United We Stand: Decentralized Multi-Agent Planning With Attrition (Supplementary Material)

Nhat Nguyen[a,*], **Duong Nguyen**[a], **Gianluca Rizzo**[b] **and Hung Nguyen**[a]

[a]The University of Adelaide, Australia
[b]HES SO Valais, Switzerland, and the University of Foggia, Italy.

A Technical Results

A.1 Details of Discounted Upper Confidence Bound on Tree (D-UCT)

Consider an arbitrary node s with a set of child nodes $\mathcal{A}_n(s)$. Whenever s is visited, the child node $s' \in \mathcal{A}_n(s)$ with the largest D-UCB is chosen as

$$a_n^{(t)}(s) = \underset{s' \in \mathcal{A}_n(s)}{\arg\max}\, X_n^{(t)}(s') \ , \tag{1}$$

where $X_n^{(t)}(s')$ is the D-UCB score for node s' at iteration t. $X_n^{(t)}(s')$ is updated using the D-UCB algorithm [2] as follows.

First, let $\gamma \in (1/2, 1)$ be a discounting factor and $C_p > 1\sqrt{8}$ an exploration constant, the upper bound confidence for child node s' is calculated as

$$X_n^{(t)}(s', \gamma) := \bar{F}_n^{(t)}(s', \gamma) + c_n^{(t)}(s', \gamma) \ , \tag{2}$$

where $\bar{F}_n^{(t)}(s', \gamma)$ is the average empirical reward for choosing s', and $c_n^{(t)}(s', \gamma)$ is the exploration bonus.

Denote the discounted number of times the child node s' of the parent node s has been visited as

$$N_n^{(t)}(s', \gamma) := \sum\nolimits_{\tau=1}^{t} \gamma^{t-\tau} \mathbf{1}_{\left\{a_n^{(\tau)}(s)=s'\right\}} \ , \tag{3}$$

where $\mathbf{1}_{\left\{a_n^{(\tau)}(s)=s'\right\}}$ is the indicator function that returns 1 if node s' was selected at round τ and 0 otherwise.

Let $F_n^{(\tau)}$ be the rollout score at iteration $\tau \leq t$ and $N_n^{(t)}(s, \gamma)$ be the discounted number of times the parent node s has been visited. Then at time t, the average reward for node s' is computed as

$$\bar{F}_n^{(t)}(s', \gamma) = \frac{1}{N_n^{(t)}(s', \gamma)} \sum\nolimits_{\tau=1}^{t} \gamma^{t-\tau} F_n^{(\tau)} \mathbf{1}_{\left\{a_n^{(\tau)}(s)=s'\right\}}, \tag{4}$$

and exploration bonus as

$$c_n^{(t)}(s', \gamma) = 2C_p \sqrt{\frac{\log N_n^{(t)}(s, \gamma)}{N_n^{(t)}(s', \gamma)}}. \tag{5}$$

A.2 Analysis of Dec-MCTS Performance in Attrition Settings

As shown in [1], Dec-MCTS has vanishing regret and converges as $t \to \infty$. We prove here the behavior of Dec-MCTS after convergence. Recall that by convergence we mean each agent stays with the same action sequence (i.e., the UCB score for each action in such sequence is the highest at that corresponding decision node).

Assume that Dec-MCTS converges at iteration τ_0 (finite) for all agents. At iteration $t > \tau_0$, let x_n^* denote the converged action sequence of agent n, and x_{-n}^* denotes the converged action sequences of every other agent except agent n. The rollout score received by agent n for executing the action sequence x_n^* given by the *marginal utility* will then be a constant L:

$$F_n^{(t)}(x_n^*) = U_n(x_n^*, x_{-n}^*) = U_g(x_n^*, x_{-n}^*) - U_g(x_{-n}^*) = L \ . \tag{6}$$

Assume that at the next iteration $t+1$, a subset of agents becomes unavailable due to failures. Let x'_{-n} be the combined sequences of actions taken by all agents except agent i and the lost agents. That is

$$x'_{-n} \subseteq x_{-n}^*.$$

and the rollout score agent i receives for the same action sequence now is

$$F_n^{(t+1)}(x_n^*) = U_n(x_n^*, x'_{-n}) = U_g(x_n^*, x'_{-n}) - U_g(x'_{-n}) \ . \tag{7}$$

