit in the vector corresponds to the element with the corresponding index. The initial set includes *Water* and *Earth* (their indexes at $\tau = 0$ in the inventory are non-zero). The agent first picks *Earth* (second index in the action vector). At $t = 1$, *Earth* becomes active in the *Current* vector of the state, the the agent selects *Water* and receives a positive reward. At $t = 2$, *Mud* is created and inserted in the inventory and c is cleared.

This supplementary material provides additional methods, results and discussion, as well as implementation details.

- Section A describes in detail the MDP formulation of Wordcraft;
- Section C contains the pseudocde of SAPIENS;
- Section D explains how we model dynamic social network structures and how their performance varies with their hyper-paramaters;
- Section E provides more information about our experimental setup and results (effect of group size, intra-group and inter-group alignment, robustness to learning hyper-parameters and effect of prioritized experience sharing). We also provide tables and figures for all metrics presented in Section 2.4 and reward plots.
- Section E.7 contains simulations with another testbed, the Deceptive Coins game.

A Details of Wordcraft as a Markov Decision Process

We consider the episodic setting, where the environment resets at the end of each episode and an agent is trained for E_{train} episodes. At each time step t, the agent observes the state s_t and selects an action a_t from a set of possible actions $\mathcal{A}$ according to its policy π^θ, where π^θ is a mapping from states to actions, parameterized by a neural network with weights θ. In return, the agent receives the next state s_{t+1} and a scalar reward r_t. Each DQN agent collects experience tuples of the form $[s_t, a_t, s_{t+1}, r_t]$ in its replay buffer.

Figure 6 offers a visualization of the states and actions encountered during an episode in Wordcraft, where the chosen actions and elements are chosen so as to reproduce the example of Figure 1. In order to solve the innovation task described in Section 2.1 we compute the maximum number of elements a player can craft within horizon T for recipe book $\mathcal{X}_{valid}$ and initial set $\mathcal{X}_0$, which we denote as $|X|$. We, then, encode each element as an integer in $[0, |X|)$. Thus, the action space is

Work	Field	Agent Model	Information type	Task	Dynamic structure?	Main conclusion
(Garnelo et al., 2021)	MARL	DRL	interaction [3]	strategic micro-management (StarCraft(Vinyals et al., 2017))	Yes	Topologies with cycles encourage strategic diversity and dynamic ones perform robustly across tasks
(Adjodah et al., 2019)	Dec-RL	DRL	rewards, NN weights	continuous control (Mujoco (Todorov et al., 2012))	No	Random topologies outperforms fully-connected ones
(Du et al., 2021)	MARL	DRL	observations	cooperative navigation (Particle World (Lowe et al., 2017))	Yes	Agents choose to communicate when they need to coordinate.
(Dubova et al., 2020)	MARLC [4]	DRL	interaction [1]	coordination game	No	Global connectivity leads to shared and symmetric protocols, while partially-connected groups learn local dialects.
(Fang et al., 2010)	computational cognitive science	belief-majority rule [5]	belief, reward	NK problem [6]	No	Partial connectivity maximizes performance
(Lazer & Friedman, 2007)	computational cognitive science	belief-majority rule [3]	belief, reward	NK type [7]4	No	Partial connectivity maximizes performance
(Cantor et al., 2021)	computational cognitive science	belief-majority rule [3]	belief, reward	innovation	No	Performance depends on both task and group structure, no topology is robustly optimal across tasks.
(Mason & Watts, 2012)	cognitive science	human	action,reward	NK problem [3]	No	Full connectivity maximizes diversity and works best even in complex tasks.
(Mason et al., 2008)	cognitive science	human	action, reward	line search	No	Partial connectivity works best in complex problems
(Derex & Boyd, 2016)	cognitive science	action,reward	innovation	Yes	partial connectivity works best	
(this work)	distributed RL and computational cognitive science	DRL	transition tuples	innovation	yes	Partially-connected structures, especially dynamics ones, perform robustly in different types of innovation tasks

Table 1: A non-comprehensive summary of the literature on the topic of the effect of social network topology on collective search

$\mathcal{A} = [0, |X|)$, with action a_t indicating the index of the currently chosen element. The state s_t contains two sets of information: a binary vector of length $|X|$ with non-zero entries for elements already crafted by the agent within the current episode (we refer to this as inventory i) and another binary vector of length $|X|$ where an index is non-zero if it is currently selected by the agent (we refer to this as current c). An agent begins with an inventory having non-zero element only for the initial set $\mathcal{X}_0$ and an all-zero selection. With the first action a_0, the selected item becomes non-zero in the selection. With the second action, a_1, we check if the combination (a_1, c_0) is valid under the recipe book and, if so, return the newly crafted element (corresponding entry in i becomes non-zero) and the reward. This two-step procedure continues until the end of the episode.

B SUMMARY OF RELATED WORKS

In this appendix we provide a non-comprehensive summary of the literature on the topic of the effect of social network topology on collective search in Table 1, where our objective is to highlight similarities and differences within and across the fields of cognitive science and DRL.

C PSEUDOCODE OF SAPIENS

We present the pseudocode of our proposed algorithm SAPIENS in Algorithm 1. SAPIENS works similarly to an off-policy reinforcement learning algorithm, with the difference that, after each episode, an experience sharing phase takes place between agents that belong in the same group.

Algorithm 1 SAPIENS (Structuring multi-Agent toPology for Innovation through ExperieNce Sharing)

Input: $\mathcal{G}, connectivity, R, p_s, LS$
$\mathcal{G}$.initializeGraph(connecticity)
$\mathcal{I}$.initializeAgent() ▷ Initialize agents
for $i \in \mathcal{I}$ **do**
    I.neighbors= I.formNeighborhood($\mathcal{G}$) ▷ Inform agent about its neighbors
    I.env = initEnv(R) ▷ Create agent's own copy of the environment based on the recipe book
end for
while training not done **do**
    for $i \in \mathcal{I}$ **do** ▷ Loop through each agent
        while episode not done **do**
            a = i.policy() ▷ Choose action
            r, s_{new} = env.step(a)
            i.B.insert($[s, r, a, s_{new}]$)
        end while
        ϵ =random()
        if $\epsilon < p_s$ **then** ▷ Share with probability p_s
            for $j \in$ i.neighbors **do**
                j.B.add(i.B.sample(L)) ▷ Sample random set of experiences of length L
            end for
        end if
        i.train() ▷ Train agent
    end for
end while

D ANALYSIS OF DYNAMIC NETWORK TOPOLOGIES

In the main paper we presented results for a single type of dynamic topolgoy. Here we present another type and analyze how they both behave for different values of their hyper-parameters. The two dynamic topologies are:

- Inspired by graphs employed in human laboratory studies (Derex & Boyd, 2016), we designed graphs where the macro structure of the graph is constant but agents can randomly change their position. In particular, we divide a group of agents into sub-groups of two agents and, at the end of each episode, move an agent to another group with a probability p_v for a duration of T_v episodes (for a visualization see Figure 3). To reduce the complexity of the implementation, we assume that only one visit can take place at a time. In the main paper we employ $p_v = 0.01$ and $T_v = 10$ across conditions and present results with different values in Appendix D, where we refer to this topology as dynamic-Boyd.
- Human behavioral ecology emphasize the importance of periodic variation in human social networks encountered throughout our evolutionary trajectory Wiessner (2014); Dunbar (2014). Due to ecological constraints human groups oscillate between phases of high and low connectivity: low-connectivity phases arise when individuals need to individually collected resources (e.g. day-time hunting) while high-connectivity phases arise when humans are idle and "forced" to be in proximity with others (e.g. fireside chats). Although these high-connectivity phases do not bare a direct evolutionary advantage, they may have played an important role by creating the conditions for the evolution of human language and culture. Inspired by this hypothesis, we have designed dynamic graphs that oscillate between a fully-connected topology that lasts for T_h episodes and a topology without sharing that lasts for T_l episodes. We present results for various values of T_h and T_d of this topology in Appendix D, where we refer to this topology as dynamic-periodic.

In Figure 8, we observe the $\%$ of group success ($S^{\mathcal{G}}$) with the dynamic-Boyd topology for different probabilities of visit (p_v) and visit duration T_v (the sub-group size is 2 in all cases). We note that, due to our implementation choice that a visit can take place only if no other agent is currently on a visit, the visit duration also affects the mixing of the group: longer visits mean that fewer visits will

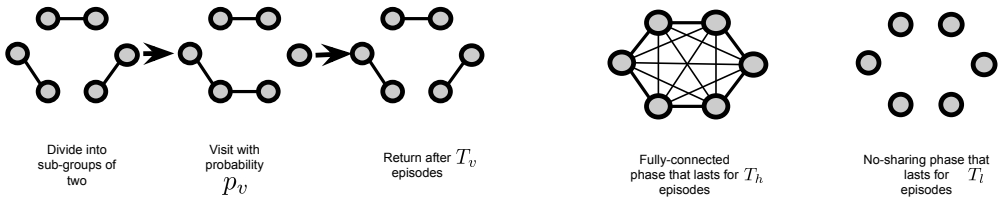

Figure 7: Two types of dynamic topologies: (Left) in the dynamic-Boyd topology the group is divided into sub-groups of two agents and a visit takes place with probability p_v and lasts T_v episodes (Right) In the dynamic-periodic topology the graph oscillates between a phase with a fully-connected topology that lasts for T_h episodes to a phase without sharing that lasts for T_l episodes.

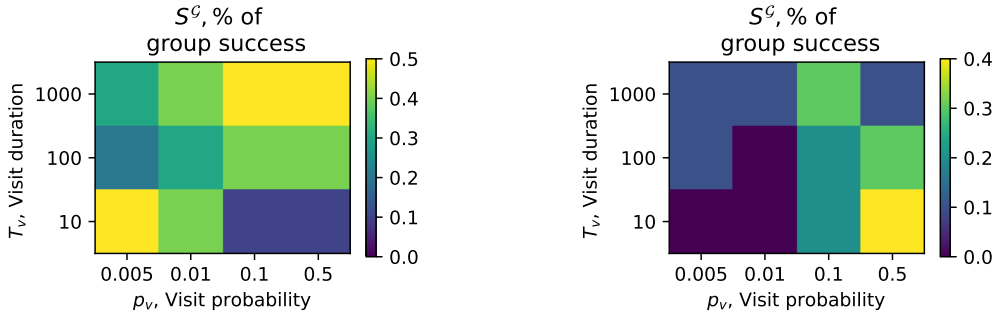

Figure 8: Examining the sensitivity of the dynamic-Boyd topology to its hyper-parameters: % of group success ($S^{\mathcal{G}}$) for the merging-paths task (left) and the best-of-ten paths task (right).

take place in total. **In the merging paths task (left), two hyper-parameter settings have a clear effect: (i) short visits with of high probability lead to bad performance. As such settings lead to a quick mixing of the population, they lead to convergence to the local optimum (ii) long visits with high probability work well. Due to the high visit probability, this setting effectively leads to topology where exactly one agent is always on a long visit. Thus, it ensures that sub-groups stay isolated for at least 1000 episodes, after which inter sub-group sharing needs to takes place to ensure that the sub-groups can progress quickly. In the best-of-ten paths task (right), this structure has a clear optimal hyper-parameterization: short visits with high probability are preferred, which maximizes the mixing of the group and makes early exploration more effective.**

In Figure 9, we observe the % of group success ($S^{\mathcal{G}}$) of the dynamic-periodic topology for various values of T_h and T_l. **In the merging paths task (left of Figure 9) medium values for the period of both phases works best, while there is some success when the low connectivity phase lasts long ($T_l = 1000$). In the best-of-ten paths task (rightof Figure 9), we observe the same medium values for the period of both phases work best: thus both the absolute value and their ratio is important to ensure that exploration is efficient. The optimal configuration is the same between the two tasks ($T_l = 100, T_h = 10$), which is a good indication of the robustness of this structure.**

E EMPIRICAL RESULTS

To ensure that all methods have the same number of samples, we assume that, for trials where a method did not find the optimal solution, and, hence, T^+ is undefined, T^+ is equal to the total number of timesteps the method was trained for, T_{train}. For each task, all methods have been trained for an equal duration of time: $T_{\text{train}} = 1e^6$ for the single path , $T_{\text{train}} = 7e^6$ for the merging paths task and $T_{\text{train}} = 2e^7$ for the best-of-ten paths task.

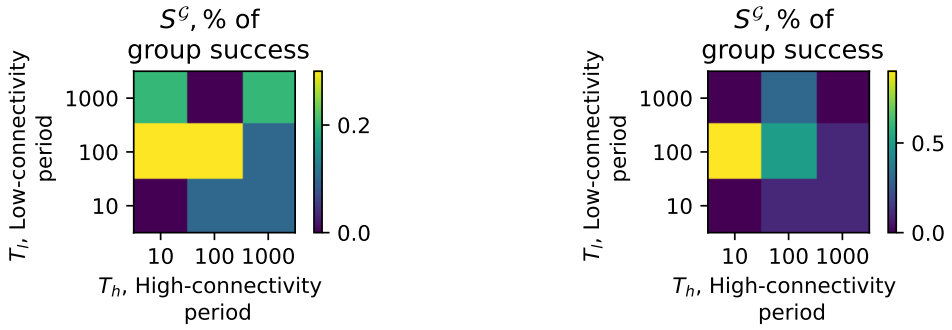

Figure 9: Examining the sensitivity of the dynamic-periodic topology to its hyper-parameters: % of group success ($S^{\mathcal{G}}$) for the merging-paths task (left) and the best-of-ten paths task (right).

