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*
[2]*University of Liverpool, UK*
[3]*University of Oxford, 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. A full version with appendix is available at https://arxiv.org/abs/2409.12397v3.

Keywords

Games under incomplete information, implicit Communication, shared-control games

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 [1, 2, 3]. Conversely, cooperative games [4] require agents to collaborate toward a shared goal. In this paper, we are interested in *shared-control games* [5], a form of cooperative games 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 progress to its location (token). Helper agents with such assistive abilities have the potential to enhance collaboration with humans in various settings, from virtual games [6, 7] 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. In robotic warehouses, poor inference can reduce efficiency and pose safety risks. 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, 5]. However, direct communication is not always feasible due to constraints like limited bandwidth, latency, noise, or task demands. 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* [5] 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 [11]. The learning process is based on sequences of observed seeker moves, with each DFA accepting those sequences that align with the intention to trigger its associated action and rejecting those that do not. The learned DFAs are then combined into a finite-state transducer that encodes the helper's overall strategy.

We empirically evaluate our proposed solution in *Gnomes at Night*^{TM}, the same testbed used by [5]. We compare 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. We test on 9×9 and larger 12×12 mazes to assess the solution's ability to generalize across maze sizes. 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. It also reduces steps, wall memory, and wall error rate by more than half.

2. Related Works

The problem of achieving coordination in multi-agent systems involves enabling autonomous agents to work together toward shared goals. Prior work spans distributed AI [12], swarm intelligence (stigmergy [13]), and game theory (correlated equilibrium [14]). However, in many settings, all agents are aware of the goal, typically assuming all agents know the goal. In contrast, we study coordination where only one agent knows the goal.

A common approach to address these challenges under incomplete information is through explicit communication, using discrete signals as in Hanabi [7], or natural language in negotiation and co-ordination games like *Deal-or-No-Deal* [15, 16], *Diplomacy* [17], and *MutualFriends* [18]. Recently, *Gnomes at Night*^{TM} was used to highlight the challenges of shared control under incomplete information, when leveraging natural language dialogue for communication [5]. In contrast, our study examines coordination without direct communication, using a mute version of *Gnomes at Night*^{TM}.

Another approach to understanding coordination is through multi-agent reinforcement learning (MARL), where agents learn cooperative strategies via trial and error in complex environments, particularly through self-play and opponent modeling [19]. However, most MARL approaches use neural networks to represent policies, which often obscures the intent inference process within the learning model. The automata-learning-based solution technique proposed in this paper provides a more explicit representation, potentially offering better explainability. *Pedestrian trajectory prediction* similarly involves anticipating future actions from past behavior, environmental conditions, and interactions with others—analogous to the helper inferring the seeker's intent. Approaches include knowledge-based models [20] and supervised deep learning methods [21].

A process mirroring the challenge of the helper attempting to infer the seeker's intended actions is *plan recognition* in planning [22]. Goal recognition involves identifying all potential goals an agent might pursue based on a sequence of observed actions [23, 24, 25, 26, 27, 28]. In this context, the domain is entirely visible, allowing for the calculation of possible goals that can be achieved through an optimal policy aligning with these observations. However, in our setting, the helper lacks information on the seeker's domain. An efficient coordination could help in the mutual understanding of each player's domain. Exploring how to develop such coordination aligns with the focus of this study.

3. Preliminaries

A *deterministic finite automaton (DFA)* is a tuple $\mathcal{D} = (2^{\text{Prop}}, Q, q_0, \delta, F)$, where Prop is the alphabet, Q the finite set of states, $q_0 \in Q$ the initial state, $\delta : Q \times 2^{\text{Prop}} \to Q$ the transition function, and $F \subseteq Q$ the accepting states. The language $\mathcal{L}(\mathcal{D})$ denotes the set of accepted traces.

We use *Angluin's L* algorithm* [11] to learn DFAs via two query types: (1) *Membership queries*, where the learner asks whether a trace σ is accepted; and (2) *Equivalence queries*, where the learner submits a hypothesized DFA and, if incorrect, receives a counterexample to refine it.