Lemma 1. *Assume that the Dec-MCTS algorithm has converged on all the agents at time step τ_0 and that the global objective function U_g is submodular. Then*

$$X_n^{(t+1)}(x_n^*, \gamma) \geq X_n^{(t)}(x_n^*, \gamma), \ \forall t \geq \tau_0.$$

Lemma 1 essentially states that once Dec-MCTS converges, the D-UCB score calculated by agent n for its converged action sequence x_n is non-decreasing even if it detects that some of the other agents have failed. Hence, it always picks and updates the same action sequence (i.e., the series of actions that has the highest D-UCB scores at each corresponding decision node).

Let $c, p \in x_n^*$ be two nodes in the converged action sequence of agent i, with c being the child node of p. After the algorithm converges at iteration τ_0, by definition, the nodes c and p are going to be selected repeatedly. Thus, at iteration t, the discounted number of

[*] Corresponding Author. Email: nhatdaoanh.nguyen@adelaide.edu.au

times c is visited can be written as

$$N_n^{(t)}(c,\gamma) = \gamma^{t-\tau_0}\, N_c^{(\tau_0)} + \sum_{\tau=0}^{t-\tau_0} \gamma^{\tau} \\ = \gamma^{t-\tau_0}\, N_c^{(\tau_0)} + \frac{1-\gamma^{t-\tau_0+1}}{1-\gamma}, \tag{8}$$

with the constant $N_c^{(\tau_0)}$ is the discounted number of times node c is chosen at τ_0. In addition, the discounted number of times p is visited at iteration t is

$$N_n^{(t)}(p,\gamma) = \gamma^{t-\tau_0}\, N_p^{(\tau_0)} + \sum_{\tau=0}^{t-\tau_0} \gamma^{\tau} \\ = \gamma^{t-\tau_0}\, N_p^{(\tau_0)} + \frac{1-\gamma^{t-\tau_0+1}}{1-\gamma}, \tag{9}$$

with the constant $N_p^{(\tau_0)}$ is the discounted number of times node p is chosen at τ_0. Finally, the accumulated rollout score for c at iteration t is

$$\sum_{\tau=1}^{t} \gamma^{t-\tau} F_n^{(\tau)} \mathbf{1}_{\left\{a_n^{(\tau)}(p)=c\right\}} = \gamma^{t-\tau_0} F^{(\tau_0)} + L \sum_{\tau=0}^{t-\tau_0} \gamma^{\tau} \\ = \gamma^{t-\tau_0} F^{(\tau_0)} + L\, \frac{1-\gamma^{t-\tau_0+1}}{1-\gamma}\;, \tag{10}$$

with the constant $F^{(\tau_0)}$ is the accumulated rollout score for c at τ_0, and L is the rollout score for c at every iterations up to τ_0 as given in (6). For brevity of notations, we denote the following values

$$\begin{aligned} N_c &= \gamma^{t-\tau_0}\, N_c^{(\tau_0)} + \frac{1-\gamma^{t-\tau_0+1}}{1-\gamma}, \\ N_p &= \gamma^{t-\tau_0}\, N_p^{(\tau_0)} + \frac{1-\gamma^{t-\tau_0+1}}{1-\gamma}, \\ F_c &= \gamma^{t-\tau_0}\, F^{(\tau_0)} + L\, \frac{1-\gamma^{t-\tau_0+1}}{1-\gamma}, \\ A &= F_n^{(t+1)}(x_n^*). \end{aligned} \tag{11}$$

