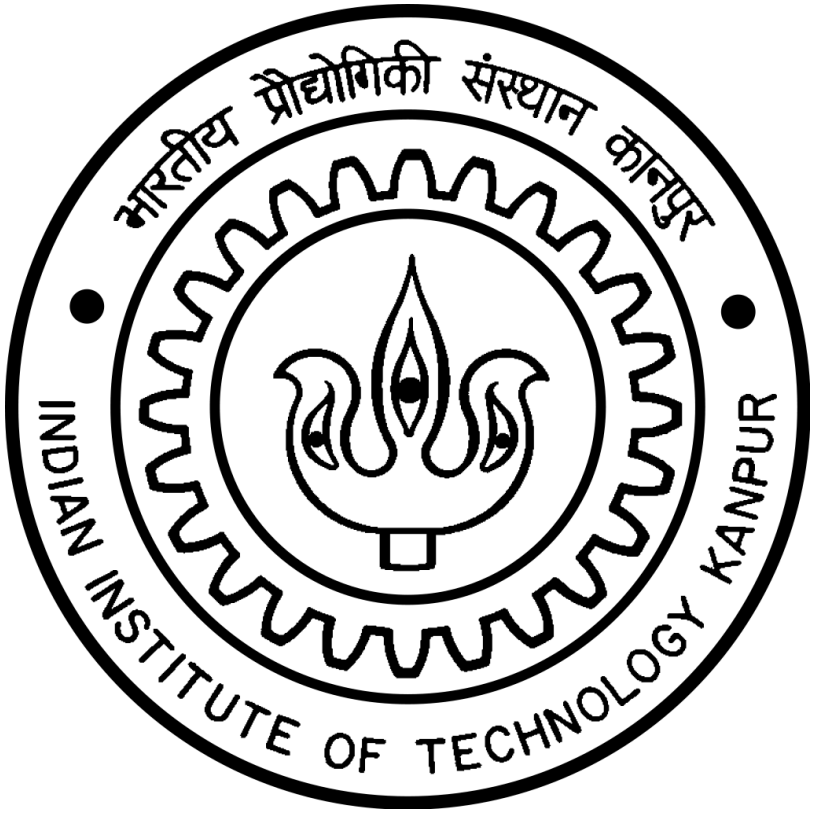
Indian Institute of Technology, Kanpur

CGS786 Midsem Report

Bayesian inference for coordinating multi-agent collaboration



Contents

Chapter 1

INTRODUCTION

Collaboration among agents is a cornerstone of achieving collective goals that surpass individual capabilities. However, effective collaboration necessitates seamless coordination, which poses significant challenges. One key aspect is the ability to understand and predict the intentions of others, a cognitive faculty known as theory-of-mind (ToM). In this report, we aim to explore the role of ToM in facilitating efficient collaboration among artificial agents.

We will delve into the fundamental challenge of imbuing artificial agents with ToM capabilities, enabling them to anticipate and adapt to the actions and intentions of their peers. By understanding how humans leverage ToM to cooperate in novel situations, we aim to develop strategies for integrating similar abilities into artificial agents. Through theoretical analysis and computational modeling, we will explore approaches for inferring intentions from observed actions and navigating the uncertainties inherent in collaborative environments.

1.1 Problem Statement

Effective collaboration among agents is essential for achieving collective goals that surpass individual capabilities. However, coordinating actions in collaborative tasks poses significant challenges. These challenges include maximizing efficiency through task division, fostering cooperation when tasks require joint effort, and ensuring spatial and temporal coordination to prevent interference and promote mutual responsiveness. Addressing these challenges is crucial for enhancing the performance of multi-agent systems in diverse collaborative environments.

1.2 Objectives

For the above problem statement, our main focus will be on a specific cooking problem: making a salad. To illustrate, imagine the process required: chopping tomatoes and lettuce, then assembling them on a plate. In this collaborative task, agents face challenges such as task division, cooperation, and spatial/temporal coordination.

Our approach will involve leveraging Bayesian inference techniques to enhance collaboration among agents. Specifically, we will:

1. Utilize Bayesian inference to optimize task allocation, ensuring efficient sub-task division.
2. Implement Bayesian-based coordination mechanisms to improve spatial and temporal coordination, minimizing interference and promoting mutual responsiveness among agents.