4. Formal Framework

We extend the *shared-control game under incomplete information* [5] 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 $\mathcal{T}^i : \mathcal{S} \times \mathcal{A}^i \to \mathcal{S}$ 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. The seeker $\mathbf{S}$ takes the initial turn.

Problem. 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. 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:

$$\max_{\pi^{\mathbf{H}}} \quad \sum_{t=0}^{T} \mathcal{R}(s_t, a_t) \tag{1a}$$

$$\text{s.t.} \quad \mathbf{a}_0 = [], s_0 = s_{\text{init}}, \exists k \in \{0, \ldots, T\} \ s.t. \ s_k = s_{\text{final}}. \tag{1b}$$

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

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

5. Solution Technique

The key challenge for the helper is to infer the seeker's required help by observing its action sequences, as direct communication is disallowed. We propose an automata-learning-based approach in which the helper constructs *intent-response DFAs*—one per helper action—to recognize patterns in the seeker's behavior that imply expected responses. These DFAs, unknown to the seeker, are combined into a finite-state transducer that maps seeker action sequences to helper actions.

5.1. Learning Helper's Intent-Response DFAs

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 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, learning one DFA per action in parallel.

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",

---

Algorithm 1 No-Communication Coordination (NCC)

Input: current state s, seeker action sequence $\mathbf{a}^{\mathbf{S}}$, action space $\mathcal{A}^{\mathbf{H}}$, transition function $\mathcal{T}^{\mathbf{H}}$, intent-response DFAs $\mathbf{D}$

Output: a set of helper actions $A^{\mathbf{H}}$

1: Initialize frequency count $f(a) = 0$ for all $a \in \mathcal{A}^{\mathbf{H}}$
2: $\{\tau_1, \tau_2, \dots\} \leftarrow \mathtt{Capping}(\mathbf{a}^{\mathbf{S}})$
3: **for** each segment τ_i and each action $a \in \mathcal{A}^{\mathbf{H}}$ **do**
4:     **if** $\mathcal{D}^a$ accepts τ_i **then**
5:         $f(a) \leftarrow f(a) + 1$
6:     **end if**
7: **end for**
8: Set $f(a) = 0$ where $\mathcal{T}^{\mathbf{H}}(s, a)$ is invalid
9: **return** a set of actions with the maximum frequency $A^{\mathbf{H}} = \arg\max_{a \in \mathcal{A}^{\mathbf{H}}} f(a)$

---

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. For the equivalence query, it is not feasible for the seeker to compare the learned DFAs with the oracle DFAs it has in mind, as the seeker's strategy $\pi^{\mathbf{S}}$ inherently embeds these oracles. We conduct the equivalence query by querying a bounded number of membership. Once the bound is reached, we conclude that the learned *intent-response DFAs*, denoted as $\mathbf{D} = \{\mathcal{D}^a\}_{a \in \mathcal{A}^{\mathbf{H}}}$ where $\mathcal{D}^a = \left\{ 2^{\mathcal{A}^{\mathbf{S}}}, Q^a, q_0^a, \delta^a, F^a \right\}$, for each helper action $a \in \mathcal{A}^{\mathbf{H}}$, are equivalent to the oracle DFAs.

5.2. 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 $\pi^{\mathbf{H}} : \mathcal{S} \times (\mathcal{A}^{\mathbf{S}})^+ \rightarrow \mathcal{A}^{\mathbf{H}}$ to solve the problem in Section 4, though there is no guarantee of optimality in general.