At iteration $t+1$ when failures occur, the values for the discounted numbers of times node c and p are chosen, and the accumulated rollout score for c can be updated as

$$\begin{aligned} N_n^{(t+1)}(c,\gamma) &= N_c + \gamma^t, \\ N_n^{(t+1)}(p,\gamma) &= N_p + \gamma^t, \\ \sum_{\tau=1}^{t+1} \gamma^{t+1-\tau} F_n^{(\tau)} \mathbf{1}_{\left\{a_n^{(\tau)}(p)=c\right\}} &= F_c\,\gamma + A\,, \end{aligned} \tag{12}$$

with A is the rollout score for c at iteration $t+1$ as given above. The inequality of Lemma 1 now can be written as

$$\begin{aligned} &\frac{F_c\gamma + A}{N_c+\gamma^t} + c_p\sqrt{\frac{2\log(N_p+\gamma^t)}{N_c+\gamma^t}} \geq \frac{F_c}{N_c} + c_p\sqrt{\frac{2\log(N_p)}{N_c}} \\ \Leftrightarrow\ &\frac{F_c\gamma + A}{N_c+\gamma^t} - \frac{F_c}{N_c} \\ &+ c_p\left(\sqrt{\frac{2\log(N_p+\gamma^t)}{N_c+\gamma^t}} - \sqrt{\frac{2\log(N_p)}{N_c}}\right) \geq 0. \end{aligned} \tag{13}$$

Let

$$f(t) = \frac{F_c\gamma + A}{N_c+\gamma^t} - \frac{F_c}{N_c} + c_p\left(\sqrt{\frac{2\log(N_p+\gamma^t)}{N_c+\gamma^t}} - \sqrt{\frac{2\log(N_p)}{N_c}}\right) \tag{14}$$

be a funtion of time t over the set of fixed paramters $\{\gamma, c_p, F_c, N_c, N_p\}$.

It can be verified that $f(t)$ is an increasing function as the derivative of $f(t)$ is positive for $t \gg \tau_0$. In addition, as $t \gg \tau_0$, the inequality of (13) becomes:

$$\begin{aligned} &\frac{F_c\gamma + A}{N_c} \geq \frac{F_c}{N_c} \\ \Leftrightarrow\qquad &A \geq F_c(1-\gamma) = \left(\gamma^{t-\tau_0} F^{(\tau_0)} + L\frac{1-\gamma^{t-\tau_0+1}}{1-\gamma}\right)(1-\gamma) \\ \Leftrightarrow\qquad &A \geq L. \end{aligned}$$

The last inequality follows from the assumption that the global utility function U_g is submodular. That is, having failures as time $t+1$ implies there are fewer agents collecting rewards, hence the local utility for agent i increases (or remains the same). Thus $A \geq L$.

Since Proposition 1 gives that $F_n^{\tau+1}(x_n^*) \geq F_n^{\tau}(x_n^*)$, there exists a τ_0 for which $f(t)$ is non-negative for some $t \gg \tau_0$. This implies the UCB scores for each node in the actions sequence x_n^* will remain the highest. Thus, agent i will continue to select x_n^* after failures. This concludes the proof.

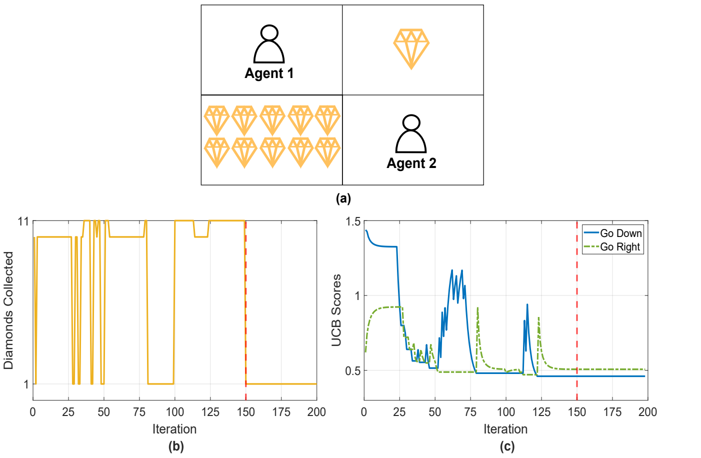

Figure 1. Diamonds collection game (a), number of diamonds collected (b), and D-UCB score for each action of Agent 1 (c).