We perform 20 independent trials for each task and method and visualize our proposed metrics with barplots and line plots of averages across trials with error bars indicating 95% confidence intervals. We test for statistical significance of our evaluation metrics separately for each task by applying ANOVA tests [8] to detect whether at least one method differs from the rest and, subsequently, employing the Tukey's range test [9] to detect which pairs of methods that differ significantly. We report the exact p values of theses tests in the text and, when applicable, illustrate them in figures using a set of asterisks whose number indicates the significance level ($p <= 0.05$: *, $p <= 0.01$: **, $p <= 0.001$: ***, $p <= 0.0001$: ****) [10].

We presented the major results of our evaluation of SAPIENS in Section 3. We now present additional information regarding the implementation of the different components (Appendix E.1), the values of all performance metrics and additional plots for experiments discussed in 3 (Appendix E.2), results on intra-group and inter-group alignment (AppendixE.3), results for groups of varying sizes (Appendix E.5) and results on various dynamic topologies (Appendix D)

E.1 IMPLEMENTATION DETAILS

Implementation of DQN We employ the same hyper-parameter for each DQN across all studied tasks and topologies: discount factor $\gamma = 0.9$, the Adam optimizer with learning rate $\alpha = 0.001$ (Kingma & Ba, 2014; Dunbar, 2014), ϵ-greedy exploration with $\epsilon = 0.01$. We employ a feedforward network with two layers with 64 neurons each. We implemented SAPIENS by extending the DQN implementation in the stable-baselines3 framework.

Implementation of A2C We used the stable-baselines3 implementation of A2C [11] and tuned the hyper-parameters: learning rate, number of steps, discount factor, the entropy coefficient and the value function coefficient. This gave us the best-performing values 0.001, 5, 0.99, 0.1 and 0.25, respectively, that we also employed in the other tasks.

Implementation of Ape-X We used the ray implementation of Ape-X DQN [12] and tuned the hyper-parameters: learning rate, discount factor, replay buffer capacity and ϵ-greedy exploration. This gave us the best-performing values in the single path task 0.001, 0.9, 5000 and 0.02, respectively.

[8] https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.f_oneway.html

[9] https://pypi.org/project/bioinfokit/0.3/

[10] https://www.graphpad.com/support/faq/what-is-the-meaning-of–or–or–in-reports-of-statistical-significance-from-prism-or-instat/

[11] `https://stable-baselines3.readthedocs.io/en/master/modules/a2c.html`

[12] `https://docs.ray.io/en/latest/rllib/rllib-algorithms.html`

Topology	R^+_∞	R^*_∞	T^+	T^*	$T^>$	S	V_{avg}	C_{avg}
no-sharing	(0.92, 0.0.036)	(1,0)	(236250, 33441)	(830000,0)	(600000, 0)	(1,0)	(0.038,0.002)	(0.697, 0.0354)
dynamic	(1,0)	(1,0)	(237222, 53885)	(346666,122041)	(109444, 98067)	(1,0)	(0.027,0.01)	(0.885, 0.026)
fully-connected	(1,0)	(1,0)	(310666, 89240)	(362000, 98503)	(51333, 20655)	(1,0)	(0.052, 0.027)	(0.891,0.034)
ring	(1,0)	(1,0)	(235333, 70190)	(305333, 78818)	(70000, 22038)	(1,0)	(0.038,0.0026	(0.697, 0.0354)
small-world	(1,0)	(1,0)	(253333, 63320)	(302666, 74110)	(49333, 31274)	(1,0)	(0.029, 0.013)	(0.912, 0.0267)
single	(0.92, 0.163)	(0.927, 0.163)	(64750, 266145)	(64750, 266145)	(0,0)	(0.2, 0.41)	(0.015, 0.013) a non-co	(1,0)
A2C	(1,0)	(1,0)	(36200, 16450)	(36200, 16450)	(0,0)	(1,0)	(0,0)	(1,0)
Ape-X	(0.93, 0.18)	(0.93, 0.18)	(270941, 102445)	(270941, 102445)	(0,0)	(0.15, 0.366)	(0.015, 0.022)	(1,0)

Table 2: Evaluation metrics for the single-path task in the form (mean of metrics, standard deviation of metric)

Topology	R^+_∞	R^*_∞	T^+	T^*	$T^>$	S	C_{avg}	V_{avg}
no-sharing	(0.657, 0.037)	(0.838,0).14	(5334000, 2311945)	(7000000,2311945	(7000000, 0)	(0.4,0.51)	(0.597, 0.06)	(0.0089,0.0021)
dynamic	(0.7,0.04)	(0.9,0.13)	(4716500,222965)	(7000000, 0)	(7000000, 0)	(0.75,0.48)	(0.597, 0.0059)	(0.005, 0.0016)
fully-connected	(0.5349,0.085)	(0.58, 0.04)	(7000000,0)	(7000000, 0)	(7000000, 0)	(0,0)	(0.597,0.0051)	(0.0764, 0.0044)
ring	(0.661,0.135)	(0.72, 0.15)	(5892000, 2288393)	(7000000, 0)	(7000000, 0)	(0.2,0.41)	(0.595, 0.0051)	(0.0149,0.021)
small-world	(0.639,0.091)	(0.774, 0.173)	(5998000, 1699076)	(7000000, 0)	(7000000,0)	(0.3, 0.483)	(0.596,0.0065)	(0.06775,0.0328)
single	(0.758, 0.187)	(0.758, 0.187)	(5235000, 2385948)	(5235000, 2385948)	(0,0)	(0.3, 0.47)	(1,0)	(0.0063, 0.0063)
A2C	(0.269,0).2	(0.269, 0.2)	(7000000, 0)	(7000000, 0)	(0,0)	(0,0)	(1,0)	(0.013, 0.038)
Ape-X	(0.573, 0.31)	(0.573, 0.31)	(6656900, 1534389)	(26656900, 1534389)	(0,0)	(0.05, 0.223)	(1,0)	(0.054,0.157)

Table 3: Evaluation metrics for the merging-paths task in the form (mean of metrics, standard deviation of metric)

Implementation of graphs used as social network structures We construct small-worlds using the Watts–Strogatz model (`watts_strogatz_graph` method of the networkx package [13]). This model first builds a ring lattice where each node has n neighbors and then rewires an edge with probability β. Compared to other techniques used in previous works studying the effect of topology Mason et al. (2008), this way of constructing small-worlds ensures that the average path lengths is short and clustering is high. These two properties are what differentiates small-worlds from random (short average path length and small clustering) and regular (long average path length and high clustering) graphs. We employ $n = 4$ and $\beta = 0.2$ in our experiments, which we empirically found to lead to good values of average path length and clustering.

