# Human-Agent Cooperation in Games under Incomplete Information through Natural Language Communication[*]

**Shenghui Chen**[1], **Daniel Fried**[2] and **Ufuk Topcu**[1]
[1]University of Texas at Austin,
[2]Carnegie Mellon University
{shenghui.chen, utopcu}@utexas.edu, dfried@cs.cmu.edu

## Abstract

Developing autonomous agents that can strategize and cooperate with humans under information asymmetry is challenging without effective communication in natural language. We introduce a *shared-control game*, where two players collectively control a *token* in alternating turns to achieve a common objective under incomplete information. We formulate a policy synthesis problem for an autonomous agent in this game with a human as the other player. To solve this problem, we propose a communication-based approach comprising a language module and a planning module. The language module translates natural language messages into and from a finite set of *flags*, a compact representation defined to capture player intents. The planning module leverages these flags to compute a policy using an *asymmetric information-set Monte Carlo tree search with flag exchange* algorithm we present. We evaluate the effectiveness of this approach in a testbed based on Gnomes at Night, a search-and-find maze board game. Results of human subject experiments show that communication narrows the information gap between players and enhances human-agent cooperation efficiency with fewer turns.

## 1 Introduction

Developing autonomous agents capable of cooperative strategic planning in games under incomplete information is an important problem in human-agent interaction. Agents with such capabilities hold the potential to improve how they work with humans in a diverse range of settings, from virtual applications like strategy board games [Bard *et al.*, 2020] and action video games [Carroll *et al.*, 2019] to physical applications like assistive wheelchairs [Goil *et al.*, 2013].

In games where the information available to each player is different or incomplete, cooperation between players is challenging without an effective exchange of information. This

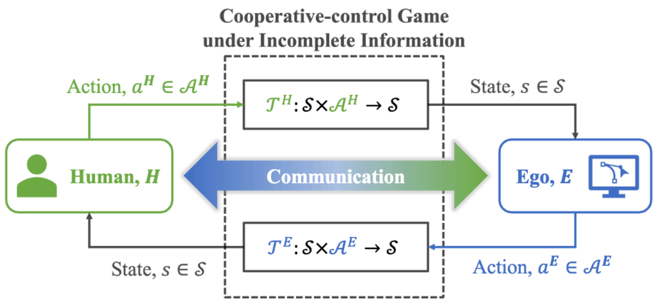

Cooperative-control Game under Incomplete Information
Action, $a^H \in \mathcal{A}^H$
$\mathcal{T}^H: \mathcal{S} \times \mathcal{A}^H \to \mathcal{S}$
State, $s \in \mathcal{S}$
Human, $H$
Communication
Ego, $E$
$\mathcal{T}^E: \mathcal{S} \times \mathcal{A}^E \to \mathcal{S}$
State, $s \in \mathcal{S}$
Action, $a^E \in \mathcal{A}^E$

Figure 1: An illustration of the shared-control game in Definition 1.

information asymmetry, often resulting from a partial observation of underlying game states or an incomplete grasp of overall game dynamics, can lead to actions that are either misaligned with the team's objectives or suboptimal for coordination. To overcome this challenge, it is crucial for players to engage in effective communication to exchange relevant information at runtime. If only autonomous agents are playing such games, they can accomplish this exchange in a structured manner, for example, with limited hints in the game of Hanabi [Bard *et al.*, 2020].

When an autonomous agent needs to partner with a human player, this challenge of effective information exchange becomes more pronounced. Humans typically rely on natural language for communication, a medium that is rich, nuanced, and inherently more complex than the structured data formats that autonomous agents might use. Therefore, successful collaboration between humans and agents often requires more than communication in structured representations—it demands communication in natural language. This requirement introduces an additional layer of complexity, as agents must be able to understand and extract contextually relevant information from natural language communication.

Consider a motivating scenario: an autonomous robot and a human coordinator team up in a search-and-rescue mission, navigating hazardous environments to reach a target. Both the robot and the human have incomplete information: the robot has sensor data of its immediate environment but lacks overall mission context, while the coordinator has an overview plan but no access to the robot's detailed sensor data.

*Gnomes at Night*, a cooperative search-and-find maze board game, offers a more manageable scenario with simi-

[*]The source code and the full paper with appendix can be found at https://github.com/vivianchen98/shared_control_language.

lar characteristics [Peaceable Kingdom, 2016]. In this board game, two players collaborate to move two magnetically connected gnome pieces through a maze board to collect treasures. The maze paths are different on each side of the board, and players can only move the gnome pieces along paths visible on their side. While they cannot move the gnome pieces through walls on their own side, they can move them through walls on the opposite side. Both players receive identical rewards when reaching a treasure.

To model the described scenarios, we introduce a two-player, turn-based game called a *shared-control game*, where players collectively control a single *token* for a common objective under incomplete information. This game features a shared state space and individual action spaces. Players take turns to move the token to new states according to their private deterministic transition functions, reflecting their respective understanding of the game dynamics. It is worth highlighting that each player does not know the transition function of the other player, creating a form of incomplete information different from what is typically encountered due to partial game state observations as seen in decentralized partially-observable Markov decision processes [Bernstein *et al.*, 2002]. Finally, we augment the game with a reward function to incentivize player cooperation in achieving the common objective. This game abstracts specific scenarios, offering a general framework to think about teamwork in situations under incomplete information.

In this paper, we pose the following problem in shared-control games: Knowing the other player is a human, how can an autonomous agent make strategic decisions to achieve smooth cooperation despite lacking complete information on the game dynamics? We hypothesize that allowing players to exchange information via natural language communication could narrow the information gap between them and lead to more efficient cooperation than without communication.

To test this hypothesis, we introduce a simple testbed based on Gnomes at Night. This testbed is a discretized version of the board game where the maze is a $9 \times 9$ gridworld, and players move in single steps. As illustrated in Figure 2, the token is the magnetically connected gnome pieces, and two players see board sides with different wall layouts. This testbed has five rounds, each featuring a different treasure location known exclusively to one player.

