View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Harvard University - DASH

Modeling Information Exchange Opportunities for Effective Human-Computer Teamwork

The Harvard community has made this article openly available. Please share how this access benefits you. Your story matters.

Citation	Kamar, Ece, Ya'akov Gal, and Barbara J. Grosz. 2013. Modeling information exchange opportunities for effective human-computer teamwork. Artificial Intelligence 195:528–550.
Published Version	doi:10.1016/j.artint.2012.11.007
Accessed	February 19, 2015 12:03:00 PM EST
Citable Link	http://nrs.harvard.edu/urn-3:HUL.InstRepos:10652553
Terms of Use	This article was downloaded from Harvard University's DASH repository, and is made available under the terms and conditions applicable to Open Access Policy Articles, as set forth at http://nrs.harvard.edu/urn-3:HUL.InstRepos:dash.current.terms-of-use#OAP

(Article begins on next page)

Modeling Information Exchange Opportunities For Effective Human-Computer Teamwork

Ece Kamar[1], Ya'akov (Kobi) Gal[2,3], and Barbara J. Grosz[3]

[1]Microsoft Research, USA
[2]Dept. of Information Systems Engineering, Ben-Gurion University, Israel
[3]School of Engineering and Applied Sciences, Harvard University, USA

Abstract

This paper studies information exchange in collaborative group activities involving mixed networks of people and computer agents. It introduces the concept of "nearly decomposable" decision-making problems to address the complexity of information exchange decisions in such multi-agent settings. This class of decision-making problems arise in settings which have an action structure that requires agents to reason about only a subset of their partners' actions - but otherwise allows them to act independently. The paper presents a formal model of nearly decomposable decision making problems, NED-MDPs, and defines an approximation algorithm, NED-DECOP that computes efficient information exchange strategies. The paper shows that NED-DECOP is more efficient than prior collaborative planning algorithms for this class of problem. It presents an empirical study of the information exchange decisions made by the algorithm that investigates the extent to which people accept interruption requests from a computer agent. The context for the study is a game in which the agent can ask people for information that may benefit its individual performance and thus the group's collaboration. This study revealed the key factors affecting people's perception of the benefit of interruptions in this setting. The paper also describes the use of machine learning to predict the situations in which people deviate from the strategies generated by the algorithm, using a combination of domain features and features informed by the algorithm. The methodology followed in this work could form the basis for designing agents that effectively exchange information in collaborations with people.

1 Introduction

Settings in which computer systems and people collaborate are becoming increasingly prevalent, arising in a wide variety of application domains (e.g, hospital care-delivery systems, systems administration applications) as well as in virtual reality and simulation systems (e.g, disaster relief, military training). A common characteristic of such collaborations is that task-related information is distributed among the group members. As a result, an agent may come to know something useful for another participant that is not directly available to that participant, and the collaboration may benefit from this information being communicated at an appropriate time.

This paper focuses on determining when information exchange is beneficial in the particular case of a computer collaborating with a person when their shared endeavor involves tasks for most of which they may operate

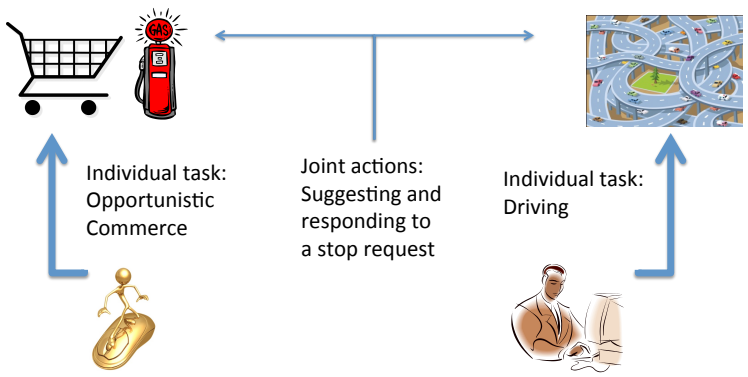

Figure 1: Mobile commerce as a nearly decomposable problem.

semi-independently. In semi-independent situations, the individual actions and joint actions that collaborative plans comprise can be reasoned about separately. The paper proposes an approach that takes advantage of this "nearly decomposable" structure of activities in semi-independent settings. To overcome the complexity barrier of collaborative decision-making in general settings, which has been shown to be intractable [Bernstein et al., 2002], it defines nearly decomposable processes and a new formal model of effective information exchange for nearly decomposable decision-making problems. It uses this model to define an approximate algorithm for computing information exchange strategies for nearly-decomposable problems. Throughout the paper, we use interruption management, a special case of information exchange, to illustrate the use of the model and operation of the algorithm. The model and algorithm do, however, handle information exchange and nearly decomposable tasks more broadly.