We have described the generation process of dynamic topologies in Appendix D. In the main paper we employ the dynamic-Boyd topology with $T_v = 10$ and $p_v = 0.001$ across tasks. These parameters have been tuned for the merging-paths task.

E.2 Overall comparison

Tables 2, 3 and 4 contain the values of all metrics discussed in Section 2.4 for the single path, merging paths and best-of-ten paths, respectively. We denote values computed after convergence of the group with underscore ∞ and values averaged over all training steps with underscore avg (note that we use $\bar{}$ over variables to denote averaging over agents in a single training step). Cells with a dash (-) indicate that we could not compute the corresponding metrics because a group failed to find a solution in all trials. We also provide the plots of volatility and average diversity for the merging paths and best-of-10 paths task (that were not included in Figure 5) due to page limit constraints).

Figure 10 presents the reward curves for all methods in the single path, merging paths and best-of-ten paths tasks respectively. Specifically, we plot the maximum reward of the group at training step t (R^+_t).

[13] `https://networkx.org/`

Topology	R^+_∞	R^*_∞	T^+	T^*	$T^>$	S	C_{avg}	V_{avg}
no-sharing	(0.2124,0.036)	(0.446, 0.131)	(20000000, 0)	(20000000,0)	(20000000, 0)	(0,0)	(0.239, 0.005)	(0.007, 0.0021)
dynamic	(0.5141,0.323)	(0775, 0.32)	(13616000, 5441395)	(20000000,0)	(20000000, 0)	(0.6,0.51)	(0.242,0.0078)	(0.04,0.0223)
fully-connected	(0.1615,0.09)	(0.1819, 0.1013)	(20000000, 0)	(20000000, 0)	(20000000,0)	(0,0)	(0.238,0.0053)	(0.007,0.003)
ring	(0.2319,0.3045)	(0.275,0.332)	(18781000, 3854816)	(18826000, 3712513)	(18045000, 6182252)	(0.1,0.31)	(0.237,0.004)	(0.047,0.019)
small-world	(0.198,0.281)	(0.216,0.275)	(18706000, 4091987)	(18746000, 3965496)	(18040000, 6198064)	(0.1,0.316)	(0.234,0.007)	(0.018,0.0049)
single	(0.178, 0.067)	(0.1785,0.0676)	(20000000, 0)	(20000000,0)	(0,0)	(0, 01)	(1,0)	(0.006,0.0031)
A2C	(0.1285,0.19)	(0.1285,0.19)	(20000000, 0)	(20000000,0)	(0,0)	(1,0)	(1,0)	(0.3244,0.35)
Ape-X	(0.481, 0.213)	(0.482, 0.213)	(20000000, 0)	(20000000,0)	(0,0)	(0.9, 0.316)	(1,0)	(0.018,0.009)

Table 4: Evaluation metrics for the best-of-ten paths task in the form (mean of metrics, standard deviation of metric)

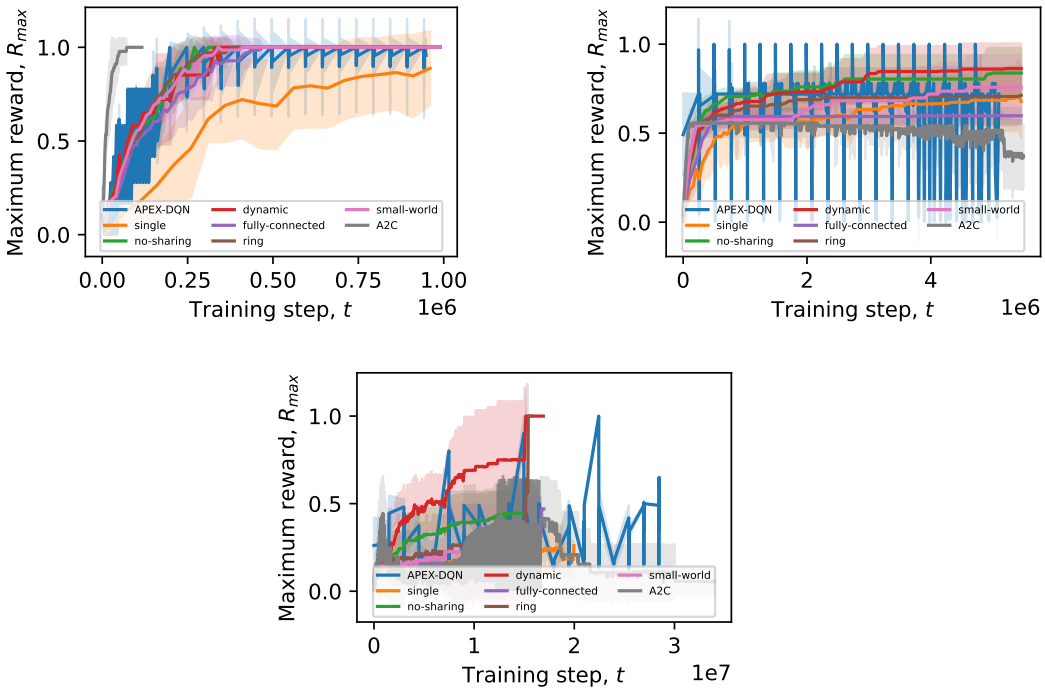

Figure 10: Maximum reward of the group at training step t (R_t^+) in the (left) single path task (middle) merging paths task (right) best-of-ten paths task

E.3 Measuring inter-group and intra-group alignment

We have so far captures the agreement between agents in a group through the behavioral metric of conformity. Here, we present a mnemonic metric for agreement, which we term alignment. Alignment is a complementary metric to the diversity (D_t^k) and group diversity ($D_t^{\mathcal{G}}$) metrics, that aims at capturing the effect of experience sharing on the replay buffers in a group. We propose a definition of alignment within a single group (intra-group alignment$A_t^{\mathcal{G}}$) and a definition of alignment between two different groups ($A_t^{\mathcal{G}_j,\mathcal{G}_j}$). Such metrics of mnemonic convergence have been linked to social network topology (Coman et al., 2016) and, as we show here, they can prove useful in analyzing groups of reinforcement learning agents.

Specifically: (i) $A_t^{\mathcal{G}}$ is the intra-group alignment. This metric captures the similarity in terms of content between the replay buffers of agents belonging to the same group. To compute this we compute the size of the common subset of experiences for each pair of agents and, then, average over all these pairs, normalizing in [0,1]. (ii) inter-group alignment $A_t^{\mathcal{G}_j,\mathcal{G}_j}$ is a similar notion of alignment but employed between different groups (e.g. how different is a group of fully-connected and a dynamic group of agents in terms of the content of their group replay buffers). To compute it we concatenate all replay buffers of a group into a single one and then compute the size of the common subset of the two replay buffers.