We propose an approach that uses natural language communication to inform an online planning algorithm. This approach includes language and planning modules interfaced through a representation of player intents called *flags*. The language module translates communication messages into flags using a large language model. The planning module then determines the next action based on the input flag and the current game state and optionally generates a new flag.

We present the *asymmetric information-set Monte Carlo tree search with flag exchange* (AISMCTS-F) algorithm, which determines the flag-based policy for the planning module. This algorithm constructs perspective-based decision trees for each player in a turn-based manner. An ego player only selects valid actions on its turn, and it assumes initially all actions are valid for the other player since it does not know the other player's transition function. A *hidden information dictionary*, stored by the ego player, records actions the other player has rejected to improve subsequent decision-making by avoiding these actions. AISMCTS-F also prioritizes actions that match the other player's communicated preferences, indicated by an input flag, when faced with equally optimal choices. Lastly, the algorithm expresses its current best estimate of what next action the other player should take via an output flag.

To assess the effectiveness of the proposed approach, we conduct human subject experiments in the Gnomes at Night testbed. We recruit human participants to play with either another human, a communication-enabled agent implemented via the proposed approach, or an agent without communication capabilities. Results indicate that enabling communication enhances human-agent cooperation efficiency, reaching treasures in fewer turns and less time per round. However, even with improvements via the proposed approach, human-agent cooperation still falls short of the efficiency seen in human-to-human gameplay, suggesting potential room for future research. Additionally, We observe distinct communication patterns in human-agent interactions compared to those between humans.

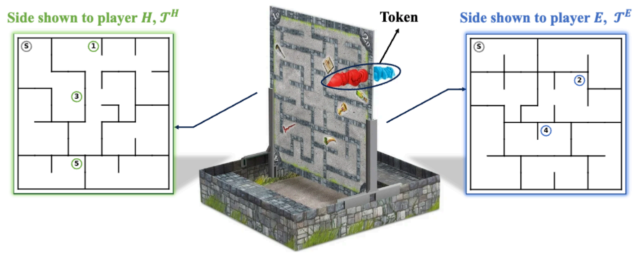

Side shown to player $H$, $\mathcal{T}^H$
Token
Side shown to player $E$, $\mathcal{T}^E$
⑤
①
③
⑤
⑤
②
④

Figure 2: Gnomes at Night testbed where the token is the magnetically connected gnome pieces and two private transition functions encode the wall layouts on each side of the board. Middle figure credit to [Peaceable Kingdom, 2016].

## 2 Cooperation in Shared-control Games under Incomplete Information

To model interactions in scenarios described in Section 1, we introduce a two-player, turn-based game, called a *shared-control game*, where players need to collectively control a single *token* to achieve a common objective under incomplete information. Within the disaster response context, the token is the physical robot, and the two players are the robot's decision-making system and the human operator respectively.

**Definition 1.** A shared-control game where player $\mathbf{E}$ and player $\mathbf{H}$ collectively controls a token in alternating turns is a tuple $\Gamma = (\mathcal{S}, s_{\text{init}}, s_{\text{final}}, \mathcal{A}^{\mathbf{E}}, \mathcal{A}^{\mathbf{H}}, \mathcal{T}^{\mathbf{E}}, \mathcal{T}^{\mathbf{H}})$ where $\mathcal{S}$ is a finite set of states, $s_{\text{init}} \in \mathcal{S}$ is an initial state, $s_{\text{final}} \in \mathcal{S}$ is a final state, $\mathcal{A}^i$ is a finite set of actions available to each player $i \in \{\mathbf{E}, \mathbf{H}\}$, and $\mathcal{T}^i : \mathcal{S} \times \mathcal{A}^i \to \mathcal{S}$ is a deterministic transition function for each player $i \in \{\mathbf{E}, \mathbf{H}\}$. We refer to player $\mathbf{E}$ as the *ego player*.

The shared-control game $\Gamma$ starts with the token in the initial state $s_{\text{init}} \in \mathcal{S}$. Players $\mathbf{E}$ and $\mathbf{H}$ take turns to move the token. Either player can initiate the game by taking an action during the first turn. During the turn of a player $i$, whom we refer to as the *player in control*, it observes the current state of the token in $s \in \mathcal{S}$ and selects an action $a$ from its own action space $\mathcal{A}^i$. Then, the token moves to a new state $s'$ according to the transition function $\mathcal{T}^i(s, a)$ of this player in control. Upon transitioning to the new state $s'$, control passes to the other player, i.e., if **H** just played, then **E** will take the next turn, and vice versa. The game continues with players alternating turns until a final state $s_{\text{final}} \in \mathcal{S}$ is reached.

Without loss of generality, assume player **H** initiates the game, a path through $\Gamma$ is a sequence $s_0 \xrightarrow{a_1^{\mathbf{H}}} s_1 \xrightarrow{a_2^{\mathbf{E}}} \cdots$ such that $s_t \in \mathcal{S}$, $a_{t+1}^i \in \mathcal{A}^i$, and $s_{t+1} = \mathcal{T}^i(s_t, a_{t+1}^i)$. We assume there exists at least one finite path $s_0 \xrightarrow{a_1^{\mathbf{H}}} s_1 \xrightarrow{a_2^{\mathbf{E}}} \cdots \rightarrow s_T$ where $s_0 = s_{\text{init}}$ and $s_T = s_{\text{final}}$.

**Incomplete Information.** In this game, players have incomplete information about the game dynamics. Each player $i$ only knows its own transition function $\mathcal{T}^i$ but not that of the other player. In this paper, we also let the final state $s_{\text{final}}$ be visible to only one player.