By harnessing Bayesian inference, we aim to develop effective strategies that enable seamless collaboration among agents in the salad-making process.

Chapter 2

Background

2.1 Multi-Agent MDPs with Sub-Tasks

Multi-Agent Markov Decision Processes (MMDPs) with sub-tasks introduce a structured approach to decision-making in multi-agent environments. This framework, represented as a tuple $\langle n, S, A_1...n, T, R, \gamma, \mathcal{T} \rangle$, encompasses:

- n: The number of agents involved.
- S: Object-oriented states describing the environment's configuration.
- $A_1...n$: Joint action space, with each agent possessing its action set.
- $T(s, a_1...n, s')$: Transition function indicating state transitions after actions.
- $R(s, a_1...n)$: Shared reward function across agents.
- γ: Discount factor for future rewards.
- $\mathcal{T}$: Partially ordered set of sub-tasks, structuring the environment and posing coordination challenges.

These sub-tasks, typically representing steps in a recipe, are denoted as $\mathcal{T} = \{\mathcal{T}_0, ..., \mathcal{T}_{|T|}\}$ and come with preconditions and postconditions. Each agent strives to devise a policy $\pi_i(s)$ that maximizes the expected discounted reward. While the environment's state is fully observable to all agents, they lack visibility into each other's policies ($\pi_{-i}(s)$) or internal representations. These sub-tasks, typically expressed as *Merge(X, Y)* for our work, guide high-level coordination, especially in cooking environments. Unlike Static Partially Observable Team Achievement Problems (SPATAPs), where task locations are fixed, here, they may vary, adding complexity to coordination.

2.1.1 Cooordination Challenges

The introduction of partial orderings in sub-tasks brings forth two key coordination challenges. Firstly, the Merge sub-task lacks specifications regarding efficient action implementation or which agent(s) should undertake it. Secondly, due to the partial ordering, sub-tasks can be accomplished in various sequences. For instance, in the Salad recipe, once the tomato and lettuce are chopped, multiple sequences can be followed for combining and plating them. This variability adds complexity to coordination efforts, as agents must align their sub-task orderings for successful collaboration.

Environment Details

The evaluation framework for assessing multi-agent collaboration draws inspiration from the popular game Overcooked, which simulates cooperative kitchen environments. The primary objective across all scenarios is to efficiently prepare recipes within a limited timeframe. Each session concludes either upon successful delivery of salad to the designated area or after a fixed time threshold of 100 steps is reached.

The kitchen environments are depicted as 2D grid-worlds, comprising movable food items and plates situated on counters, alongside fixed stations like knife stations. A comprehensive representation of the state encompasses entities, their types, positions, and statuses(Table 2.1).

Table 2.1: Object state representation

Type	**Location**	**Status**
Agent	$\{x, y\}$	[]
Plate	$\{x, y\}$	[]
Counter	$\{x, y\}$	[]
Delivery	$\{x, y\}$	[]
Knife	$\{x, y\}$	N/A
Tomato	$\{x, y\}$	{chopped, unchopped}
Lettuce	$\{x, y\}$	{chopped, unchopped}

Agents, portrayed as chef characters, navigate the kitchen by moving orthogonally or remaining stationary. They cannot traverse through obstacles or occupy the same space simultaneously; any such attempt results in stasis. Object manipulation involves picking up items upon contact and depositing them within counters. Food preparation involves transporting items to knife stations for chopping, while the merging of food with plates is feasible. Notably, agents are restricted to carrying a single item at any given time and are incapable of direct item exchange.

The classification of object states and interaction dynamics highlights how things work together in the kitchen. When you chop unchopped food with a knife, it becomes chopped. Also, different items can combine together, like mixing ingredients. This all happens in a way that affects how objects move and interact with each other.

Interaction dynamics

- Food.unchopped + Knife → Food.chopped + Knife
- Food1 + Food2 → [Food1, Food2]
- X + Y[] → Y[X]

2.2 Bounded Real-Time Dynamic Programming(BRTDP) Algorithm

Bounded Real-Time Dynamic Programming (BRTDP) is an approximate dynamic programming algorithm used to solve large-scale Markov Decision Processes (MDPs). It's designed to address the computational challenges posed by MDPs with large state spaces by limiting the depth of lookahead during value updates.