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.
- $\varrho \colon \mathcal{S} \times \mathbf{a}_{t+1}^{\mathbf{S}} \rightarrow 2^{\mathcal{S}}$ is the transition function, where $\mathbf{a}_{t+1}^{\mathbf{S}} = [a_{t_1}^{\mathbf{S}}, \dots, a_{t_n}^{\mathbf{S}}]$ is the observed seeker's action sequence, such that $\varrho(s, \mathbf{a}_{t+1}^{\mathbf{S}}) = \{\mathcal{T}^{\mathbf{H}}(s, a^{\mathbf{H}}) \mid a^{\mathbf{H}} \in \tau(s, \mathbf{a}_{t+1}^{\mathbf{S}})\}$.
- $\tau \colon \mathcal{S} \times \mathbf{a}_{t+1}^{\mathbf{S}} \rightarrow 2^{\mathcal{A}^{\mathbf{H}}}$ is the output function such that $\tau(s, \mathbf{a}_{t+1}^{\mathbf{S}}) = \text{NCC}(s, \mathbf{a}_{t+1}^{\mathbf{S}}, \mathcal{A}^{\mathbf{H}}, \mathcal{T}^{\mathbf{H}}, \mathbf{D})$. See Algorithm 1.

This construction avoids the exponential blowup of DFA composition by evaluating each DFA independently on extracted segments. Hence, the transducer size is linear in the size of the DFAs, and the cost of obtaining intended actions is also linear in $|\mathcal{A}^{\mathbf{H}}|$. T generates a strategy by allowing the helper to arbitrarily select an action returned by the output function $\tau(s, \mathbf{a}^{\mathbf{S}})$, which provides all equally likely intended actions. The strategy is non-Markovian, as τ depends on the full seeker sequence rather

than just the last state or action.

It is worth noting that every helper's intent-response DFA $\mathcal{D}^a$ in **D** is defined only based on the seeker's actions. Consequently, as long as the seeker utilizes the same policy $\pi^{\mathbf{S}}$ to express its intentions, we can apply these DFAs **D** across various games that share the same action space of both players.

6. Simulation Experiment

Gnomest at Night Testbed. We illustrate the coordination challenge in shared-control games with incomplete information using the setup shown in Figure 1. The left board displays the feasible moves for the seeker, while the right board shows those for the helper. Notably, each player is constrained by their own set of walls, leading to distinct feasible moves for each. In the example shown in Figure 1, the token starts in the top-left corner and must reach the bottom-left goal state. However, the seeker begins inside a T-shaped enclosure that prevents independent progress, making cooperation with the helper essential. For example, when the token is at L1, the helper must move right to L2 to free the seeker from the enclosure. Later, at L3, the helper must move down to L4 so the seeker can continue toward the goal.
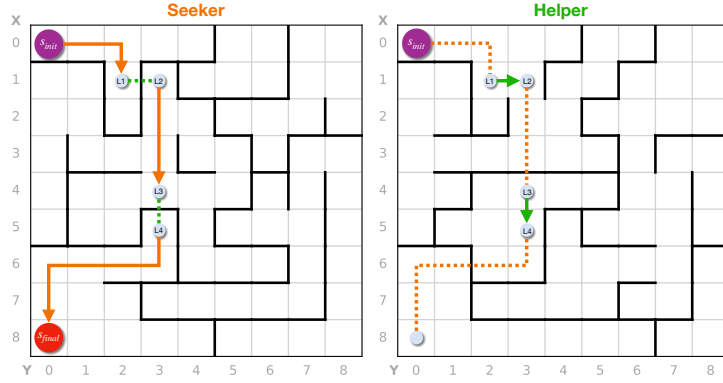

Figure 1: Shared-control game in *Gnomes at Night*^{TM}: Purple dot marks token at s_{init}, red dot marks s_{final}, and maze layouts captured in $\mathcal{T}^{\mathbf{S}}$ (left) and $\mathcal{T}^{\mathbf{H}}$ (right).

Configurations. We evaluate our approach in the *Gnomes at Night*^{TM} testbed [5], where each configuration consists of a maze layout and a treasure location. To assess generalization, we use 10 unseen layouts each 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+).