**Common Reward.** We augment $\Gamma$ with a common reward function $\mathcal{R} : \mathcal{S} \times \cup_{i \in \{\mathbf{E},\mathbf{H}\}} \mathcal{A}^i \rightarrow \mathbb{R}$ that assigns equal real-valued reward $\mathcal{R}(s, a)$ to both players following the execution of action $a$ by either player in state $s$. This function can capture cooperative objectives, such as reaching a final state in as few turns as possible.

In the *Gnomes at Night* testbed, the state space is the maze grid itself, and both players have the same set of actions: `noop`, `right`, `up`, `left`, `down`. Each player only sees one side of the board, captured by a private transition function. Players receive equal rewards for finding a treasure and incur penalties for hitting walls or taking excessive steps. This testbed lays the foundation for exploring the problem of cooperative policy synthesis under incomplete information.

**Cooperative Policy Synthesis.** In a shared-control game, we study the problem of computing cooperative policies for an autonomous agent as the ego player **E**, with a human as the other player **H**. We allow communication between players via the exchange of messages. Let $M$ be a message space, we model the behavior of player $i$ as a message-based policy $\pi^i : \mathcal{S} \times M \rightarrow \mathcal{A}^i \times M$.

**Problem 1.** In a shared-control game $\Gamma$ with a common reward function $\mathcal{R}$, and given a human player whose behavior is denoted as $\pi^{\mathbf{H}}$, to compute the ego player's policy $\pi^{\mathbf{E}}$ that maximizes the cumulative total reward while adhering to the game transitions:

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

$$\text{s. t.} \quad s_0 = s_{\text{init}}, \tag{1b}$$

$$\text{for all } t \geq 0,$$

$$\begin{cases} a_{t+1}^{\mathbf{H}}, m_{t+1}^{\mathbf{H}} = \pi^{\mathbf{H}}(s_t, m_t^{\mathbf{E}}) & \text{on } \mathbf{H}\text{'s turn}, \\ a_{t+1}^{\mathbf{E}}, m_{t+1}^{\mathbf{E}} = \pi^{\mathbf{E}}(s_t, m_t^{\mathbf{H}}) & \text{on } \mathbf{E}\text{'s turn}, \end{cases} \tag{1c}$$

$$s_{t+1} = \mathcal{T}^i(s_t, a_{t+1}^i), \quad \text{on player } i\text{'s turn}. \tag{1d}$$

Here $m_t^i$ represents the message sent by player $i$ on turn $t$, and $T$ is the turn at which the final state $s_{\text{final}}$ is reached. We highlight the messages sent by the ego player and the human player in blue and green respectively. Each player decides its next move considering the message sent by the other player in the last turn. This problem generalizes a "mute version", where messages $m_t^i$ remain unspecified for both players throughout all turns, offering a baseline comparison for understanding the value of communication.

## 3 Related Work

Research from various fields is relevant to our problem. This section provides a non-exhaustive overview on two fronts.

**Cooperative Games.** Several strategic games have stood out as testing environments for enhancing player cooperation, including the card game *Hanabi* [Bard *et al.*, 2020], the action video game *Overcooked* [Carroll *et al.*, 2019], and dialogue games involving negotiation (e.g., *Deal-or-No-Deal* [Lewis *et al.*, 2017; He *et al.*, 2018] and *Diplomacy* [Paquette *et al.*, 2019]) and coordination (e.g., *MutualFriends* [He *et al.*, 2017]). However, *Gnomes at Night* features a unique set of challenges and game dynamics. Unlike Overcooked, which operates under complete information, Gnomes at Night, similar to Hanabi, is characterized by incomplete information. However, the nature of this incompleteness is different: it is due to private transition functions in Gnomes at Night instead of partial observation of game states in Hanabi. Additionally, Gnomes at Night requires natural language communication, significantly different from the hint-based communication in Hanabi and the non-verbal coordination in Overcooked. Unlike negotiation games, Gnomes at Night focuses on pure cooperation and collaborative problem-solving, without deceit or manipulation. Finally, other dialogue coordination games usually only require coordination at the end of the dialogue, while our setting requires players to cooperate on each turn by sharing control of a single token.

**MCTS-based Planning Techniques.** Monte Carlo tree search (MCTS) is an algorithm that combines tree-based search with Monte Carlo random sampling to efficiently explore and evaluate vast decision spaces [Browne *et al.*, 2012]. MCTS has proven to be effective in competitive strategic games, such as chess [Campbell *et al.*, 2002], Shogi [Silver *et al.*, 2018], and Go [Silver *et al.*, 2016; Silver *et al.*, 2017], making it ideal as the foundation of our planning module. However, standard MCTS works best for games with complete information, but we need to deal with games under incomplete information. Thus, we adopt the information-set MCTS (ISMCTS), which maintains perspective-based trees for each player, whose nodes correspond to players' information sets and edges correspond to moves from that player's viewpoint [Cowling *et al.*, 2012]. Recent studies have also demonstrated the effectiveness of MCTS-based algorithms in cooperative game contexts, such as in Settlers of Catan [Szita *et al.*, 2010], The Resistance [Cowling *et al.*, 2015], and no-press Diplomacy [Jacob *et al.*, 2022]. Another work enhanced ISMCTS by introducing re-determinizing methods to prevent hidden information leakage for Hanabi [Goodman,

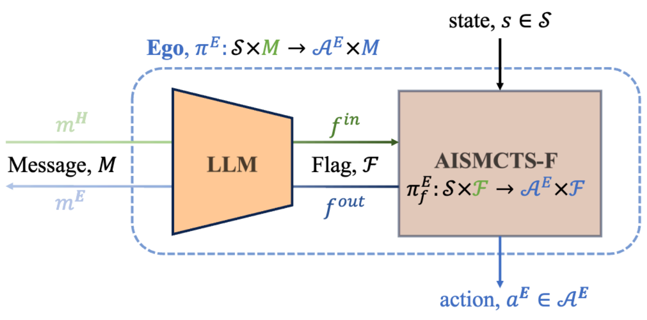

state, $s \in \mathcal{S}$
Ego, $\pi^E: \mathcal{S}\times M \to \mathcal{A}^E \times M$
$m^H$
Message, $M$
$m^E$
LLM
$f^{in}$
Flag, $\mathcal{F}$
$f^{out}$
AISMCTS-F
$\pi_f^E: \mathcal{S}\times\mathcal{F} \to \mathcal{A}^E\times\mathcal{F}$
action, $a^E \in \mathcal{A}^E$

Figure 3: Communication-based approach that generates $\pi^{\mathbf{E}} : \mathcal{S} \times M \to \mathcal{A}^{\mathbf{E}} \times M$ via a language module (orange trapezoid) and a planning module (brown rectangle) interfaced through flags.