Figure 12 presents intra-group alignment in the three tasks. **We observe that, in all tasks, intra-group alignment increases with connectivity and that it reduces when the agents enter the exploitation phase. Thus, intra-group alignment can prove useful in characterizing the exploration behavior of a group.** In Figure 13, we present the inter-group alignment in the single path, merging paths and best-of-ten paths tasks. We observe that the topologies do not differ significantly in the single path task. **In the merging task, we observe that inter-group alignment is lower during the exploration phase, compared to other tasks, and that the small-world is the slowest to align with all other structures. Perhaps this explains why this topology finds the optimal solution with the least probability: by propagating information quickly, the group early on**

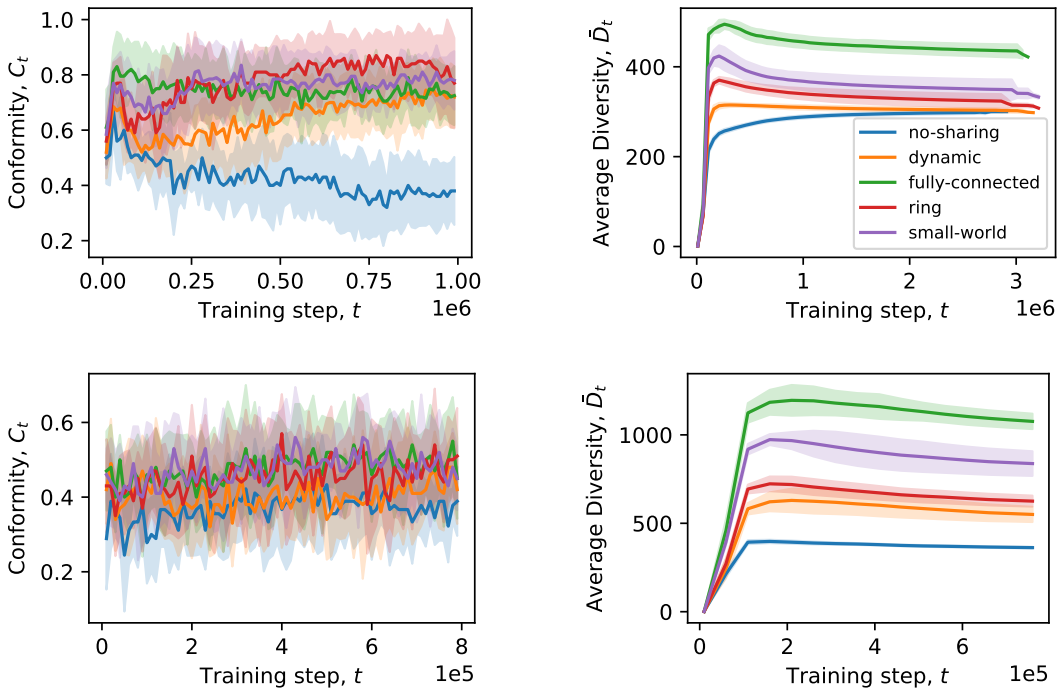

Figure 11: Analyzing group behavior in the merging paths task (top row) and best-of-10 paths task (bottom row). (left) Conformity C_t is a behavioral metric that denotes the percentage of agents in a group that followed the same trajectory in a given evaluation trial (right) Average Diversity $\bar{D}_t$ is a mnemonic metric that denotes the number of unique experiences in the replay buffer of an agent, averaged over all agents.

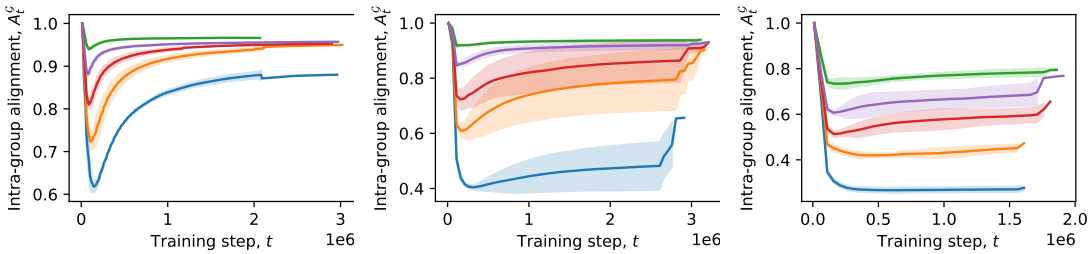

Figure 12: Intra-group alignment $A_t^{\mathcal{G}}$ in the single path task (left), merging paths task (middle) and best-of-ten paths task (right)

converges to the local optimum in this task. In the best-of-ten task, the no-sharing setting has the smallest alignment with all other structures. This reinforces our main conclusion in this work: experience sharing affects individuals and different topologies do so in different ways.

E.4 Robustness to learning hyper-parameters

In Figure 14 we present how the performance of SAPIENS varies for different values of the learning hyperparameters learning rate and disocunt factor in the single path task under a fully-connected and a dynamic topolgoy, as well as the *no-sharing* condition. We observe that, although convergence to the optimal solution is not always achieved, the dynamic topology is at least as effective as the others either in terms of convergence rate and/or final performance in all conditions.

E.5 Effect of group size

We here examine the effect of the group size for all social network structures in the merging-paths and best-of-ten paths task. To visualize the progression of a group on the paths of the different tasks, we focus on specific elements in the tasks: (i) ($[A_8, B_8, C_2]$ in the merging-paths task. The first two correspond to reaching the end of the paths corresponding to the two local optima. To reduce the computational complexity of experiments, we do not study the last element of the optimal path (C_4), but focus on C_2 instead. This is sufficient to detect whether a group has discovered the optimum path. Here, we observe that the fully-connected topology fails to find the optimal path regardless of its size (with a small success probability for $N = 10$). We observe that the ability of the ring , small-world and dynamict topologies to avoid the local optima improves with the group size (ii) $[B_4, A_2, E_2]$ in the best-of-ten tasks. B_4 is the fourth element on the optimal path (again we do not study the last element to reduce complexity). To avoid cluttering the visualization we only present two of the nine sub-optimal paths. In this task, we again observe that the fully-connected network fails to discover the optimal task. Among all structures and group sizes, the large dynamic network performs best, while the performance of ring and small-world is also best for $N = 50$. We observe that small networks sizes ($N = 2, N = 6$) are slower at exploring (we can see that as they rarely find the second element of the sub-optimal paths, which is required to conclude that path B is the optimal choice).

