Learning to Coordinate without Communication under Incomplete Information

Shenghui Chen[1], Shufang Zhu[2], Giuseppe De Giacomo[3], and Ufuk Topcu[1]

[1] University of Texas at Austin, USA
{shenghui.chen, utopcu}@utexas.edu
[2] University of Liverpool, UK
shufang.zhu@liverpool.ac.uk
[3] University of Oxford, UK
giuseppe.degiacomo@cs.ox.ac.uk

Abstract. Achieving seamless coordination in cooperative games is a crucial challenge in artificial intelligence, particularly when players operate under incomplete information. While communication helps, it is not always feasible. In this paper, we explore how effective coordination can be achieved without verbal communication, relying solely on observing each other's actions. Our method enables an agent to develop a strategy by interpreting its partner's action sequences as intent signals, constructing a finite-state transducer built from deterministic finite automata, one for each possible action the agent can take. Experiments show that these strategies significantly outperform uncoordinated ones and closely match the performance of coordinating via direct communication.

1 Introduction

In artificial intelligence, autonomous agents often compete or cooperate, reflecting real-world interactions. Games offer structured settings to study such behaviors. Much of the research has focused on adversarial games, where agents pursue goals despite adversarial environments [5,7]. Conversely, cooperative games [6] require agents to collaborate toward a shared goal. In this paper, we are interested in *shared-control games* [4], a form of cooperative game in which two players, the *seeker* and the *helper*, collectively control a single token to achieve a goal. For instance, in robotic warehouses, a human operator (seeker) navigates to retrieve items while a support robot (helper) clears obstacles, allowing the operator to advance to item locations (token). Helper agents with such assistive abilities have the potential to enhance collaboration with humans in various settings, from virtual games [3,2] to physical applications like assistive wheelchairs [8].

Shared-control games are especially challenging when players have incomplete or differing information. Such asymmetry, from partial observations or limited game understanding, can cause misaligned or suboptimal actions. Direct communication offers a solution by enabling the exchange of relevant information between players. Recent work leverages large language models to express and interpret intentions via natural language, improving coordination in human-AI

teams [9,10,4]. However, direct communication is not always feasible. In such cases, coordination must rely on inferring intent from observed behavior alone.

In this paper, we consider scenarios where direct verbal communication is unavailable. In such settings, the helper must infer when assistance is needed based solely on the seeker's trajectory. Our framework generalizes *shared-control games* [4] by allowing multi-step control for the seeker and introducing a helper strategy that interprets the observed trajectory for effective coordination. To obtain a helper strategy, we represent it as a finite-state transducer composed of several deterministic finite automata (DFAs), each corresponding to a specific helper action. Each DFA is learned using a variant of Angluin's L* algorithm [1]. The learned DFAs are then combined into a finite-state transducer that encodes the helper's overall strategy.

We evaluated our proposed solution in *Gnomes at Night*^{TM}, the same testbed used by [4], comparing the helper's performance in our no-communication coordination approach with two other cases: a worst-case scenario where the helper does not try to coordinate at all, and a best-case scenario where the helper coordinates through direct communication. We measure success rates and the number of steps to complete the game across a given number of trials and different maze configurations (9×9 and larger 12×12). Results show that no-communication coordination with our solution significantly improves success rates over no coordination in both maze sizes and performs comparably to direct communication.

2 Formal Framework

We extend the *shared-control game under incomplete information* [4] to allow the *seeker* to retain control for multiple steps before transferring it to the *helper*. This modification enables intent to be expressed over action sequences rather than isolated moves. A shared-control game with seeker multi-step dynamics is defined as a tuple $\Gamma = (\mathcal{S}, s_{\text{init}}, s_{\text{final}}, \mathcal{A}^{\mathbf{S}}, \mathcal{A}^{\mathbf{H}}, \mathcal{T}^{\mathbf{S}}, \mathcal{T}^{\mathbf{H}})$, where $\mathcal{S}$ is the finite state space; s_{init} and s_{final} are initial and goal states; $\mathcal{A}^i$ and $\mathcal{T}^i$ are the private action sets and deterministic transition functions for each agent $i \in \{\mathbf{S}, \mathbf{H}\}$. We extend the seeker's transition to action sequences via: $\mathcal{T}^{\mathbf{S}}_*(s, [a_1, \ldots, a_n]) = \mathcal{T}^{\mathbf{S}}(\ldots \mathcal{T}^{\mathbf{S}}(\mathcal{T}^{\mathbf{S}}(s, a_1), a_2), \ldots, a_n)$. A common reward function $\mathcal{R} : \mathcal{S} \times (\mathcal{A}^{\mathbf{S}} \cup \mathcal{A}^{\mathbf{H}}) \to \mathbb{R}$ captures the cooperative objective of minimizing steps to the goal.