2019]. These works highlight MCTS's versatility in not only competitive but also cooperative settings.

## 4 Method

This section outlines a communication-based approach that solves Problem 1 by leveraging information exchange through natural language communication. It processes a human partner's message and the current state to determine the ego player's next action and optionally generates a responding message. As shown in Figure 3, this approach consists of a language module harnessing a large language model and a planning module based on Monte Carlo tree search, interfaced through a finite set of flags that capture player intents expressed in natural language messages.

We define a *flag* as a compact representation of intents. In a shared-control game, the flag space $\mathcal{F}$ is the union of both players' action spaces and essential responses one player might have to an intent the other player expresses, including acceptance, rejection, inquiry, and no response, i.e.,

$$\mathcal{F} = \mathcal{A}^{\mathbf{H}} \cup \mathcal{A}^{\mathbf{E}} \cup \{\texttt{Accept}, \texttt{Reject}, \texttt{Inquiry}, \texttt{None}\}. \tag{2}$$

Concretely, the flag space in the Gnomes at Night testbed is

$$\begin{aligned}\mathcal{F} = \{&\texttt{noop}, \texttt{right}, \texttt{up}, \texttt{left}, \texttt{down},\\ &\texttt{Accept}, \texttt{Reject}, \texttt{Inquiry}, \texttt{None}\}.\end{aligned} \tag{3}$$

With this representation defined, we model the ego player's behavior as a flag-based policy $\pi_f^{\mathbf{E}} : \mathcal{S} \times \mathcal{F} \to \mathcal{A}^{\mathbf{E}} \times \mathcal{F}$.

### 4.1 Planning Module: AISMCTS-F

This subsection presents an algorithm for the planning module, called *Asymmetric Information-Set Monte Carlo Tree Search with Flag exchange* (AISMCTS-F), that computes a flag-based policy $\pi_f^{\mathbf{E}}$ from the perspective of an ego player **E**, detailed in Algorithm 1.

To explain this algorithm, we introduce these notations: a node $v$ keeps track of its parent $p(v)$, children $C(v)$, incoming action $a(v)$, current state $s(v)$, total reward $T(v)$, and visit count $N(v)$.

Inspired by ISMCTS [Cowling *et al.*, 2012], this algorithm first initializes two trees with root nodes $v_0^i$ for each player $i$. Then the main skeleton of this algorithm (lines 1–15) unfolds in four phases:

**Algorithm 1** AISMCTS-F from player **E**'s perspective

```
Memory: hidden information dictionary $\Omega$, last flag $f^{last}$
Parameter: $\Gamma = (\mathcal{S}, s_{\text{init}}, s_{\text{final}}, \mathcal{A}^{\mathbf{E}}, \mathcal{A}^{\mathbf{H}}, \mathcal{T}^{\mathbf{E}}, \mathcal{T}^{\mathbf{H}}), \mathcal{R}, \sigma^{\mathbf{E}}$, and the number of iteration $n$
Input: current state $s_r \in \mathcal{S}$, input flag $f^{in} \in \mathcal{F}$,
Output: action $a^\star \in \mathcal{A}^{\mathbf{E}}$, output flag $f^{out} \in \mathcal{F}$
1:  Create single-node trees with roots $v_0^{\mathbf{E}}$ and $v_0^{\mathbf{H}}$ respectively, and initialize onRollout = False
2:  for $n$ iterations do
3:      Set $r = 0, s = s_r$, denote the player in control as $i$
4:      while $s \neq s_{\text{final}}$ do
5:          $a$ = EXPLORE($v^i, s$)                  ▷ Selection/Simulation
6:          $r \leftarrow r + \mathcal{R}(s, a)$
7:          $s \leftarrow \mathcal{T}^i(s, a)$
8:          for each player $i$ do                   ▷ Expansion
9:              $v^i \leftarrow$ FINDORCREATECHILD($v^i$, $a$)
10:         $i \leftarrow$ the other player
11:     for each player $i$ do                       ▷ Backpropagation
12:         BACKPROPAGATE($v^i$, $r$)
13:     Reset $v^{\mathbf{E}} \leftarrow v_0^{\mathbf{E}}, v^{\mathbf{H}} \leftarrow v_0^{\mathbf{H}}$
14:     Set onRollout $\leftarrow$ False
15: return SELECTBESTACTION($v_0^{\mathbf{E}}, f^{in}$)

16: function FINDORCREATECHILD($v$, $a$)
17:     if not onRollout then
18:         Instantiate a node $u$ whose $p(u) = v, a(u) = a$
19:         if $u \notin C(v)$ then
20:             Add $u$ to $C(v)$
21:             Set $v \leftarrow u$
22:             Set onRollout $\leftarrow$ True
23:         return $u$

24: function BACKPROPAGATE($v$, $r$)
25:     $N(v) \leftarrow N(v) + 1$
26:     $T(v) \leftarrow T(v) + r$
27:     if $p(v)$ exists then
28:         BACKPROPAGATE($p(v)$, $T(v)$)
```

- *Selection*: For each iteration, the algorithm traverses both trees from the root to a leaf node by selecting nodes that maximize the upper confidence bound (ucb) value [Kocsis and Szepesvári, 2006]:

$$\text{ucb}(v) = \frac{T(v)}{N(v)} + c \cdot \sqrt{\frac{\log(N(p(v)))}{N(v)}}, \tag{4}$$

  where $c$ balances exploration and exploitation[1].