Overall, **this scaling analysis indicates that increasing the group size in a fully-connected topology will not improve performance, while benefits are expected for low-connectivity structures, particularly for the dynamic topology.** We believe that this observation is crucial. In studies of groups of both human and artificial agents, we often encounter the conviction that, larger groups perform better and that size is a more important determinant than connectivity, the latter justifying why connectivity is often ignored Kline & Boyd (2010); Horgan et al. (2018); Mnih et al. (2016); Schmitt et al. (2019); Nair et al. (2015). Our results here point to the contrary.

E.6 Prioritized experience sharing

We now examine how sharing prioritized experiences instead of randomly sampled ones affects the performance of SAPIENS. In Figure 16 we repeat the same experiment with Figure 4, with the difference that all methods compute priorities, which they employ both for implementing a prioritized

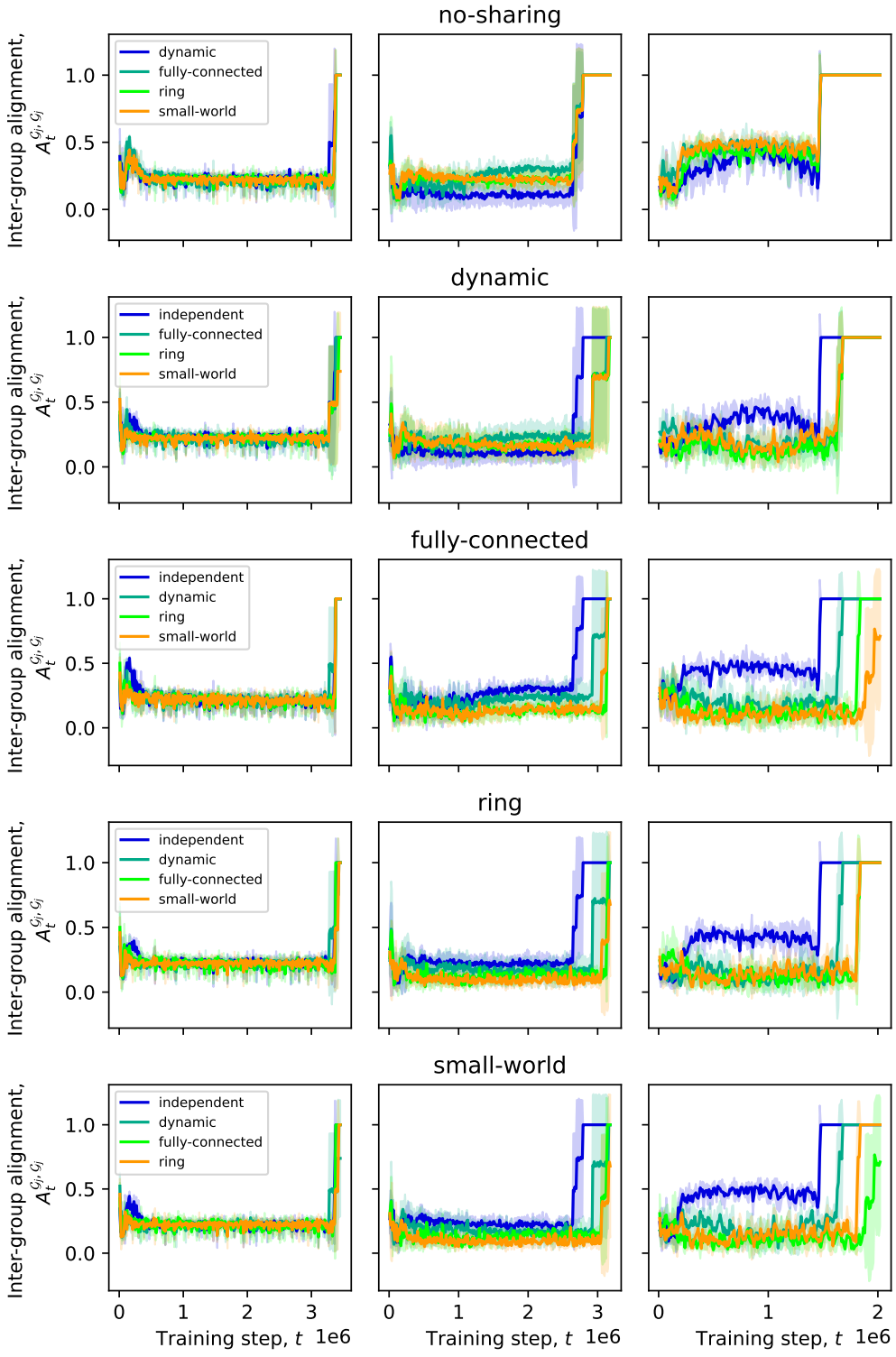

Figure 13: Inter-group alignment $A_t^{\mathcal{G}_j,\mathcal{G}_j}$ in the single path task (left), merging paths task (middle) and best-of-ten paths task (right). In each row we compare one topology with all the rest.

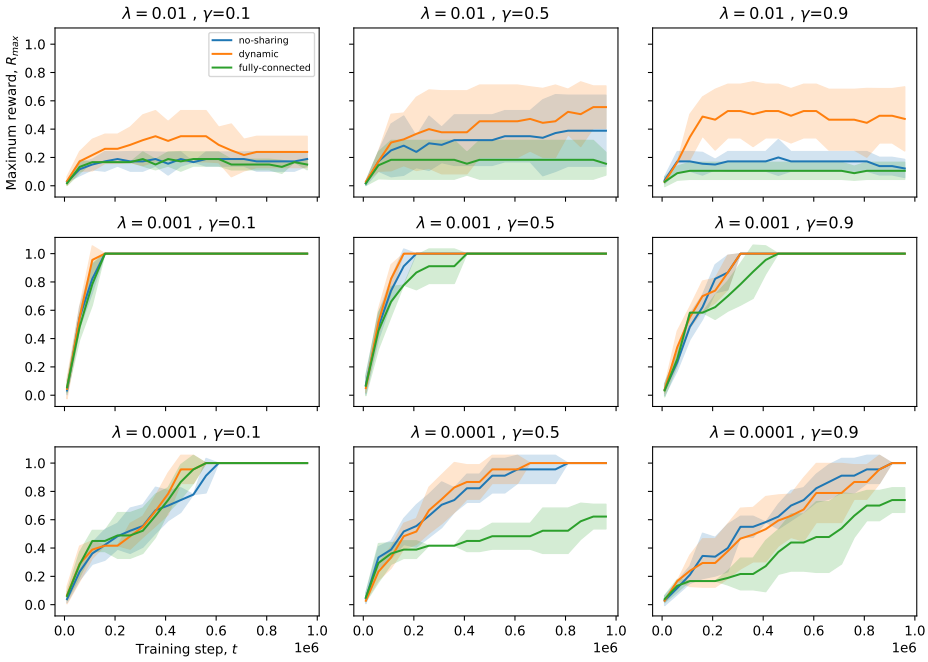

Figure 14: Varying the learning hyper-parameters learning rate (λ) and discount factor (γ) in different social network topologies in the single path task.)