Problem 1. Given Γ and a reward function $\mathcal{R}$, the seeker follows a policy $\pi^{\mathbf{S}} : \mathcal{S} \times \mathcal{A}^{\mathbf{H}} \to (\mathcal{A}^{\mathbf{S}})^+$ unknown to the helper, but whose resulting actions the helper can observe. There is no communication between the seeker and the helper. The goal is to learn a helper policy $\pi^{\mathbf{H}} : \mathcal{S} \times (\mathcal{A}^{\mathbf{S}})^+ \to \mathcal{A}^{\mathbf{H}}$ that maximizes cumulative reward:

$$\max_{\pi^{\mathbf{H}}} \sum_{t=0}^{T} \mathcal{R}(s_t, a_t) \quad \text{s.t.} \quad \mathbf{a}_0 = [], s_0 = s_{\text{init}}, s_k = s_{\text{final}}, \quad \text{for some} 0 \le k \le T.$$

$$\begin{cases} \mathbf{a}^{\mathbf{S}}_{t+1} = \pi^{\mathbf{S}}(s_t, a^{\mathbf{H}}_t) & \text{on } \mathbf{S}\text{'s turn,} \\ a^{\mathbf{H}}_{t+1} = \pi^{\mathbf{H}}(s_t, \mathbf{a}^{\mathbf{S}}_t) & \text{on } \mathbf{H}\text{'s turn,} \end{cases} \quad s_{t+1} = \begin{cases} \mathcal{T}^{\mathbf{S}}_*(s_t, \mathbf{a}^{\mathbf{S}}_{t+1}) & \text{on } \mathbf{S}\text{'s turn,} \\ \mathcal{T}^{\mathbf{H}}(s_t, a^{\mathbf{H}}_{t+1}) & \text{on } \mathbf{H}\text{'s turn,} \end{cases}$$

where t indexes turns, and T denotes the total number of turns allowed.

3 No Communication Coordination (NCC)

The seeker pre-determines a policy $\pi^{\mathbf{S}}$ to express intent through action sequences. The helper must learn to strategically perform the actions expected by the seeker when the seeker cannot proceed. To develop a corresponding strategy $\pi^{\mathbf{H}}$ for the helper, we introduce an automata-learning-based technique. The key insight is that when the seeker does not need assistance, it will naturally follow the shortest path. In this case, if the action sequence taken deviates from the shortest path, the extra actions taken are interpreted as intent information. We capture such intent information by associating each helper action with a DFA that accepts such indicative sequences, and use Angluin's L* algorithm [1] to learn these *intent-response DFAs*. The helper plays the role of the learner, querying the seeker (as the teacher) through membership and equivalence queries.

Membership Query. The seeker generates an action sequence, knowing which action it expects the helper to perform. The helper extracts intent segments from the observed sequence, infers an expected action, and performs it. If the performed action matches the seeker's intent, the seeker replies "Yes", and all extracted segments are positive examples for the corresponding intent-response DFA $\mathcal{D}^a$ (where a is the helper's action) and negative for all others. A "No" indicates negative membership for $\mathcal{D}^a$. By counterfactual intuition, if no coordination is needed, the seeker would naturally follow the shortest path. Hence, redundancies in the sequence suggest that the seeker's intent is embedded in segments outside this shortest path. To identify these "informative" segments, the helper constructs a subgraph of visited states, computes the shortest path from prior to current location, and removes it from the action sequence. The remaining segments are hence intent segments.

Equivalence query. Since the seeker's strategy $\pi^{\mathbf{S}}$ implicitly encodes the oracle DFAs, direct comparison is infeasible. Instead, we approximate equivalence by issuing a certain number of membership queries. Once this bound is reached, we treat the learned *intent-response DFAs* $\mathbf{D} = \{\mathcal{D}^a\}_{a \in \mathcal{A}^{\mathbf{H}}}$, where $\mathcal{D}^a = \left\{2^{\mathcal{A}^{\mathbf{S}}}, Q^a, q_0^a, \delta^a, F^a\right\}$, as equivalent to the oracle DFAs.