Nearly decomposable task structures, as well as important challenges of interruption management, may be illustrated by the example of an automated agent for mobile commerce that works with a person to plan shopping stops on the route home from work. The agent generates suggested route changes to meet the driver's shopping needs opportunistically, while keeping in mind the cost of the suggested route changes. The driver and agent share the goal of the person of getting to his or her destination effectively, but each of them performs an individual task: the person drives the car, and the agent generates an optimal route. For significant portions of the time, each operates alone, but at key points the agent may interrupt the person to suggest a stop [Kamar et al., 2008]. The agent might also benefit from interrupting the person to get information about road conditions or shopping preferences. To succeed, the agent needs to decide wisely whether and when to interrupt the driver for such information. A navigation system that continuously interrupts may distract drivers' attention from the road or irritate them to the point where they ignore or turn off the system. Figure 1 illustrates this situation schematically. To represent the fact that participants are mostly performing their own tasks, and they interact only when needing to exchange information, the arcs representing individual actions are thicker than the arcs representing joint actions.

Information exchange actions, and interruptions in particular, usually generate costs, including communication and cognitive costs. For computer agents to avoid overburdening their human partners, they must manage such interactions appropriately. This paper combines theoretical model development, machine learning and empirical studies of several factors that affect people's perceptions of, and responses to, agents' information exchange actions. This research followed an incremental design sequence: (1) constructing an initial, formal

model of information exchange, (2) developing an algorithm for computing information exchange strategies efficiently based on this model; (3) conducting empirical studies to determine when people's willingness to accept information exchange requests differs from the strategies computed by the algorithm; (4) using machine learning to predict the conditions under which people deviate from the information exchange decisions made by the algorithm.

We formalize the planning problem for nearly decomposable tasks as a *NEarly Decomposable Markov Decision-Making Process* (NED-MDP). An NED-MDP distinguishes between individual actions relating to a participant's individual task and joint actions that relate to coordinating and sharing information between participants. It explicitly represents the division of nearly decomposable tasks into these two types of actions, embedding this nearly decomposable structure in the transition and reward functions used by agents in the NED-MDP.

This paper defines NED-MDPs formally and presents a novel algorithm, NED-DECOP, that exploits their structure to approximate the optimal strategy for sharing information and coordinating in the collaborative activity. The NED-DECOP algorithm explicitly reasons about the effects of agents' coordination strategies on a group's joint performance in the future. Importantly, it is not myopic. Furthermore, as demonstrated in the paper, depending on the way participants' individual tasks are solved, the complexity of the algorithm may be no worse than the complexity of the algorithm used to solve participants' individual tasks.

We designed an agent that computed the value of interruption opportunities using the nearly decomposable structure of these problem situations. The agent's decisions were empirically evaluated using a collaborative game in which a person and a computer agent have separate tasks, but their success is a joint measure combining their performances. In performing their separate tasks, the agent and the person can help each other by exchanging information; doing so has a cost. Throughout the game, the person has information of value to the agent's task, which the agent may benefit from knowing. To obtain this information, the agent must interrupt the person. Because the human player has complete information about the game states, she/he can, in principle, calculate the actual value of any interruption and accept only those that are truly beneficial to both participants. A wide range of prior research suggests, however, that people might deviate from this "purely rational" approach [Bazerman, 2001, Camerer, 2003, Falk and Fischbacher, 2006]. As a result, an intended to be beneficial interaction for collaboration may turn into a performance degrading disturbance.

To understand how people perceive the benefit of an interruption, we collected over 700 game instances that varied in the extent to which providing information to the agent benefited the collaborative activity, the distribution of benefit between players, and the moment in the game at which the interruption occurred. We also varied the perceived partner type: although in all cases, subjects interacted with a computer agent, some subjects were told their partner was another person. We measured whether people accepted interruption requests in different conditions, examining in particular the influence of four factors on people's decisions to accept or reject these requests: (1) the actual benefit of an interruption to the collaboration, (2) the actual benefit of an interruption to the person (to the person's individual task performance), (3) the actual benefit of an interruption to the agent (to the agent's task performance), and (4) the perceived type of the agent (whether computer or person). We measured whether (2), (3) and (4) affected people's acceptance rates, even though only the combined benefit (1) really mattered to the players' score in the game.