2.2.1 Iterative Value Updates:

It iteratively updates the value function $V(s)$ for each state s using the Bellman equation:

$$V(s) \leftarrow \min_a \left(R(s,a) + \sum_{s'} T(s'|s,a)V(s') \right)$$

where:

- $R(s,a)$ is the immediate reward of taking action a in state s.
- $T(s'|s,a)$ is the transition probability from state s to s' under action a.
- $V(s')$ is the value of the next state s'.

For our case,

$$Q^b_{\mathcal{T}_i}(s,a) = R_{\mathcal{T}_i}(s,a) + \sum_{s' \in S} T(s'|s,a)V^b_{\mathcal{T}_i}(s')$$
$$V^b_{\mathcal{T}_i}(g) = 0, V^b_{\mathcal{T}_i}(s) = \min_{a \in A_i} Q^b_{\mathcal{T}_i}(s,a)$$

where,

- $V^b_{\mathcal{T}_i}(g) = 0$, indicating that the value of the goal state g under $V^b_{\mathcal{T}_i}$ is 0.
- $Q^b_{\mathcal{T}_i}(s,a)$ is the expected future reward of a towards the completion of sub-task $\mathcal{T}_i$.
- $V^b_{\mathcal{T}_i}(s)$ is the estimated value of the current state under sub-task $\mathcal{T}_i$.
- b = [l,u], where l and u are the lower and upper bound of the reward.

Each time step is penalized by 1 and movement (as opposed to staying still) by an additional 0.1. The lower-bound was initialized to the **Manhattan distance between objects (which ignores barriers)**. The upper-bound was the **sum of the shortest-paths between objects** which ignores the possibility of more efficiently passing objects. For more details on BRTDP, refer to `https://dl.acm.org/doi/pdf/10.1145/1102351.1102423`.

Chapter 3

Bayesian Delegation Model

Bayesian Delegation is an innovative algorithm for coordinating multiple agents, utilizing inverse planning to make probabilistic inferences about the tasks other agents are performing. The core idea is to model the hidden intentions of other agents to dynamically decide whether to divide tasks or cooperate on them. This approach enables agents to select the appropriate task when multiple options are available, without direct communication.

3.1 Representation of Task allocations

In the Bayesian Delegation framework, task allocations are represented through a probability distribution denoted by $P(ta)$ where $ta \in \mathbf{t_a}$. Here, $\mathbf{t_a}$ encompasses all conceivable assignments of agents to specific sub-tasks. For instance, if there are two tasks labeled $\mathcal{T}_1$ and $\mathcal{T}_2$, and two agents denoted as i and j, then $\mathbf{t_a}$ will be a set consisting of below 4 elements:

- $(i : \mathcal{T}_1, j : \mathcal{T}_2)$: This allocation means that agent i is assigned to task $\mathcal{T}_1$ and agent j is assigned to task $\mathcal{T}_2$.
- $(i : \mathcal{T}_2, j : \mathcal{T}_1)$: In this allocation, agent i is assigned to task $\mathcal{T}_2$ while agent j is assigned to task $\mathcal{T}_1$.
- $(i : \mathcal{T}_1, j : \mathcal{T}_1)$: Here, both agents i and j are assigned to the same task $\mathcal{T}_1$.
- $(i : \mathcal{T}_2, j : \mathcal{T}_2)$: Similarly, both agents i and j are assigned to task $\mathcal{T}_2$ in this allocation.

Hence, we can see that $\mathbf{t_a}$ include both possibilities that agents will cooperate with each other(i.e. work on shared sub-task) or will divide and conquer(i.e. work on separate sub-tasks).
Agents continually update their beliefs regarding the most probable task allocation based on a shared history of observations. This probabilistic representation enables agents to dynamically adapt their coordination strategies based on the inferred task allocations, facilitating effective multi-agent collaboration without the need for direct communication.

3.2 Posterior inference

At each time step, agents employ Bayesian inference to select the most likely task allocation, denoted as ta^*, based on the posterior probability distribution $P(ta|H_{0:T})$, i.e., $ta^* = \arg\max_{ta} P(ta|H_{0:T})$, where $H_{0:T}$ represents the observed history of states and actions up to time step T. It is a set containing pairs of state-action tuples from time step 0 to time step T, represented as $H_{0:T} = \{(s_0, a_0), (s_1, a_1), ..., (s_T, a_T)\}$. Each tuple (s_t, a_t) consists of the state s_t observed at time step t and the corresponding action a_t taken by the agents at that time step. This posterior is calculated using Bayes' rule:

$$P(ta|H_{0:T}) \propto P(ta) \times \prod_{t=0}^{T} P(\mathbf{a_t}|s_t, ta)$$

Here, $P(ta)$ is the prior probability of task allocations, and $P(\mathbf{a_t}|s_t, ta)$ is the likelihood of observing actions given the current state and task allocation.