Helper's Strategy Construction. The learned intent-response DFAs $\mathbf{D}$ allow the helper to recognize the seeker's intent solely by analyzing the seeker's action sequences. When the seeker cannot proceed, it becomes the helper's turn to strategically provide assistance. Given a game $\Gamma = (\mathcal{S}, s_{\text{init}}, s_{\text{final}}, \mathcal{A}^{\mathbf{S}}, \mathcal{A}^{\mathbf{H}}, \mathcal{T}^{\mathbf{S}}, \mathcal{T}^{\mathbf{H}})$ and the learned intent-response DFAs $\mathbf{D}$, we define a strategy generator, i.e., finite-state transducer T, from which we can immediately obtain a helper's strategy $\pi^{\mathbf{H}} : \mathcal{S} \times (\mathcal{A}^{\mathbf{S}})^+ \to \mathcal{A}^{\mathbf{H}}$ to solve Problem 1. Despite lacking formal optimality guarantees, we show empirically that the resulting coordination closely matches that achieved with direct communication.

During the helper's turn, the helper uses the current state s and the seeker's previous action sequence $\mathbf{a}^{\mathbf{S}}$ to infer the expected next action. Informative segments are extracted from $\mathbf{a}^{\mathbf{S}}$ and evaluated against each intent-response DFA in $\mathbf{D}$. Accepted actions are filtered by the helper's transition function, and those with highest frequency are returned as intended actions. Formally, the strategy generator $T = (\mathcal{S}, s_{\text{init}}, \mathcal{A}^{\mathbf{S}}, \mathcal{A}^{\mathbf{H}}, \mathcal{T}^{\mathbf{S}}, \mathcal{T}^{\mathbf{H}}, \varrho, \tau)$ is constructed as follows:

- $\mathcal{S}, s_{\text{init}}, \mathcal{A}^{\mathbf{S}}, \mathcal{A}^{\mathbf{H}}, \mathcal{T}^{\mathbf{S}}, \mathcal{T}^{\mathbf{H}}$ are the same as in Γ.
- $\varrho\colon \mathcal{S} \times \mathbf{a}^{\mathbf{S}}_{t+1} \to 2^{\mathcal{S}}$ is the transition function, where $\mathbf{a}^{\mathbf{S}}_{t+1} = [a^{\mathbf{S}}_{t_1}, \ldots, a^{\mathbf{S}}_{t_n}]$ is the observed seeker's action sequence, such that $\varrho(s, \mathbf{a}^{\mathbf{S}}_{t+1}) = \{\mathcal{T}^{\mathbf{H}}(s, a^{\mathbf{H}}) \mid a^{\mathbf{H}} \in \tau(s, \mathbf{a}^{\mathbf{S}}_{t+1})\}$.
- $\tau\colon \mathcal{S} \times \mathbf{a}^{\mathbf{S}}_{t+1} \to 2^{\mathcal{A}^{\mathbf{H}}}$ is the output function such that $\tau(s, \mathbf{a}^{\mathbf{S}}_{t+1}) = \arg\max_{a \in \mathcal{A}^{\mathbf{H}}}(s, \mathbf{a}^{\mathbf{S}}_{t+1}, \mathcal{A}^{\mathbf{H}}, \mathcal{T}^{\mathbf{H}}, \mathbf{D})$.

T generates a strategy by allowing the helper to arbitrarily select an action returned by the output function $\tau(s, \mathbf{a}^{\mathbf{S}})$.

4 Evaluation

We evaluate our approach in the *Gnomes at Night*™ testbed [4], where each configuration consists of a maze layout and a treasure location. To assess generalization, we use 10 unseen layouts for 9×9 and 12×12 mazes, each with 5 distinct treasure positions, 50 configurations per size. Experiments were run on a MacBook Pro (Apple M1, 8GB RAM, Python 3.9+).

There are three coordination types with different levels of information exchange. In **no coordination (NC)**, the seeker plans its path via a modified A* algorithm, while the helper, lacking communication, guesses randomly. In **direct communication coordination (DCC)**, the seeker explicitly communicates its needs, which the helper executes. In our proposed **no communication coordination (NCC)**, the seeker embeds help requests into its trajectory, which the helper interprets to act accordingly with our DFA-based method. Across all settings, the seeker's behavior is fixed, only the helper's strategy changes.

Each coordination type is evaluated with $n = 100$ trials per configuration. A trial is considered successful if either agent reaches the treasure within $m = 300$ steps (for 9×9) or $m = 600$ steps (for 12×12); failure otherwise.