The results show, unsurprisingly, that when the benefit provided by responding to the interruption was substantial (either clearly positive or negative), people's behavior aligned with that predicted by the NED-

DECOP algorithm. However, when the benefit was relatively small, the two diverged. Most interestingly, when the benefit was equivocal, people's decisions to accept depended on whether they perceived the interruption to come from a person or from a computer agent. They tended to accept interruptions from other people more readily than from computers in these situations. The decision also depended on people's perception of the benefit of the interruption to individual participants. For interruptions with small benefits or losses, people tend to over estimate the individual benefit to themselves, and undervalue the benefit of interruptions to their partners, even though the game score and overall reward to the person depended only on the sum of these scores.

We analyzed the disparity between people's behavior and that predicted by the NED-DECOP algorithm using two different standard machine learning algorithms to identify the influence of the four factors listed above on people's behavior. We compared the performance of both personalized and general learning methods in predicting people's decisions. The best results, which were obtained with a model that combined these two approaches, achieved almost 90% accuracy.

The key contributions of this paper are the definition of nearly-decomposable decision-making problems, the design of the NED-DECOP algorithm for computing agents' coordination strategies for such problems, and a new methodology for designing interruption-management policies in collaborative group activities involving both people and computer agents. The methodology uses the NED-DECOP algorithm to establish a fully rational baseline of behavior. This baseline is then used to evaluate empirically how people perceive requests for interruptions in diverse types of situations, determining when people deviate from the optimal policy. Finally, machine learning is used to build learned models that can predict the probability that people will accept an interruption, using a combination of domain features and features informed by the algorithm. This methodology can form the basis for the iterated design of agents that effectively exchange information when collaborating with people.

This paper is organized as follows. The next section describes prior work on decentralized multi-agent collaboration strategies and on interruption management. Section 3 describes the game we used to investigate the interruption management problem. Section 4 shows how NED-MDPs can be used to approximate the optimal strategies for a computer agent in this setting, and how these strategies can be used in practice to make a decision whether to interrupt people in the domain. Section 5 provides a detailed empirical evaluation of this approach in the laboratory using human subjects. Section 6 uses machine learning techniques to measure the extent to which people's decision-making differs from the rational models. Section 7 discusses how to adapt our methodology to other domains and to integrate the learned model into agent design. Section 8 concludes and describes some important challenges for future work. This paper significantly extends earlier work [Kamar et al., 2009] to formally define NED-MDP, include a general algorithm for solving NED-MDPs, and show how the model can be improved using machine learning.

2 Related Work

The work reported in this paper relates to two very different areas of prior work and a range of approaches within each: decision-theoretic multi-agent planning and interruption management. The subsections below discuss related work in these two areas respectively.

2.1 Decision Theoretic Multi-Agent Planning

Various approaches have modeled collaborative decision-making and planning in environments in which there are uncertainty and costs, including Multi-agent Markov Decision Process (MMDP) [Boutilier, 1999a], Communicative Multiagent Team Decision Problem (COM-MTDP) [Pynadath and Tambe, 2002a], and Decentralized Markov Decision Processes (Dec-MDP and Dec-POMDP) [Bernstein et al., 2002]. These models are shown to be equivalent in terms the problems they represent and the computational complexity for solving them [Seuken and Zilberstein, 2008].

Since the introduction of Decentralized MDPs as a formal model for decentralized multi-agent planning, a number of algorithms have been proposed for computing exact or approximate solutions. Noteworthy examples of approaches include dynamic programming [Oliehoek et al., 2008a, Hansen et al., 2004, Seuken and Zilberstein, 2007], policy search with admissible heuristics [Szer et al., 2005] and policy search with pruning [Amato et al., 2007]. The complexity of solving Dec-MDP models has been proven to be NEXP-hard. In particular Bernstein et al. [2002] have shown that solving decentralized MDPs can be doubly exponential for the two agent case. Finding ϵ-approximate policies for Dec-MDPs is also NEXP-hard [Rabinovich et al., 2003]. Thus, general techniques for DEC-MDPs are not a feasible approach for the types of problems this paper addresses. We focus this review on approaches that exploit the structure of DEC-MDPs to facilitate decision-making, and refer the reader to the papers by Oliehoek [2012] and Seuken and Zilberstein [2008] for a general account on Decentralized MDPs.