Let's illustrate the significant implication of the lemma through an example. Consider a grid-world diamonds collection game [5], where two agents play in a team using Dec-MCTS. An exploration factor C_p and a discounting factor γ for Dec-MCTS are chosen as 0.5 and 0.75 respectively. As shown in Figure 1, when simulating the game we see that the D-UCB scores for Agent 1 fluctuate until it converges to $\{0.4607, 0.5072\}$ (after 135 iterations in our example). The empirical average reward of *Go Right* is estimated by Agent 1 by dividing its contribution (one diamond) to the global utility (11 diamonds), while the discounted number of visits is approximately 4, yielding the asymptotic score of 0.5072. The D-UCB score for *Go Down* (the sub-optimal action) is non-deterministic depending on the random choices made by the two agents during the initial transient. This value is not updated in convergence as that branch of the search tree is not sampled, due to the MCTS selection policy.

Using the marginal contribution as the utility improves stability and convergence speed, but it causes issues when agents fail, as shown. At iteration 150, Agent 2 fails (or leaves the game). In this case, even if the optimal choice for Agent 1 would be *Go Down* (due to the higher amount of diamonds), it sticks with *Go Right*. This happens because both the exploration bonus and the local contribution to the overall *hypothetical global reward* remain the same, despite the real global reward has been reduced.

B Proof of Theorem 1

Before going through the proof of Theorem 1, we first prove the existence of at least one PSNE in the formulated game.

Lemma 2. *A finite coordination game will always have at least one PSNE, if maximizing players' local utilities corresponds to maximizing the global objective, i.e., the players' local utility functions satisfy,* $\forall x_n, x'_n \in \hat{\mathcal{X}}_n,\ \forall x_{-n} \in \hat{\mathcal{X}}_{-n},\ \forall n \in \mathcal{N}$,

$$U_n(x_n, x_{-n}) - U_n(x'_n, x_{-n}) > 0 \ \Rightarrow\ \Phi(x_n, x_{-n}) - \Phi(x'_n, x_{-n}) > 0\ , \tag{15}$$

where $\Phi(\cdot)$ *is a function that represents the global objective.*

Proof. Every finite coordination game in which the global objective function is aligned with the local utility functions of the players, that is, satisfies the property as in (15), is a generalized ordinal potential game [4]. Let ϕ be a potential function of a coordination game $\mathcal{G}$. Then the equilibrium set of $\mathcal{G}$ corresponds to the set of local maxima of ϕ. That is, an action profile $x = (x_n, x_{-n})$ is a NE point for $\mathcal{G}$ if an only if for every $n \in \mathcal{N}$,

$$\phi(x) \geq \phi(x'_n, x_{-n}),\ \forall x'_n \in \hat{\mathcal{X}}_n\ .$$

Consider $x^* = (x^*_n, x^*_{-n}) \in \hat{\mathcal{X}}$ for which $\phi(x^*)$ is maximal (which is true by definition for a finite set $\hat{\mathcal{X}}$), then for any $x' = (x'_n, x_{-n})$:

$$\phi(x^*_n, x^*_{-n}) > \phi(x'_n, x_{-n}) \Leftrightarrow U_n(x^*_n, x^*_{-n}) > U(x'_n, x_{-n})\ .$$

Hence, the game possesses a pure strategy NE. □

We now proceed with the main proof. It is well known that, for any finite matrix game, if all players apply the same Regret Matching policy the empirical distribution of all players' joint action converges to the set of *Coarse Correlated Equilibria (CCE)* [3]. We prove a stronger result of convergence to a PSNE under the assumption of submodular utility functions.

As we formulate the problem of multi-agent information gathering as maximization of a submodular function, the considered matrix game generated by the active agents and their corresponding sets of best feasible paths at each decision point satisfies the following two properties:

- *Property 1*: $\sum_{n \in \mathcal{N}} \lambda_n\, U_n(x)$ is concave in x,
- *Property 2*: $U_n(x_n, x_{-n})$ is convex in x_{-n},

where $x := (x_n, x_{-n})$ denotes a pure joint action in which agent n chooses path x_n and the other agents select x_{-n}. The combination of the two properties implies that player n's local utility function $U_n(\cdot)$ is concave in x_n given x_{-n} is fixed.