- *Expansion*: Upon reaching a leaf node that has not expanded all possible actions, the algorithm chooses one of these unexplored actions at random and adds a new child node for this action in both trees if such a node does not already exist.

- *Simulation*: From new nodes, the algorithm simulates playouts using a rollout policy until the game ends.

[1]We use ucb for its computational efficiency and set $c = \sqrt{2}$.

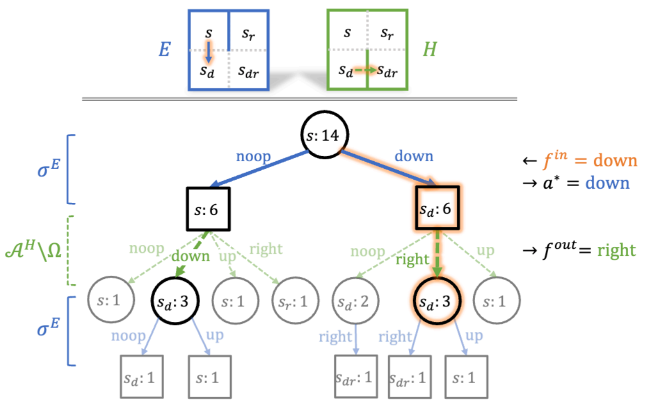


Figure 4: AISMCTS-F: a tree from the perspective of an ego player **E** in a minimal maze example shown at the top.

- *Backpropagation*: Then in both trees, the algorithm backpropagates the results to update the statistics of the nodes visited during this iteration.

After $n$ iterations[2], the algorithm returns the action associated with a child node that has been visited the most from among children of the root node $v_0^{\mathbf{E}}$ in the ego player's tree.

In the proposed algorithm, we modify two key functions, EXPLORE and SELECTBESTACTION, to utilize the information exchanged in communication. We explain these two functions with a tree from the perspective of an ego player in a minimal maze example, as shown in Figure 4.

**EXPLORE.** This function is responsible for choosing an action in both the *Selection* and *Simulation* phases. In a shared-control game, each player has a private transition function that is inaccessible to the other player. Each private transition function $\mathcal{T}^i$ induces a set of valid actions in any given state $s \in \mathcal{S}$, which we denote as

$$\sigma^i(s) = \{a \in \mathcal{A}^i | \mathcal{T}^i(s,a) \text{ is defined}\}. \tag{5}$$

A player **E** thus only knows $\sigma^{\mathbf{E}} \subseteq \mathcal{A}^{\mathbf{E}}$ but not $\sigma^{\mathbf{H}}$, because such sets cannot be defined without the knowledge of $\mathcal{T}^{\mathbf{H}}$.

This algorithm starts with the ego player assuming all actions available to the other player are valid. Accordingly, as illustrated in Figure 4, the ego player's tree expands in an interleaved way: the branching in blue from ego-player-controlled nodes (circles) is less extensive than branching in green from human-player-controlled nodes (squares). This difference occurs because the ego player has less information about the human player's transition function than its own.

We introduce a *hidden information dictionary*, stored in the ego player's memory, as a structure to record the information accumulated through interaction about the transition function of the other player. For player **E**, We denote this structure as

$$\Omega : \mathcal{S} \rightarrow 2^{\mathcal{A}^{\mathbf{H}}}, \tag{6}$$

where $\Omega(s) \subseteq \mathcal{A}^{\mathbf{H}}$ is the set of actions rejected by player **H** at state $s$. In the example shown in Figure 4, the algorithm starts the tree-construction with

$$\Omega = \{s : [\texttt{left}], s_d : [\texttt{left}, \texttt{down}]\}. \tag{7}$$

[2]We set $n = 100$ to meet the real-time requirements in this paper.

**Algorithm 1** (continued)

```
26: function EXPLORE($v^i$, $s$)
27:     if not onRollout then                                ▷ Selection
28:         if $\Sigma(v^i, s)$ is not instantiated then
29:             $\Sigma(v^i, s) = \begin{cases} \mathcal{A}^{\mathbf{H}} \setminus \Omega(s), & \text{if } i = \mathbf{H}, \\ \sigma^{\mathbf{E}}(s), & \text{if } i = \mathbf{E}. \end{cases}$
30:         if $\Sigma(v^i, s) \neq \emptyset$ then
31:             return $a \leftarrow \text{pop}(\Sigma(v^i, s))$
32:         return $a(c^*)$ where $c^* = \arg\max_{c \in C(v^i)} \text{ucb}(c)$
33:     else                                                 ▷ Simulation
34:         return $a \in_{\text{random}} \begin{cases} \mathcal{A}^{\mathbf{H}} \setminus \Omega(s), & \text{if } i = \mathbf{H}, \\ \sigma^{\mathbf{E}}(s), & \text{if } i = \mathbf{E}. \end{cases}$

35: function SELECTBESTACTION($v_0^{\mathbf{E}}$, $f^{in}$)
36:     if $f^{in}$ = Inquiry then
37:         $f^{out}$ = Inquiry
38:     else if $f^{in}$ = REJECT then
39:         Add $f^{last}$ to $\Omega(s(v_0^{\mathbf{E}}))$
40:     else if $f^{in} \notin \sigma^{\mathbf{E}}(s(v_0^{\mathbf{E}}))$ then
41:         $f^{out}$ = Reject
42:     Compute $\mathcal{C} = \arg\max_{c \in C(v_0^{\mathbf{E}})} N(c)$
43:     if $\exists c^f \in \mathcal{C}$ s.t. $a(c^f) = f^{in}$ then
44:         $a^\star \leftarrow f^{in}$, $c^\star \leftarrow c^f$
45:     else
46:         Choose $c^\star \in \mathcal{C}$ , $a^\star \leftarrow a(c^\star)$
47:     if $f^{out}$ not already set then
48:         Compute $\mathcal{G} = \{g \in C(c^\star) | a(g) \notin \Omega(s(c^\star))\}$
49:         if $\mathcal{G} = \emptyset$ then
50:             $f^{out}$ = NONE
51:         else
52:             $f^{out} = a(g^\star)$ where $g^\star \in \arg\max_{g \in \mathcal{G}} N(g)$
53:     return $a^\star$, $f^{out}$, and set $f^{last} \leftarrow f^{out}$
```