Baselines. We evaluate three coordination types with three different levels of information exchange: In the **no coordination (NC)** setting, the seeker plans its path using a modified A* algorithm (see Algorithm 2 in the appendix of the full version), while the helper attempts to guess the seeker's desired next action, but due to a lack of communication, its actions are essentially random. With **direct communication coordination (DCC)**, both players have a clear communication channel, allowing the seeker to directly inform the helper of its desired help, which the helper then executes on its turn. In our proposed **no communication coordination (NCC)** setting, the seeker incorporates its required help into its trajectory using the proposed DFA-based approach. The helper interprets the seeker's trajectory and chooses its next action based on the perceived intent. For all conditions, the seeker implementation remains, and only the helper strategy varies.

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); otherwise, it is marked as a failure.

Seeker and Helper Implementations The seeker plans paths with a modified A* algorithm that minimizes wall violations under its own and inferred partner constraints. Upon violation, it replans

Table 1

Number of walls memorized by the seeker and the error rate in its memory across 50 configurations per maze size, comparing No Coordination (NC) and No-Communication Coordination (NCC).

Memorized Metric	9×9 Maze		12×12 Maze	
	NC	**NCC**	NC	**NCC**
Number of Walls	23.27 ± 2.23	$\mathbf{11.79 \pm 1.48}$	40.35 ± 3.64	$\mathbf{12.18 \pm 1.56}$
Wall Error Rate (%)	32.38 ± 12.31	$\mathbf{14.17 \pm 12.36}$	44.39 ± 11.02	$\mathbf{17.42 \pm 21.55}$

and inserts intent-expressive actions. Deviations by the helper trigger belief updates about unknown walls. See Appendix A and B of the full version for details.

To train the helper, we collect 100 trajectories from 9×9 mazes and use the L* algorithm to learn intent-response DFAs for `right`, `up`, `left`, and `down`. Average learning time is under 0.4s. Jaccard similarity [29] with oracle DFAs ranges from 0.58 to 0.80.
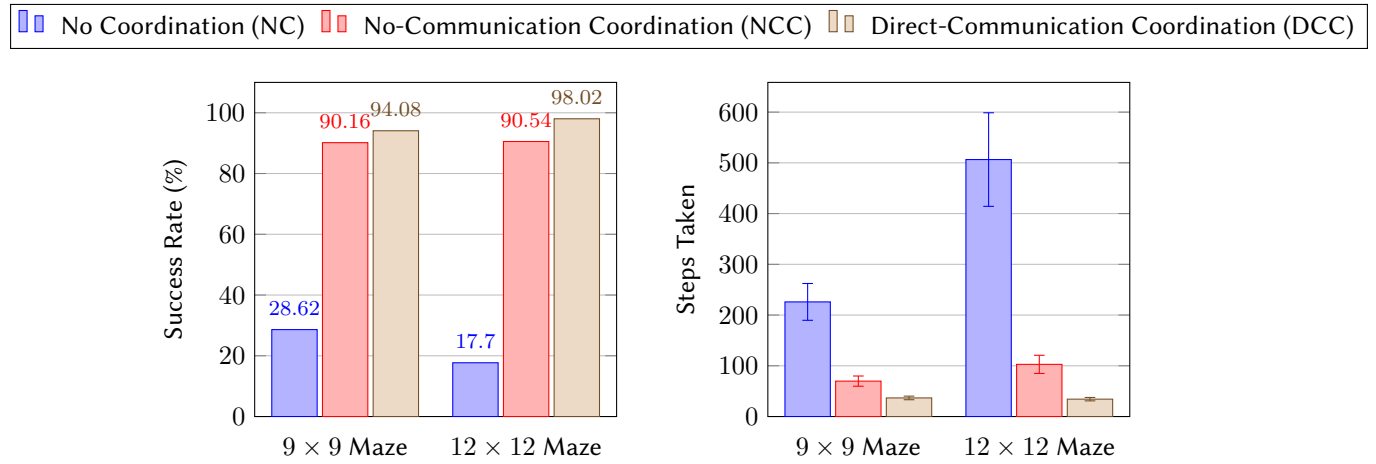

Figure 2: Success rates (left) and steps taken (right) compared across 50 different configurations for 9×9 and 12×12 mazes, illustrating performance across No Coordination (NC), No-Communication Coordination (NCC), and Direct-Communication Coordination (DCC).