3.3 Likelihood Calculation

The likelihood $P(a_t|s_t, ta)$ is computed based on the expected future reward for each agent towards completing their assigned sub-task. It is represented as:

$$P(\mathbf{a_t}|s_t, ta) \propto \prod_{i:\mathcal{T}\in ta} \exp(\beta \cdot Q^*_{\mathcal{T}_i}(s, a_i))$$

Here, $Q^*_{\mathcal{T}_i}(s, a_i)$ represents the expected future reward for agent i in completing sub-task T_i, and β controls the belief about the optimality of other agents' actions.

3.4 Prior Calculation

The prior probability $P(ta)$ is computed from the environment state, giving higher weight to task allocations that can be completed more efficiently. First, $P(ta) = 0$ for all ta that have sub-tasks without satisfied preconditions, else it is calculated as:

$$P(ta) \propto \sum_{\mathcal{T}\in ta} \frac{1}{V_{\mathcal{T}}(s)}$$

Where $V_T(s)$ is the estimated value of the current state under sub-task $\mathcal{T}$. These Priors are reinitialized when new sub-tasks have their preconditions satisfied and when others are completed.

3.5 Action Planning

Action planning, integral to Bayesian Delegation, involves selecting optimal actions based on the inferred task allocation. Bounded real-time dynamic programming (BRTDP) is employed to find approximately optimal action policies and Q-values for each sub-task [2.2]. This planning process considers the movements of other agents and ensures efficient coordination.

3.6 Coordination Strategies

After action planning, agents utilize the inferred task allocation to address two primary types of coordination challenges:

3.6.1 Avoiding Interference

- **Level-0 Models:** Agents create models of other agents assuming they remain stationary while working on their respective sub-tasks. These models, referred to as "level-0 models," allow agents to anticipate the actions of others without the need for real-time communication. By assuming other agents' actions are fixed, agents can simplify the planning process by treating the environment as if they were the only active agent.

- **Single-Agent Transition Function:** The multi-agent transition function is transformed into a single-agent transition function, where the actions of other agents are integrated into the environment dynamics. This allows each agent to plan its actions independently while considering the anticipated actions of others based on the level-0 models.

- **Planning with Level-0 Models:** Using the transformed single-agent transition function, agents employ planning algorithms such as Bounded real-time dynamic programming (BRTDP) to find approximately optimal action policies. These policies enable agents to execute actions that minimize interference with others while maximizing progress towards their individual sub-tasks.

3.6.2 Efficient Coordination

- **Fictitious Centralized Planner:** When agents collaborate on the same sub-task, they simulate a centralized planner that coordinates the actions of all agents involved. This fictitious planner enables agents to jointly plan actions that lead to optimal outcomes for the shared task.

- **Transformed Action Space:** To facilitate joint planning, the action space is transformed to include actions for all agents working on the shared sub-task. Each agent selects actions based on the joint policies determined by the centralized planner, ensuring alignment and cooperation towards achieving the shared objective.