This piece of information can be useful for reducing the branching factor at the human-player-controlled nodes. Essentially, if the token steps into a visited state where the human has rejected actions $\Omega(s)$, the algorithm will refrain from expanding the corresponding edges as it now believes these actions are unfeasible and not worth exploring. For example, since the human has rejected `left` at state $s$ and `left`, `down` at state $s_d$, these corresponding edges are absent in the example tree. In both the *Selection* and *Simulation* phases, we use the hidden information dictionary via $\mathcal{A}^{\mathbf{H}} \setminus \Omega(s)$ (see lines 29 and 34).

**SELECTBESTACTION.** This function realizes the flag exchange in the algorithm: given the root node whose state $s(v_0^{\mathbf{E}})$ is the current state in gameplay and an input flag $f^{in} \in \mathcal{F}$, this function operates in three stages to compute the next action $a^\star$ and an output flag $f^{out} \in \mathcal{F}$.

In the initial stage (lines 36–41), the function processes three flag types. For `Inquiry`, it recognizes a question from the human player, setting the output flag $f^{out}$ to `Inquiry` too and directing the language module to formulate a response considering the current game information. When encountering `Reject`, it interprets the human player's response as a refusal of the last proposed action $f^{last}$ and updates the hidden information dictionary accordingly. Lastly, if faced with an invalid action, it issues a `Reject` flag, prompting the language module to inform the human player of an obstructing wall, conveying hidden information from the ego player to the human player.

The second stage (lines 42–46) in the function selects a node $c^\star$ from among the root node's children and returns the action leading to that node $a^\star$ as the ego player's next move. The function first finds out a set of child nodes with the highest visit count as $\mathcal{C}$ (see line 42). The actions leading up to these nodes are equally optimal. Input flags can serve to break ties, steering the ego player towards an action preferred by the human. For instance, in Figure 4, when both `noop` and `down` lead to nodes with equal visit counts, an input flag favoring `down` would guide the selection towards this action, highlighted by a path in orange glow. Absent a relevant input flag, the function may randomly select a node among $\mathcal{C}$ and return the action leading to this node.

In the final stage (lines 47–52), if the output flag $f^{out}$ is neither set to `Inquiry` nor `Reject`, the function generates an output flag based on an action associated with a child node of the previously chosen node $c^\star$, pruning any nodes with actions listed in the hidden information dictionary at $c^\star$'s state (see line 48). The output flag is the action leading to the highest visit count node in the pruned list, given such a list is not empty. In the minimal example shown in Figure 4, $f^{out}$ is `right`, which says the ego player wants the other player to move right next. However, as this move is in fact not allowed in its maze, player **H** sends a `Reject` flag. The algorithm will then update the hidden information dictionary to include `right` at state $s_d$.

In the worst-case scenario, the runtime of Algorithm 1 is $O\left(n(l + \log|\mathcal{S}| + |\mathcal{A}^{\mathbf{E}}| + |\mathcal{A}^{\mathbf{H}}|)\right)$, where $n$ is the number of iterations and $l$ is the average length of simulations.

### 4.2 Language Module

For parsing a human message into an intent flag, we first handle cases of empty or absent messages by returning a `None` flag. Subsequently, we employ few-shot prompting [Brown *et al.*, 2020] with `gpt-3.5-turbo` to categorize messages into flags. Our prompts use 6 examples that pair messages and intent flags, with the specific prompt detailed in the supplementary material.

To generate messages based on an intent flag, we use templated sentences as well as GPT-generated responses. For action flags, our messages request human interaction using the template *"Can you {}?"*. For `Reject` flags, which indicates an action proposed by the human player is invalid for the ego player at the current state, we notify the human with the template *"I cannot {} because there is a wall in that direction."* to explain the obstruction. For `Inquiry` flags, we call `gpt-3.5-turbo` with current game information to generate responses to specific questions. Implementation details for this are also available in the supplementary material.

## 5 Human Subject Experiment

To assess the effectiveness of the proposed communication-based approach, we conduct human subject experiments with the following hypotheses:

**H1.** Using a large language model for inferring intents is more accurate than for predicting next actions directly.

**H2.** Enabling information exchange via natural language communication improves cooperation efficiency.

**Independent Variables.** We recruit participants to play as the human player and vary the ego player among three types: another human (***H***), a communication-enabled agent (***A-Comm***) implemented with the proposed approach, and a mute agent (***A-Mute***) implemented with the same planning module but without communication.

**Dependent Measures.** We use the number of turns as a proxy measure for cooperation efficiency in this turn-based game and measure the time to complete each round. We also analyze the quantity and length of messages both players sent during gameplay.

**Experiment Design.** We employ a between-subject design, where each participant interacts exclusively with one type of ego player. Upon giving consent, participants are asked to play five rounds of Gnomes at Night on our testbed website, each displaying a different treasure location.

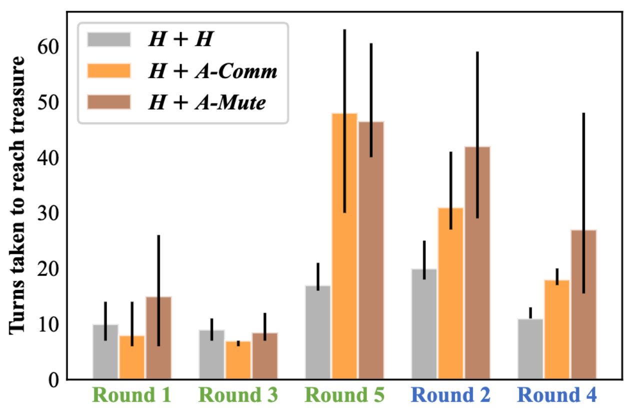
**Participants.** We recruit a total of 150 consenting participants on Prolific [Palan and Schitter, 2018] to play with the three types of ego players. The average age is 34.71, with a gender ratio of 0.53 female.