Let x be a CCE of the considered game, and let $\bar{x} = \mathbf{E}_\pi[x]$, we then prove that $\bar{x}$ is a pure strategy NE of the game. Without loss of generality, assume that $\lambda_n = 1,\ \forall n \in N$. As x is a CCE point, it satisfies

$$\mathbf{E}\left[U_n(x)\right] \geq \mathbf{E}\left[U_n(x'_n, x_{-n})\right]\ , \tag{16}$$

for every $n \in N$ and every action $x'_n \in \hat{\mathcal{X}}_n$. Also, since $\bar{x} \in \hat{\mathcal{X}}$, using Property 2 we have

$$\mathbf{E}\left[U_n(x'_n, x_{-n}\right] \geq U_n\big(x'_n, \mathbf{E}\left[x_{-n}\right]\big) = U_n(x'_n, \bar{x}_{-n})\ . \tag{17}$$

Combining (16) and (17) yields

$$\mathbf{E}\left[U_n(x)\right] \geq U_n(x'_n, \bar{x}_{-n})\ . \tag{18}$$

Replacing $x'_n = \bar{x}_n$ and then summing over all $n \in N$

$$\sum_{n \in N} \mathbf{E}\left[U_n(x)\right] \geq \sum_{n \in N} U_n(\bar{x}_n, \bar{x}_{-n}) = \sum_{n \in N} U_n(\bar{x}).$$

Using Property 1 implies

$$\sum_{n \in N} \mathbf{E}\Big[U_n(x)\Big] = \mathbf{E}\left[\sum_{n \in N} U_n(x)\right] \leq \sum_{n \in N} U_n\Big(\mathbf{E}\left[x\right]\Big)\ .$$

Therefore

$$\sum_{n \in N} \mathbf{E}\Big[U_n(x)\Big] = \sum_{n \in N} U_n(\bar{x})\ .$$

Thus, $U_n(\bar{x}) = \mathbf{E}\left[U_n(x)\right]$ for every n, and (18) becomes

$$U_n(\bar{x}) \geq U_n(x'_n, \bar{x}_{-n})\ .$$

for every $x'_n \in \hat{\mathcal{X}}_n$. Therefore, $\bar{x}$ is a pure Nash equilibrium. This implies that the time average of the joint action of all players converges to a PSNE solution.

C Further Analysis of Regret Matching Coordination Algorithm

C.1 Rationale behind Nash equilibrium solution

Nash equilibrium is particularly useful in situations where agents have incomplete information about the strategies of other agents, i.e., coordination when agents can not maintain perfect communication. In such cooperative situations with limited communication, NE provides an effective way to find a set of strategies for each agent that is robust to uncertainty and incomplete information. In a Nash solution, no agent can improve its outcome by unilaterally deviating from the NE strategy. Thus, NE is a stable outcome for the agents involved in the planning process where all the agents share the same common objective. Yet, a remaining challenge is that there often exists multiple Nash equilibria and thus how to make sure the combination of these individual Nash-based strategies, which each agent independently computes, defines an optimal equilibrium. The selection of a good Nash equilibrium among the many options, known as an equilibrium selection problem, remains an open question for further investigation. Within the scope of this work, to address this dilemma, we propose that the agents synchronize their reached Nash points to identify the most payoff-dominant Nash solution. The agents then simply follow the best computed Nash equilibrium to select their decisions. In Appendix D, we demonstrate via extensive simulation results that our approximate Nash-based approach achieves an overall good efficiency compared to the optimal solution and substantially improves over the main competing approaches, both in terms of convergence speed and global utility achieved.

C.2 The distributed and parallel execution of Regret Matching

In multi-agent systems where agent loss and unreliable communication are expected to occur, a single point of failure is often unacceptable. Moreover, a centralized approach that requires a global view of the game is often intractable due to the exponential growth of the game's size and complexity. Therefore, we devise a distributed