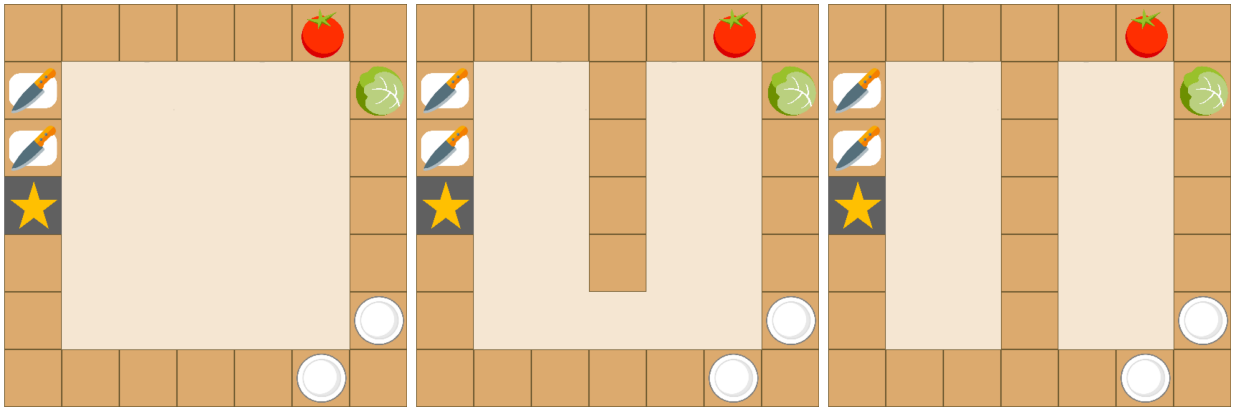
- **Emergent Decentralized Cooperation:** Despite planning jointly, agents maintain their autonomy, allowing for emergent decentralized cooperation. This flexibility enables agents to discover efficient and novel strategies for solving shared sub-tasks collaboratively, such as passing objects between each other to accomplish goals efficiently.

- **Stochasticity in Joint Planning:** Due to the stochastic nature of planning processes and the independence of agents' planning, the resulting joint policies may not always be perfectly synchronized. However, agents adapt to these uncertainties and adjust their actions dynamically based on observed outcomes and ongoing coordination efforts.

Chapter 4

Results

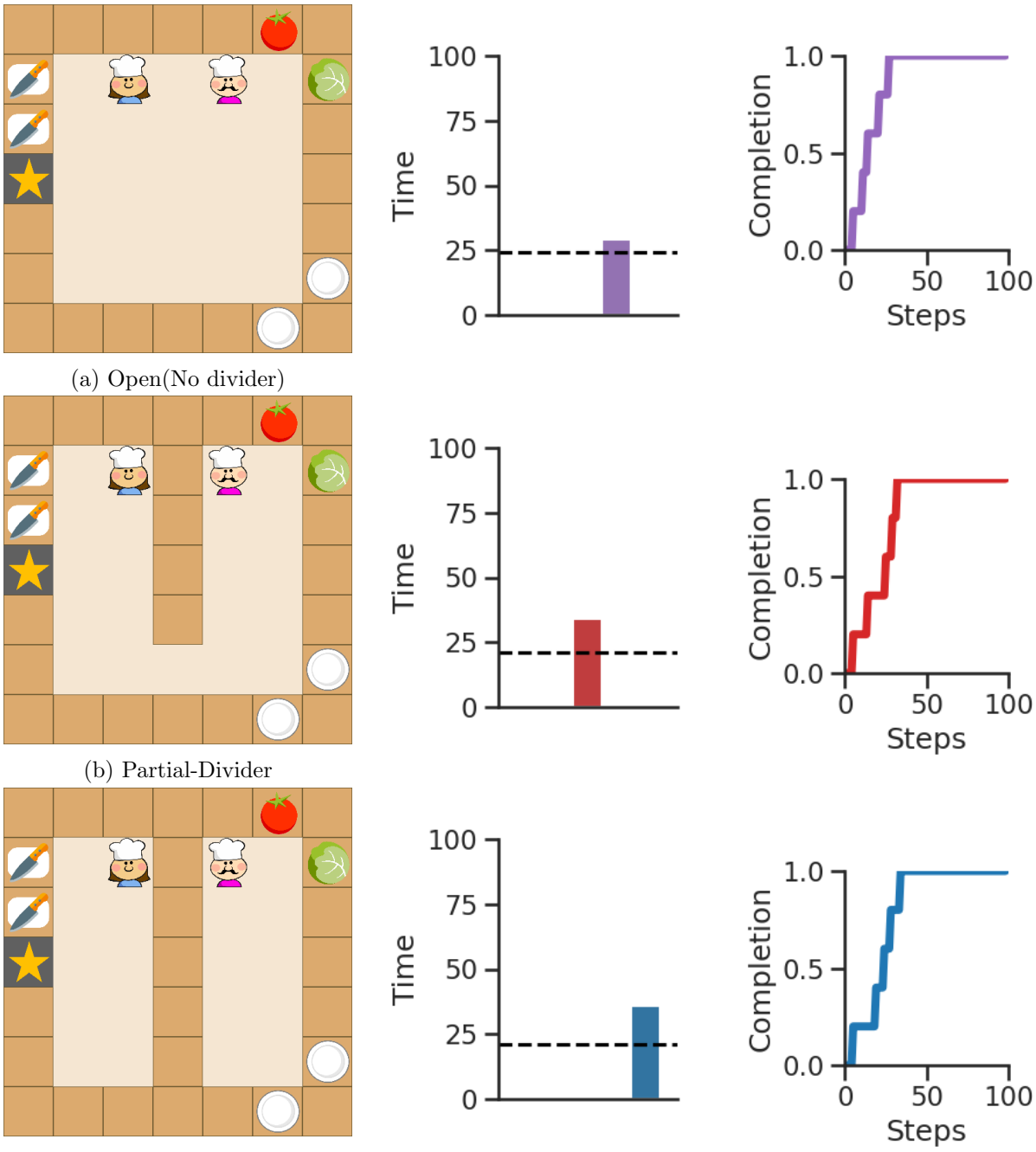
We have evaluated the model in Three environment conditions (in our case 3 types of kitchens) for making a salad. Below are the environment conditions:



(a) Open(No divider) (b) Partial-Divider (c) Full-Divider

Figure 4.1: Environment Conditions

Our model was trained with both two agents and three agents for each environmental condition, aimed at evaluating its performance across various scenarios. The softmax during inference used $\beta = 1.3$, which was optimized to data from the behavioral experiment. A crucial aspect of this evaluation was assessing the model's adaptability to different conditions, particularly in scenarios where multiple agents are involved, as this reflects real-world complexities.
Understanding the behaviors of other agents is paramount, especially in intricate tasks where actions can be executed in multiple sequences. In our experiment, the preparation of a Salad recipe presented such complexity, with three possible orderings: (a) combining chopped tomato and lettuce before plating, (b) plating lettuce first followed by tomato addition, or (c) plating tomato first followed by lettuce addition. Compounding this challenge, no single agent possessed the capability to independently execute sub-tasks. Our findings indicate that despite the inherent complexities, our model demonstrated robust performance, effectively navigating the challenges posed by the Salad recipe. Notably, it successfully completed all sub-tasks within the designated time constraints. Figure 4.2 visually illustrates the model's proficiency in tackling these challenges, showcasing its ability to coordinate actions and converge on efficient solutions.
These results underscore the efficacy of our model in handling multi-agent environments and addressing intricate task structures. By effectively learning to coordinate actions and adapt to diverse conditions, our model exhibits promising capabilities for real-world applications where collaboration among agents is essential for task accomplishment.



(a) Open(No divider)

(b) Partial-Divider

(c) Full-Divider

Figure 4.2: Performance results for each kitchen-recipe composition for Two agents. The row shows kitchen and the columns shows graphs related to each environment. Within each row, first graph shows the time steps taken to complete all tasks and the second graph shows the fraction of sub-tasks completed over time.

Chapter 5

Discussion

We've developed Bayesian Delegation, an algorithm inspired by human theory-of-mind. It facilitates efficient ad-hoc coordination by rapidly inferring others' sub-tasks, enabling agents to decide when to collaborate and when to divide tasks for greater efficiency. This approach also allows agents to complete sub-tasks that neither could achieve alone, reflecting key aspects of human cooperation. Similar to humans, Bayesian Delegation effectively predicts task allocations with minimal data, mirroring human-like decision-making processes. However, a notable challenge emerges when agents engage in joint planning for a single sub-task: the absence of a mechanism to signal completion of individual contributions to the collective effort. Addressing this challenge stands as a critical future endeavor, one that will enhance the algorithm's efficacy and align it more closely with human-like collaboration dynamics.

Moreover, our current model training process demands substantial time and memory resources. For instance, training a model with two agents necessitates approximately 10 GB of memory and around 2 hours to complete all sub-tasks across various environments. Scaling up to three agents exacerbates resource requirements, with the risk of system crashes during training due to memory overload.

Moving forward, our focus will encompass devising strategies to address the aforementioned challenges. Specifically, we aim to implement mechanisms for signaling individual task completion within joint efforts, enhancing the algorithm's adaptability and efficiency. Additionally, we will explore avenues to optimize resource utilization during model training, seeking to reduce both time and memory requirements without compromising performance.