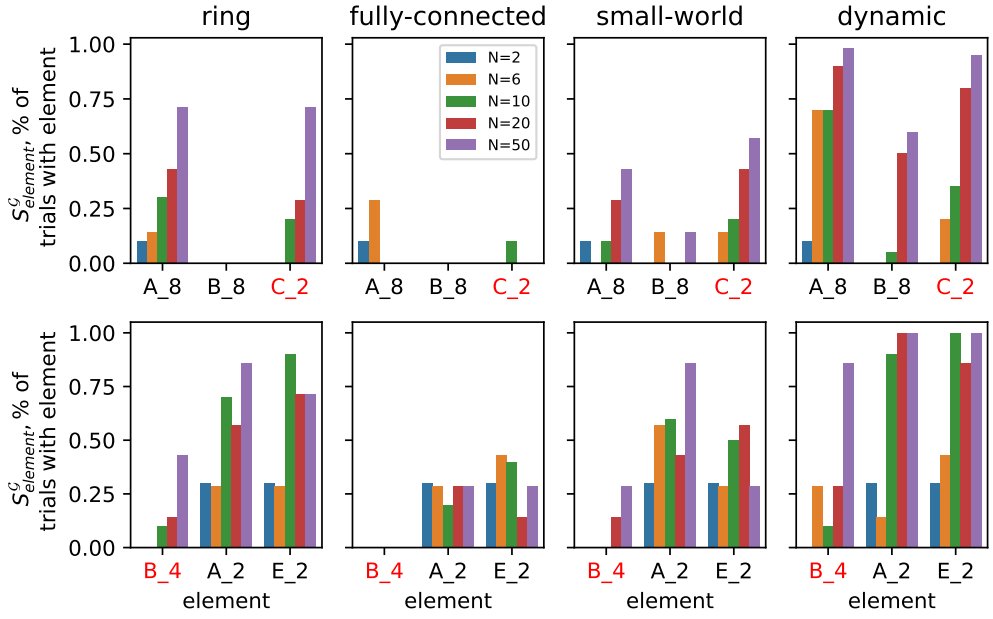

Figure 15: Scaling of different social network structures in the merging paths (top row) and best-often paths tasks (bottom row). We highlight the element belonging to the optimal path in red.

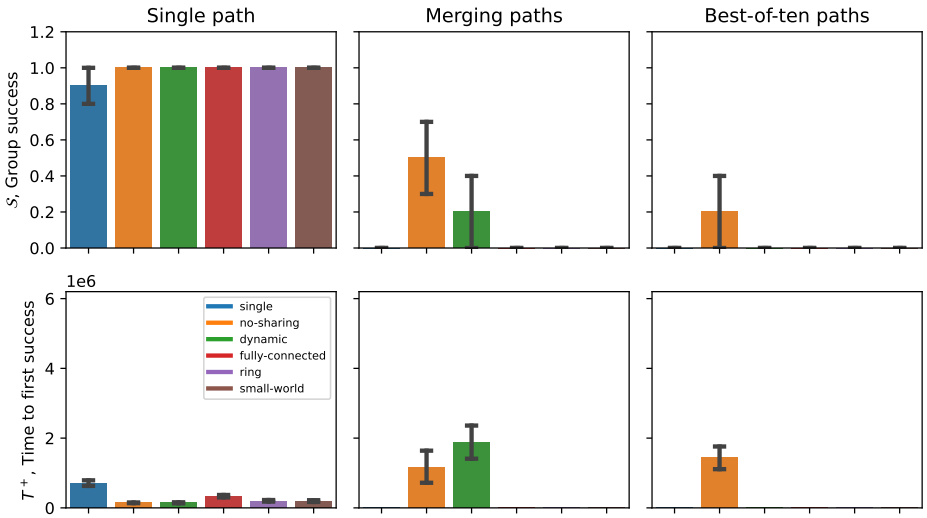

Figure 16: Examining the effect of prioritization in experience sharing. For more details about the setup, we refer the reader to Figure 4

replay buffer and sharing experiences by sampling them in proportion to their priorities. As we see, using priorities negatively impacts experience sharing, while it helps speed up the performance of the single agent in the single path task. This behavior has been observed in previous works Souza et al. (2019) and can be attributed to the fact that the priorities of the sender do not necessarily agree with the priorities of the receiver and, therefore, destabilize learning.

E.7 Additional test-bed: the deceptive coins game

Deceptive games are grid-world tasks introduced to test the ability of deep RL agents to avoid local optima. (Bontrager et al., 2019). Here, we perform preliminary experiments with our own JAX-based implementation of one of the games: the first difficulty level of the deceptive coins game (see Figure 17 for an illustration). Here, the agent can navigate in the grid-world during an episode and collect diamonds, which give a unit of reward. The game finishes once the agent reaches the fire, which offers an additional reward, or when a timeout of 14 time steps is reached. There are two possible paths the agent can follow: moving left and reaching the fire will give a reward of two while moving right and reaching the fire will give a reward of five. The second path is more rewarding but is harder to complete because, once an agent discovers the easier-to-find diamond on the left, it is deceived into following the left path. Once an agent commits on a path (reaches the edge of the grid-world) a barrier is raised so that the agent cannot go back within that episode.

We now examine the performance of SAPIENS under different social network structures (fully-connected, small-world, ring, dynamic), as well as the no-sharing, A2C and Ape-X baselines for three group sizes: 6 , 10 and 20 agents. We present the reward plots for the 3 sizes in Figures 18, 19 and 20, respectively, and present an overall comparison in Figure 21 (equivalent to Figure 4 for the Wordcraft tasks).

We observe that all conditions found either the local or the global optimum and that : a) A2C fails for all network sizes. This behavior has been observed in previous works (Bontrager et al., 2019) and can be attributed to the fact that policy-gradient methods are more susceptible to local minima b) no-sharing gets stuck in the local optimum in half of the trials when the group size is small. Increasing the group size increases the probability that at least one agent in the group will escape the local minima by ϵ-greedy exploration c) partially connected structures find the global minima across network sizes d) fully-connected converges to the local optimum for the large group size, although the global optimum was discovered at the early exploration phase (see Figure 20). Thus, too much experience sharing is harmful e) Ape-X fails with high probability for all network sizes.

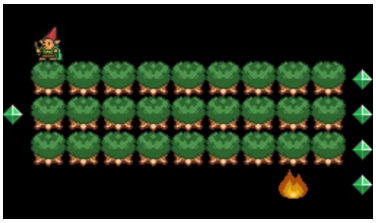

Figure 17: A screenshot of our implementation of the Deceptive Coins task. Collecting diamonds gives a positive reward and touching the fire terminates the game.