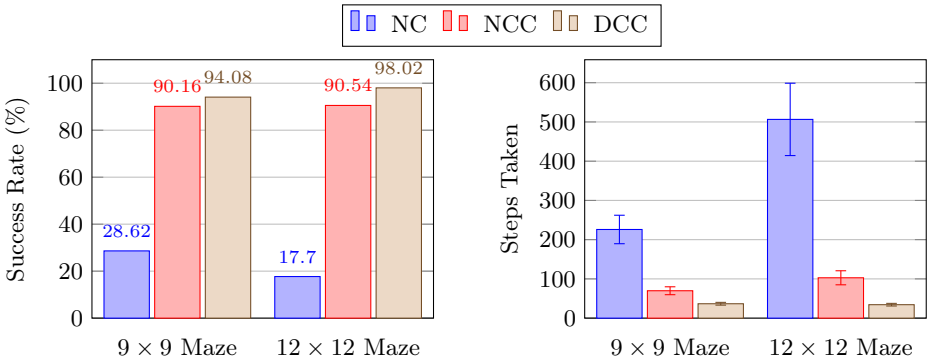

Fig. 1: Success rates (left) and steps taken (right) compared across 50 different configurations for 9×9 and 12×12 mazes.

Success rate is computed as the fraction of successful trials averaged over 50 configurations. The left plot in Figure 1 shows that NCC significantly outperforms NC, improving success rates by 61.54% (9×9) and 72.84% (12×12). NCC

approaches oracle-level performance, with success rates within 4-7% of DCC. A Mann-Whitney U test confirms NCC significantly outperforms NC ($p < 0.001$) in both sizes, while differences between NCC and DCC are not statistically significant ($p > 0.1$). *Steps taken* is reported as the mean and standard deviation of steps to termination across trials and configurations. The right plot in Figure 1 shows that NCC reduces steps compared to NC in both maze sizes ($p < 0.001$), but requires more steps than DCC ($p < 0.001$), as expected as NCC requires more steps to effectively express its intentions through its trajectory.

5 Conclusion

We studied how a helper agent can learn to coordinate with a seeker in cooperative games without communication. Our approach uses automata learning to infer the seeker's intent by constructing a DFA for each helper action. Experiments show that this method approaches the performance of an oracle with direct communication. Future work includes iterative strategy refinement, extension to temporal objectives, and adapting to settings with greater non-determinism, such as human or environmental interactions.

References

1. Angluin, D.: Learning regular sets from queries and counterexamples. Information and Computation **75**(2), 87–106 (1987)
2. Bard, N., Foerster, J.N., Chandar, S., Burch, N., Lanctot, M., Song, H.F., Parisotto, E., Dumoulin, V., Moitra, S., Hughes, E., Dunning, I., Mourad, S., Larochelle, H., Bellemare, M.G., Bowling, M.: The hanabi challenge: A new frontier for ai research. Artificial Intelligence **280**(C) (2020)
3. Carroll, M., Shah, R., Ho, M.K., Griffiths, T., Seshia, S., Abbeel, P., Dragan, A.: On the Utility of Learning about Humans for Human-AI Coordination. In: Advances in Neural Information Processing Systems (2019)
4. Chen, S., Fried, D., Topcu, U.: Human-agent cooperation in games under incomplete information through natural language communication. In: International Joint Conference on Artificial Intelligence, Human-Centred AI track (2024)
5. Cimatti, A., Pistore, M., Roveri, M., Traverso, P.: Weak, strong, and strong cyclic planning via symbolic model checking. Artificial Intelligence **147**(1-2), 35–84 (2003)
6. Dafoe, A., Bachrach, Y., Hadfield, G., Horvitz, E., Larson, K., Graepel, T.: Cooperative ai: Machines must learn to find common ground. Nature pp. 33–36 (2021)
7. Geffner, H., Bonet, B.: A Concise Introduction to Models and Methods for Automated Planning. Synthesis Lectures on Artificial Intelligence and Machine Learning, Springer Cham (2013)
8. Goil, A., Derry, M., Argall, B.D.: Using machine learning to blend human and robot controls for assisted wheelchair navigation. In: IEEE International Conference on Rehabilitation Robotics (2013)
9. Guan, C., Zhang, L., Fan, C., Li, Y., Chen, F., Li, L., Tian, Y., Yuan, L., Yu, Y.: Efficient human-ai coordination via preparatory language-based convention (2023)
10. Liu, J., Yu, C., Gao, J., Xie, Y., Liao, Q., Wu, Y., Wang, Y.: Llm-powered hierarchical language agent for real-time human-ai coordination. In: International Conference on Autonomous Agents and Multiagent Systems (2024)