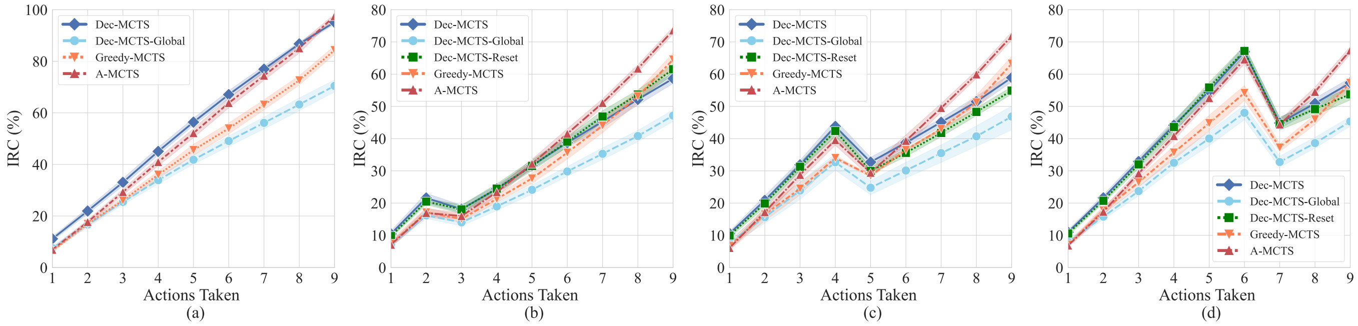

Figure 2. Evolution over the mission of the Instantaneous Reward Coverage (IRC) in the *Forced Failure* setting for different times of attrition: no attrition (a), attrition after 2 actions (b), attrition after 4 actions (c), and attrition after 6 actions (d). Results are with 95% confidence.

approach for executing the coordination algorithm. In our approach, when an agent experiences a loss of communication with other teammates, the agent can predict the other agent's behaviors by simulating their decision choices according to the information received previously. However, when communication loss occurs repeatedly over a certain number of times, it is treated as an agent attrition situation and the remaining agents simply form a new game in which the set of players only contains the active agents. On the one hand, the distributed execution of RM allows the players to independently learn and adapt their strategies based on local information, without the need for a central authority or synchronous communication. On the other hand, the parallel execution of RM (the agents execute the same algorithms in parallel) helps by exploring different NE possibilities to avoid local optima and identify the most payoff-dominant one.

C.3 Computational complexity of Algorithm 2

Regret Matching was proven to guarantee a convergence rate of $\mathcal{O}(1/\sqrt{(T})$ after T iterations [3]. We discuss here how Algorithm 2 scales concerning the size of the problem and the number of available actions for each agent to choose from. For matrix games, where each player has a finite set of actions and the payoffs are given by a matrix, the RM algorithm can be implemented in polynomial time. Specifically, a matrix game with N players and at most M actions per player has M^N action combination in total. Each player has one local utility (or payoff) for each action combination and thus it requires $N \times M^N$ integer numbers to represent all possible players' utilities. Therefore, as the number of players and the number of actions per player increase, the size of the game tree grows exponentially, making it intractable to compute the entire tree in memory or time using a centralized approach.

In contrast, using our proposed distributed approach as presented in Algorithm 2, the computational complexity required for the computation of an NE solution is reduced significantly. In particular, at each learning time step, each RM player learns only its utility vector (of size at most equal to M) for updating its action decision policy in the next time step. As a result, the total amount of queries overtime required by each player to run the algorithm will scale according to $\mathcal{O}(M \times T)$, where T is the number of iterations until convergence. Note that in the implementation of our proposed approach, each agent has to do the same calculations for other simulated players. Thus, the total computational complexity of our solution would scale as $\sim \mathcal{O}(N \times M \times T)$, which is linearly proportional to the number of agents, the number of agent's actions, and the number of iterations until convergence. Consequently, our proposed approach can converge to a NE solution in a distributed and scalable way, making it more suitable, effective, and practical in real-world scenarios, where the players may have access to different and asynchronous information.

D Additional Experimental Resuls

D.1 Time of Attrition Analysis

To perform a baseline evaluation of our algorithm, we consider a setting with no attrition and measure the task performance in terms of instantaneous reward coverage throughout the mission. As shown in Fig. 2a, under this static environment, although the IRC of A-MCTS appeared to be less than Dec-MCTS initially, it ended up comparable and even slightly outperformed the state-of-the-art at the end of the mission. This shows that our proposed algorithm can discover paths that guarantee more long-term rewards and thus is also a good fit for multi-agent coordinated information gathering in general settings.