We first discuss algorithms that exploit the structure which is embedded in particular classes of Dec-MDPs. When the joint state is fully observable (or completely non-observable) by all agents, then solving decentralized planning problems has been shown to be more tractable than general Dec-MDPs [Boutilier, 1999b, Pynadath and Tambe, 2002b]. In particular, Transition-independent Decentralized MDPs model problems in which agents operate completely independently, but are connected through a reward structure that depends on their joint histories. Becker et al. [2003] present an optimal algorithm for solving Transition-independent Dec-MDPs, and Nair et al. [2005] propose a multi-agent planning algorithm for agents embedded in a sensor network that engenders transition independence. As a result, the computational overhead engendered by communication arises only in a subset of the state space. Other approaches focus on settings in which the joint state space can be factored and interactions among agents are local, thereby constrained by regions of the state space [Oliehoek et al., 2008b, Melo and Veloso, 2011]. In particular, Varakantham et al. [2009] partition the state space in domains where interactions between agents are limited to subclasses of the state space called "coordination locales". This approach uses a branch-and-bound approach for agents' task assignment and a special method for determining whether an agent should change its individual policy and interact with other agents because they are in the same locale. They consider interaction only in those places. Our own work also exploits structure but for different kinds of problems – those in which agents' tasks (not the state space) are partitioned into individual and joint actions. In contrast to the approaches above, we allow interaction to occur at any point in the state space.

Another approach that exploits the special structure of a subclass of DEC-MDPs for efficient planning is Event-driven Dec-MDPs (ED-DEC-MDPs). ED-DEC-MDPs factor the state space according to agents' individual tasks and the transition model is decoupled based on limited interactions between agents Becker et al. [2004]. Mostafa and Lesser [2009] generalized ED-DEC-MDPs to represent complex interactions in agents' reward functions. They assume that agents fully observe their local states and constrain interactions between

these local states to depend on previous events. Temporally Decoupled Dec-POMDPs decompose agents' interactions into individual models that are loosely decoupled [Witwicki and Durfee, 2010]. Although they do not factor the state space like ED-DEC-MDPs, they still require that agents' interactions depend on past events. In contrast, in a NED-MDP, agents may not fully observe their local states, and interactions may occur concurrently, as in the empirical setting we will describe in the following Section. More recent work by Melo et al. [2012] focuses on optimizing communication decisions in Dec-MDPs with sparse interactions. It assumes that agents' observations are independent of other agents' actions, that agents can query other agents' local states about their communication actions, and that communication actions do not affect the state transitions. None of these assumptions hold in a NED-MDP.

Other approaches to reducing the complexity overhead from information exchange consider models with zero or fixed communication costs. With zero costs, the complexity of the multi-agent decision-making problem reduces to the complexity of solving the single-agent problem, as agents can synchronize information at each time-step [Pynadath and Tambe, 2002a]. Roth et al. [2007] propose a multi-agent decision-making model that exploits this fact in team decision-making. In this model, policies for individual agents are computed off-line, while communication decisions are exogenous to the model and are made during run-time. Decisions about how and when to communicate are performed greedily [Roth et al., 2005, 2006]. Finally, some approaches combine both transition independence and constraints on communication. For instance, Goldman and Zilberstein [2008] present approximate algorithms for transition-independent Dec-MDPs that assume that agents can achieve the perfect view of the global state when they communicate and communication actions have a fixed cost.

While these approaches have proven effective in settings that meet the constraints for which they were designed, they do not, either individually or taken together, address the challenges that arise in human-computer information exchange, because their underlying assumptions do not hold. In particular, for most such mixed computer-human settings communication is not free, and communication costs necessarily depend on the agents' joint state space. In addition, the transition matrix over the state space depends on agents' joint actions.

Approximate solution algorithms have been proposed as an alternative approach to reducing complexity. A polynomial-time algorithm presented by Beynier and Mouaddib [2005] approximates the joint policy with individual agent policies that are connected by temporal and resource constraints. This algorithm assumes that individual agent models are fully observable and thus excludes communication decisions from the model. Xuan et al. [2001] propose an approximate algorithm that uses heuristic functions to compute the expected benefit of communication by comparing information gain with the cost. These heuristic methods are empirically evaluated; they do not guarantee performance bounds. As a special case of these heuristic methods, myopic approaches capture the expected benefit of interruption by assuming that communication can only happen at the present time and ignoring future opportunities [Becker et al., 2005]. These approaches diverge from the optimal if the setting allows repeated and frequent communication between agents.