Metrics. We report three metrics for each coordination type: (1) **Success rate**, defined as the fraction of successful trials averaged over 50 configurations; (2) **Steps taken**, reported as the mean and standard deviation of steps to termination across trials and configurations; and (3) **Seeker memory**, evaluated by comparing the seeker's memorized wall constraints with the helper's actual maze layout, reporting the mean and standard deviation of both the number of memorized walls and their error rate.

Hypotheses. (H1) NCC outperforms NC in success rate, but underperforms DCC. **(H2)** NCC yields fewer steps than NC, but more than DCC. **(H3)** NCC lowers both the number and error rate of memorized walls compared to NC.

6.1. Results

On H1 (Success Rate). The left plot in Figure 2 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 [30] confirms NCC significantly outperforms NC ($p < 0.001$) in both sizes, while differences between NCC and DCC are not statistically significant ($p > 0.1$). These results not only support H1 but surpass our initial expectations.

On H2 (Steps Taken). The right plot in Figure 2 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 These results support H2.

On H3 (Seeker Memory). Table 1 shows NCC reduces both constraint count and error rate versus NC: by 49.4% and 56.2% in 9×9, and 69.8% and 60.8% in 12×12, respectively. These results support H3, showing NCC minimizes unnecessary exploration and improves intent identification efficiency.

7. Conclusion and Future Work

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 in *Gnomes at Night*^{TM} show that this method approaches the performance of an oracle with direct communication.

Future work includes developing an iterative version that refines the helper's strategy over time, extending from standard reachability to temporal objectives, and adapting to settings with greater non-determinism, such as human or environmental interactions.

Acknowledgments

Declaration on Generative AI

During the preparation of this work, the author(s) used GPT-4o for: Grammar and spelling check, paraphrasing and rewording, and improving writing style. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] A. Cimatti, M. Roveri, P. Traverso, Strong planning in non-deterministic domains via model checking, in: International Conference on Artificial Intelligence Planning Systems, 1998, p. 36–43.

[2] A. Cimatti, M. Pistore, M. Roveri, P. Traverso, Weak, strong, and strong cyclic planning via symbolic model checking, Artificial Intelligence 147 (2003) 35–84.

[3] H. Geffner, B. Bonet, A Concise Introduction to Models and Methods for Automated Planning, Synthesis Lectures on Artificial Intelligence and Machine Learning, Springer Cham, 2013.

[4] A. Dafoe, Y. Bachrach, G. Hadfield, E. Horvitz, K. Larson, T. Graepel, Cooperative ai: Machines must learn to find common ground, Nature 593 (2021) 33–36.

[5] S. Chen, D. Fried, U. Topcu, Human-agent cooperation in games under incomplete information through natural language communication, in: International Joint Conference on Artificial Intelligence, Human-Centred AI track, 2024.

[6] M. Carroll, R. Shah, M. K. Ho, T. Griffiths, S. Seshia, P. Abbeel, A. Dragan, On the Utility of Learning about Humans for Human-AI Coordination, in: Advances in Neural Information Processing Systems, 2019.

[7] N. Bard, J. N. Foerster, S. Chandar, N. Burch, M. Lanctot, H. F. Song, E. Parisotto, V. Dumoulin, S. Moitra, E. Hughes, I. Dunning, S. Mourad, H. Larochelle, M. G. Bellemare, M. Bowling, The hanabi challenge: A new frontier for ai research, Artificial Intelligence 280 (2020).

[8] A. Goil, M. Derry, B. D. Argall, Using machine learning to blend human and robot controls for assisted wheelchair navigation, in: IEEE International Conference on Rehabilitation Robotics, 2013, pp. 1–6.

[9] C. Guan, L. Zhang, C. Fan, Y. Li, F. Chen, L. Li, Y. Tian, L. Yuan, Y. Yu, Efficient human-ai coordination via preparatory language-based convention, 2023. arXiv:2311.00416.