To evaluate our algorithm's adaptability to failures, we considered the *Forced Failure* setting, in which after a specific number of actions have been taken, half of the agents (chosen at random) become unavailable. Specifically, Fig. 2b, c, and d shows the instantaneous reward coverage with attrition at the early stage (e.g., after 2 actions), middle stage (e.g., after 4 actions), and later stage (e.g., after 6 actions) of the mission respectively. As these figures show, resetting the tree for replanning produced no significant benefits compared to those that adopted the same tree. This is because every MCTS process starts with the exploration phase where agents intentionally take random actions to learn the reward distribution. As such, resetting the tree without sufficient planning would cause the produced joint policy from this period to be sub-optimal. In addition, as Dec-MCTS uses the *marginal contribution* as the utility function, it is unable to recognize the reduction of the global reward and hence is unable to adapt to failures efficiently. Indeed, the gap between it and Dec-MCTS-Global is halved compared to the case with no failure. However, using the global utility function alone is not enough, as sampling other agents' action sequences introduces a lot of variance in the estimation of the global utility. By assuming that the policies of other agents are fixed, both A-MCTS and Greedy-MCTS can overcome this instability issue and adapt to agent failures better, with A-MCTS performing the best in all cases as the regret matching method allowing the agents to discover better joint policies and thus provides better guidance for the exploration-exploitation of the search tree. It

Table 1. Optimality analysis of regret matching.

	Number of Agents					**Number of Components**					**Actions per Component**				
	2	**3**	**4**	**5**	**6**	**10**	**11**	**12**	**13**	**14**	**7**	**9**	**11**	**13**	**15**
PFO (%)	90	85	65	45	40	45	40	45	45	40	65	40	40	30	50
RNO	0.97	0.96	0.97	0.97	0.97	0.97	0.97	0.98	0.98	0.97	0.98	0.97	0.98	0.98	0.98

is also interesting to note that the superiority of A-MCTS compared to Dec-MCTS is slightly reduced (from 15% to 10%) as attrition occurs later. This is expected as when some agents failed in the final stage of the mission, the remaining agents would not have enough action budget left to recover the lost rewards.

D.2 A Closer Look At Regret Matching Behavior

In this section, we study the optimality of the Nash equilibrium point computed by the regret matching algorithm in A-MCTS in the context of the multi-agent underwater data collection problem (Problem 2). We use the following two metrics to evaluate the performance of our algorithm:

- *Probability of Finding Optimal policy (PFO)*: Probability that the Nash policy of our regret matching is optimal in a given setting.

$$PFO = \frac{1}{T}\sum_{t=1}^{T}\mathbf{1}_{\{p(t)=p^*(t)\}}$$

- *Ratio between Nash policy and Optimal policy (RNO)*: Ratio between the utility of the Nash policy of our regret matching and the optimal in a given setting.

$$RNO = \frac{U_g(p(t))}{U_g(p^*(t))}$$

The optimal strategy is computed using an exhaustive search algorithm. Table 1 shows the results of this study with a default number of agents of 6, number of components per agent of 10, and number of actions per component of 9. As expected, the probability of finding the optimal strategy decreases significantly when we increase the number of agents. Indeed, with every added agent, the size of the game increases exponentially, thus potentially causing regret matching to get stuck at local-optimal points. The same behavior was not observed when we increased the number of components per agent or the number of actions per component. Regardless of this, in cases where A-MCTS did not find the optimal strategy, it still sustainably achieved a ratio of 98% compared to the optimal.

References

[1] G. Best, O. M. Cliff, T. Patten, R. R. Mettu, and R. Fitch. Dec-mcts: Decentralized planning for multi-robot active perception. *Int. J. Robot. Res.*, 38(2-3):316–337, 2019.

[2] A. Garivier and E. Moulines. On upper-confidence bound policies for switching bandit problems. In *Algorithmic Learning Theory*, pages 174–188. Springer, 2011.

[3] S. Hart and A. Mas-Colell. A simple adaptive procedure leading to correlated equilibrium. *Econometrica*, 68(5):1127–1150, 2000.

[4] J. R. Marden, G. Arslan, and J. S. Shamma. Cooperative control and potential games. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(6):1393–1407, 2009.

[5] M. Sewak. *Deep reinforcement learning*. Springer, 2019.