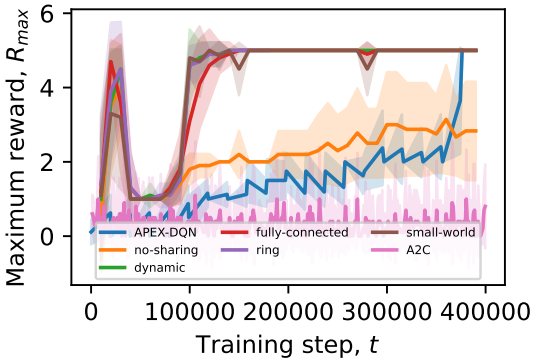

Figure 18: Performance for a group with 6 agents

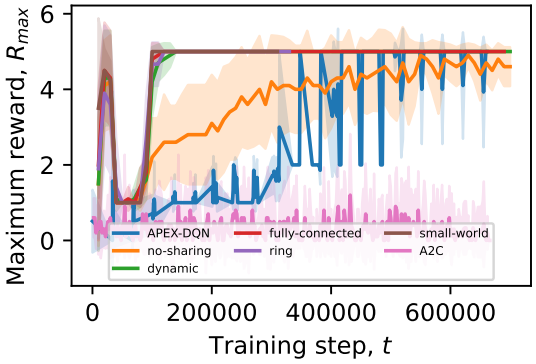

Figure 19: Performance for a group with 10 agents

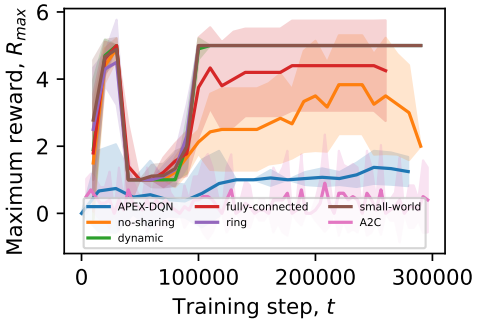

Figure 20: Performance for a group with 20 agents

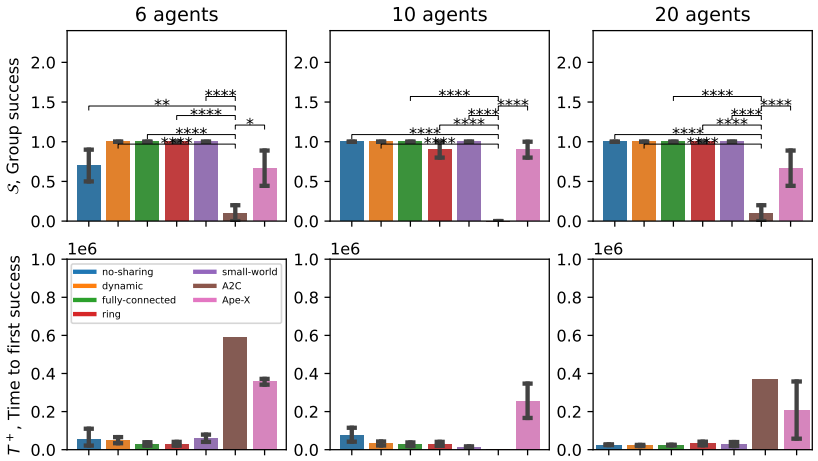

Figure 21: Overall performance comparison for a group with: 6 agents (first column), 10 agents (second column) and 20 agents (third column) task. We present two metrics: group success ($\mathcal{S}$) denotes whether at least one agent in the group found the optimal solution (top row) and T^+, Time to first success, is the number of training time steps required for this event (bottom row). Note that T^+ can be computed only for $\mathcal{S} > 0$ its error bars and significance tests can only be computer for $\mathcal{S} > 1$. We denote statistical significance levels with asterisks.)

In general, our conclusions in this task are consistent with what we observe in Wordcraft, in particular the merging paths task that has a similar deceptive nature.

E.8 Robustness to amount of sharing (p_s and L_s)

In Section 2.3 we formulated SAPIENS and described two hyper-parameters: p_s is the probability of sharing a batch of experience tuples at the end of an episode and L_s is the length of this batch. Here, we test the robustness of SAPIENS to these two hyper-parameters, which both control the amount of shared information and, therefore, interact with hyper-parameters of the DQNs (in particular the learning rate) to control the rate at which information is shared to the rate of individual learning. Specifically, we evaluate the dynamic topology (with the same hyper-parameters employed in the main paper, i.e., visit duration $T_v = 10$ and probability of visit $p_v = 0.05$) and the fully-connected topology in the deceptive coins game (described in Appendix E.7) with 20 DQN agents.

In Figure 22 we present group success (S) averaged across trials for a parametric analysis over $L_s \in (1, 6, 36)$ and $p_s \in (0.35, 0.7, 1)$. We observe that the dynamic topology finds the optimal solution across conditions except for a small probability of failure for $(L_s = 1, p_s = 0.35)$ and $(L_s = 1, p_s = 0.7)$. These values correspond to low amounts of information sharing. In this case, the dynamic structure becomes more similar to a no-sharing structure: the amount of shared information is not enough to help the agents avoid local optima they fall into due to individual exploration. For the fully-connected topology we observe that performance degrades for high amounts of information ($(L = 36, p_s = 0.35)$, $(L = 36, p_s = 0.7)$, $(L = 36, p_s = 1)$). This is in accordance with our expectation that fully-connected topologies lead to convergence to local optima. Interestingly, this structure performs well when $p_s = 1$ and $L_s \leq 6$. Thus, sharing more frequently is better than sharing longer batches: we hypothesize that this is because longer batches have more correlated data, making convergence to local optima more probable.

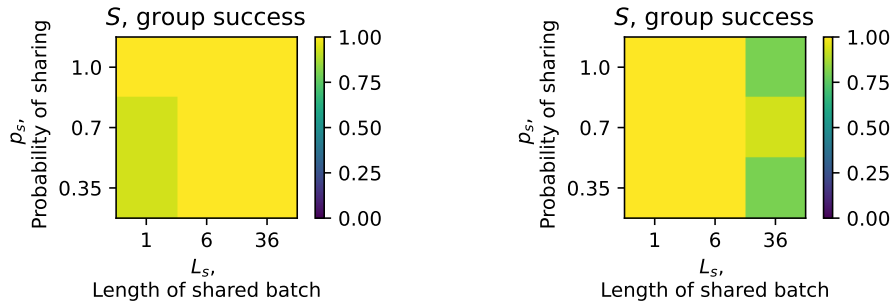

Figure 22: Robustness of group success S to sharing hyper-parameters p_s and L_s for dynamic (left) and fully-connected (right)