### 5.1 Results and Discussion

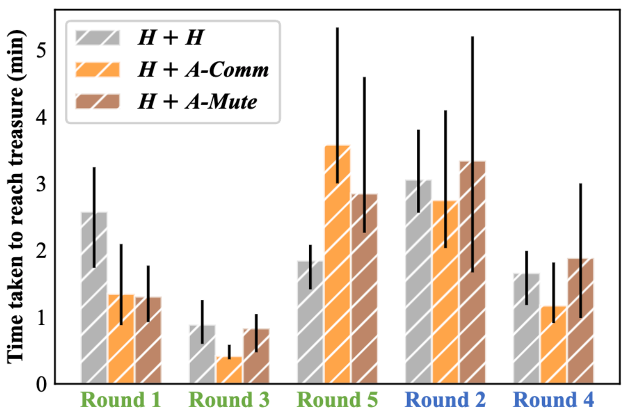
**Regarding H1.** From the human-to-human gameplay data collected, we pick out 286 pairs of conversation snippets and their corresponding subsequent actions. We then annotate snippets with intent flags. Also using few-shot prompting, we construct prompts with 10% of the tuples and evaluate our chosen LLM, `gpt-3.5-turbo`, with the remaining 90% of the tuples. The details are in the supplementary material. Our evaluation demonstrates that predicting flags achieves an accuracy of 74.32%, higher than the 53.31% accuracy of directly predicting the next action. This result supports hypothesis H1 and justifies our proposed approach.

**Regarding H2.** Figure 5a shows that teams of humans and communication-enabled agents ($H$ + *A-Comm*) outperform the teams of humans and mute agents ($H$ + *A-Mute*) in 4 out of 5 rounds by taking fewer turns to reach the treasure. An ANOVA analysis of three groups revealed a significant difference in the number of turns taken ($F(2, 627) = 10.53, p < .01$), with post-hoc Tukey HSD testing confirming that $H$ + *A-Comm* complete rounds with fewer turns than $H$ + *A-Mute* ($p < .05$). These results support hypothesis H2.

**Analysis on Completion Time.** We also measure the time taken to complete each round. Results in Figure 5b show $H$ + *A-Comm* reaches treasures with less or comparable time than $H$ + *A-Mute* in 4 out of 5 rounds, highlighting the value of communication considering players do not need to spend time sending messages in the latter case. The same figure also shows $H$+*A-Comm* outpaces $H$+$H$ in 4 out of 5 rounds. However, this may not be a reliable indicator that cooperation between humans and communication-enabled agents reaches



(a) Turns taken.



(b) Time taken (min).

Figure 5: Bar plots of median cooperation efficiency measures, with 95% confidence intervals from bootstrap resampling with 10,000 samples. Round 1, 3, 5 (green): treasure visible to human player; round 2, 4 (blue): treasure visible to ego player (see Figure 2).

the efficiency of human-to-human cooperation, since humans may take more time to type messages.

**Round 5 as a Special Case.** Both turns and time taken results deviate from our hypothesis in round 5, as depicted in the left maze of Figure 6, where a spanning wall blocks all access to the treasure. We speculate that this unique treasure position necessitates more complex strategy coordination over a longer time horizon than what one-step intent communication allows. Therefore, human teams significantly outperform both types of human-agent pairs. The slight performance gap between agents that can communicate and those that cannot could simply be due to the time spent on communication.

**Communication Paradigms.** We analyze the average number of messages each player sent during gameplay. The data, presented in Table 1, reveals that humans exchange a similar number of messages with comparable lengths. In contrast, the communication-enabled agent sends around twice as many messages of longer lengths than those sent by its human partner, demonstrating two different communication paradigms. The autonomous agent appears to employ frequent querying as a strategy to obtain more information from its human partner, while human players tend to engage in more equal conversations.

| **Player H / Player E** | *H / H* | *H / A-Comm* |
|---|---|---|
| Average message count | 7.35 / 7.34 | 9.24 / 17.57 |
| Average message length | 14.89 / 11.88 | 11.81 / 25.41 |

Table 1: Average message count and length per round. *H + A-Mute* results are excluded since no messages are allowed.

Figure 6 qualitatively shows the effectiveness of the proposed approach in learning hidden information via communication. By visualizing a heatmap of the hidden information dictionary stored at the end of *H + A-Comm* gameplays, we show that the ego player successfully discerns the general layout of walls on the human player's side. Yet, the same figure also reveals instances of mistakenly identified walls, possibly due to human errors or language parsing inaccuracies.

## 6 Conclusion

In this paper, we introduce a shared-control game where two players take turns to collectively control a token under incomplete information. We formulate a policy synthesis problem in this game for an ego player whose partner is a human and hypothesize that communication can narrow the information gap between players for more efficient cooperation. Our approach combines a language module that translates natural language messages into intent flags and a planning module employing our AISMCTS-F algorithm for informed decision-making. We conduct human subject experiments where 150 consenting participants play with either another human, a communication-enabled agent, or a mute agent. Results show that the communication-enabled agents tend to outperform mute agents in collaboration with a human player.

**Future Work.** Several experiment results point to directions for future research. First, the deviations in round 5 suggest future work should extend the communication mechanism to support strategy-making over a longer time horizon. Second, the false positive information in the heatmap highlights the need for methods more robust to human errors and inaccuracies from language parsing. Finally, while this paper adopts a minimal set of intent flags, the proposed approach is a general framework that should work with any type of intent representation, as long as it can be processed by the planning module. Future work should explore representations that can encode richer or composite intents.