[10] J. Liu, C. Yu, J. Gao, Y. Xie, Q. Liao, Y. Wu, Y. Wang, Llm-powered hierarchical language agent for real-time human-ai coordination, in: International Conference on Autonomous Agents and Multiagent Systems, 2024, p. 1219–1228.

[11] D. Angluin, Learning regular sets from queries and counterexamples, Information and Computation 75 (1987) 87–106.

[12] M. R. Genesereth, M. L. Ginsberg, J. S. Rosenschein, Cooperation without communication, in: Readings in Distributed Artificial Intelligence, Elsevier, 1988, pp. 220–226.

[13] L. Marsh, C. Onof, Stigmergic epistemology, stigmergic cognition, Cognitive Systems Research 9 (2008) 136–149.

[14] R. J. Aumann, Subjectivity and correlation in randomized strategies, Journal of Mathematical Economics 1 (1974) 67–96.

[15] M. Lewis, D. Yarats, Y. Dauphin, D. Parikh, D. Batra, Deal or no Deal? End-to-end Learning of Negotiation Dialogues, in: Conference on Empirical Methods in Natural Language Processing, 2017, pp. 2443–2453.

[16] H. He, D. Chen, A. Balakrishnan, P. Liang, Decoupling Strategy and Generation in Negotiation Dialogues, in: Conference on Empirical Methods in Natural Language Processing, 2018, pp. 2333–2343.

[17] P. Paquette, Y. Lu, S. S. Bocco, M. Smith, S. O-G, J. K. Kummerfeld, J. Pineau, S. Singh, A. C. Courville, No-press Diplomacy: Modeling Multi-Agent Gameplay, in: Advances in Neural Information Processing Systems, 2019.

[18] H. He, A. Balakrishnan, M. Eric, P. Liang, Learning symmetric collaborative dialogue agents with dynamic knowledge graph embeddings, in: Annual Meeting of the Association for Computational Linguistics, 2017, pp. 1766–1776.

[19] J. Foerster, R. Y. Chen, M. Al-Shedivat, S. Whiteson, P. Abbeel, I. Mordatch, Learning with opponent-learning awareness, in: International Conference on Autonomous Agents and MultiAgent Systems, 2018, p. 122–130.

[20] D. Helbing, P. Molnár, Social force model for pedestrian dynamics, Physical Review E 51 (1995) 4282–4286.

[21] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, S. Savarese, Social LSTM: human trajectory prediction in crowded spaces, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 961–971.

[22] H. A. Kautz, J. F. Allen, et al., Generalized plan recognition, in: AAAI National Conference on Artificial Intelligence, 1986, p. 32–37.

[23] M. B. Vilain, Getting serious about parsing plans: A grammatical analysis of plan recognition, in: AAAI National Conference on Artificial Intelligence, 1990, p. 190–197.

[24] E. Charniak, R. P. Goldman, A bayesian model of plan recognition, Artificial Intelligence 64 (1993) 53–79.

[25] N. Lesh, O. Etzioni, A sound and fast goal recognizer, in: International Joint Conference on Artificial Intelligence, 1995, p. 1704–1710.

[26] R. P. Goldman, C. W. Geib, C. A. Miller, A new model of plan recognition, in: Conference on Uncertainty in Artificial Intelligence, 1999, p. 245–254.

[27] D. Avrahami-Zilberbrand, G. A. Kaminka, Fast and complete symbolic plan recognition, in: The International Joint Conference on Artificial Intelligence, 2005, p. 653–658.

[28] M. Ramırez, H. Geffner, Plan recognition as planning, in: The International Joint Conference on Artificial Intelligence, 2009, p. 1778–1783.

[29] P. Jaccard, Étude comparative de la distribution florale dans une portion des alpes et des jura, Bull Soc Vaudoise Sci Nat 37 (1901) 547–579.

[30] H. B. Mann, D. R. Whitney, On a test of whether one of two random variables is stochastically larger than the other, The annals of mathematical statistics (1947) 50–60.