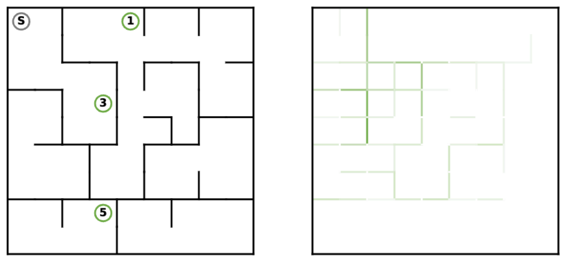


Figure 6: Left: maze side visible to *H*. Right: heatmap displaying hidden information inferred by *A-Comm*.

## Ethical Statement

The user study was approved by the University of Texas at Austin Institutional Review Boards under ID STUDY00005476.

## Acknowledgments

This work was supported in part by the National Science Foundation (NSF) under grants 1652113, 271643-874F, and 2211432. Additional support was provided by the Defense Advanced Research Projects Agency (DARPA) under Agreement No. HR00112490410.

## References

[Bard *et al.*, 2020] Nolan Bard, Jakob N Foerster, Sarath Chandar, Neil Burch, Marc Lanctot, H Francis Song, Emilio Parisotto, Vincent Dumoulin, Subhodeep Moitra, Edward Hughes, et al. The Hanabi challenge: A new frontier for AI research. *Artificial Intelligence*, 280:103216, 2020.

[Bernstein *et al.*, 2002] Daniel S Bernstein, Robert Givan, Neil Immerman, and Shlomo Zilberstein. The complexity of decentralized control of Markov decision processes. *Mathematics of Operations Research*, 27(4):819–840, 2002.

[Brown *et al.*, 2020] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901, 2020.

[Browne *et al.*, 2012] Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A survey of Monte Carlo tree search methods. *Transactions on Computational Intelligence and AI in Games*, 4(1):1–43, 2012.

[Campbell *et al.*, 2002] Murray Campbell, A Joseph Hoane Jr, and Feng-hsiung Hsu. Deep Blue. *Artificial intelligence*, 2002.

[Carroll *et al.*, 2019] Micah Carroll, Rohin Shah, Mark K Ho, Tom Griffiths, Sanjit Seshia, Pieter Abbeel, and Anca Dragan. On the utility of learning about humans for human-AI coordination. In *Advances in Neural Information Processing Systems*, 2019.

[Cowling *et al.*, 2012] Peter I Cowling, Edward J Powley, and Daniel Whitehouse. Information set Monte Carlo tree search. *Transactions on Computational Intelligence and AI in Games*, 4(2):120–143, 2012.

[Cowling *et al.*, 2015] Peter I Cowling, Daniel Whitehouse, and Edward J Powley. Emergent bluffing and inference with Monte Carlo tree search. In *Conference on Computational Intelligence and Games*, pages 114–121, 2015.

[Goil *et al.*, 2013] Aditya Goil, Matthew Derry, and Brenna D Argall. Using machine learning to blend human and robot controls for assisted wheelchair navigation. In *International Conference on Rehabilitation Robotics*, pages 1–6, 2013.

[Goodman, 2019] James Goodman. Re-determinizing information set Monte Carlo tree search in Hanabi. *ArXiv preprint arXiv:1902.06075*, 2019.

[He *et al.*, 2017] He He, Anusha Balakrishnan, Mihail Eric, and Percy Liang. Learning symmetric collaborative dialogue agents with dynamic knowledge graph embeddings. In *Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017.

[He *et al.*, 2018] He He, Derek Chen, Anusha Balakrishnan, and Percy Liang. Decoupling strategy and generation in negotiation dialogues. In *Conference on Empirical Methods in Natural Language Processing*, 2018.

[Jacob *et al.*, 2022] Athul Paul Jacob, David J Wu, Gabriele Farina, Adam Lerer, Hengyuan Hu, Anton Bakhtin, Jacob Andreas, and Noam Brown. Modeling strong and human-like gameplay with KL-regularized search. In *International Conference on Machine Learning*, pages 9695–9728, 2022.

[Kocsis and Szepesvári, 2006] Levente Kocsis and Csaba Szepesvári. Bandit based Monte-Carlo planning. In *European Conference on Machine Learning*, pages 282–293, 2006.

[Lewis *et al.*, 2017] Mike Lewis, Denis Yarats, Yann Dauphin, Devi Parikh, and Dhruv Batra. Deal or no deal? End-to-end learning of negotiation dialogues. In *Conference on Empirical Methods in Natural Language Processing*, 2017.

[Palan and Schitter, 2018] Stefan Palan and Christian Schitter. Prolific. ac—A subject pool for online experiments. *Journal of Behavioral and Experimental Finance*, 17:22–27, 2018.

[Paquette *et al.*, 2019] Philip Paquette, Yuchen Lu, Seton Steven Bocco, Max Smith, Satya O-G, Jonathan K Kummerfeld, Joelle Pineau, Satinder Singh, and Aaron C Courville. No-press diplomacy: Modeling multi-agent gameplay. In *Advances in Neural Information Processing Systems*, 2019.

[Peaceable Kingdom, 2016] Peaceable Kingdom. Gnomes at night. https://www.walmart.com/ip/Peaceable-Kingdom-Gnomes-at-Night-Game-2-to-4-Players-Ages-6/585042377, 2016. Accessed: 2024-01-15.

[Silver *et al.*, 2016] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

[Silver *et al.*, 2017] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of Go without human knowledge. *Nature*, 550(7676):354–359, 2017.

[Silver *et al.*, 2018] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science*, 362(6419):1140–1144, 2018.

[Szita *et al.*, 2010] István Szita, Guillaume Chaslot, and Pieter Spronck. Monte-Carlo tree search in Settlers of Catan. In *Advances in Computer Games*, pages 21–32, 2010.