2.2 Interruption Management

Our work relates to several threads of prior work on interruption management in the computational and cognitive sciences. McFarlane [2002] compared four strategies for deciding when to interrupt users: immediate, negotiated (user choosing the delivery time), mediated (an agent mediates the interaction with the user), and scheduled (delivering notifications within a predetermined schedule). His results show that none of the pro-

posed strategies is the single best approach, echoing the need for decision-making to choose among different interruption strategies.

Several works have suggested agent-designs for interruption management that are based on reasoning about the benefit to either the computer system or the user, but not both. One approach is to use a predictive model of people's reaction to notifications from computers to decide whether to postpone communication [Horvitz and Apacible, 2003, Horvitz et al., 1999, 2002, Fogarty et al., 2005, Hudson et al., 2003]. These approaches ignore the potential benefit of an interruption to the computer agent's task completion, which may affect overall collaboration and quality of the group endeavor. Approaches that focus solely on the needs of the system, do not reason about the effect of an interruption on users' behavior [Scerri et al., 2003]. Finally, Voida and Mynatt [2009] propose an agent design for interrupting users that assumes knowledge of the relevancy of each task the user is performing, but does not consider the cost or benefit of the interruption itself.

Work in human-computer interaction has shown that people often underestimate their own benefit from an interruption and perceive the party generating the interruption to gain more than themselves [O'Conaill and Frohlich, 1995, Kraut and Attewell, 1997]. The way a notification system is perceived at the initial stages of interaction is an important determinant for the success of further interactions [LeeTiernan et al., 2001]. The need to consider both sides of the interaction was echoed by [Rudman and Zajicek, 2006] who suggest that computer agents should initiate interactions that are perceived to be useful by their users. Our work is the first to provide a principled approach to interruption management as a collaborative activity that reasons about the effects of interruption to all participants.

Work in psychology has investigated the differences between people's self report of their willingness to be interrupted and other people's prediction about this willingness in face-to-face communication [Avrahami et al., 2007]. Other works have investigated the effects of interruptions on people's performance of cognitive tasks. These include the relevance of an interruption to the primary task and the characteristics of the primary task [Czerwinski et al., 2000], the length and memory requirements of an interruption [Gillie and Broadbent, 1989], the mental load of a user at the time of interruption [Iqbal and Bailey, 2006], and the timing and complexity of an interruption [Hodgetts and Jones, 2005].

The approach described in this paper also differs from proposals to deploy user-interface design to alleviate the costs of interruptions. It has been shown that providing people with a history of recent actions helps them to recover from interruptions [Renaud, 2000]. McCrickard et al. [2003] investigated the effect on task performance of saliency-conveying signals (e.g., color variation) in notifications as a way of enabling systems to that trade-off the utility of a notification with the cost of interruption. A similar line of research has been conducted to study multi-modal interfaces as a way of delivering interruptions of various utilities [Arroyo and Selker, 2003]. Awareness displays are designed to display information about the workload of a partner, and they have been shown to be useful for better managing interruptions between people [Dabbish and Kraut, 2003].

3 Approaches for Collaborative Decision-Making

This section models nearly-decomposable decision-making problems with traditional approaches to collaborative multi-agent decision-making. We then describe a new approach that makes explicit the interdependencies between the individual and joint actions of each agent. This approach can lead to exponential savings in computation time as compared to traditional DEC-MDP solution algorithms.

3.1 The DEC-MDP Approach

A Decentralized Markov Decision Process (Dec-MDP) is a formalism for multi-agent planning that captures the uncertain and partially observable nature of the real-world. A Dec-MDP includes a set of states with associated transition probabilities, a set of actions and observations for each agent, and a joint reward function [Bernstein et al., 2002]. A solution of a Dec-MDP is an optimal joint policy for all agents that is represented as a mapping from states to actions. In nearly decomposable problems, the joint state space is factored into the states representing each agent's task. Agents may have partial observability about their own tasks and about other agents' tasks. In the Dec-MDP formalization of these problems, the joint states encapsulates agents' local observations, and the transition function describes how these observations depend on each other. Formally, we model a 2-agent nearly decomposable decentralized planing problem as a Decentralized Markov Decision Problem (Dec-MDP) with a tuple $\langle \mathbf{I}, \mathbf{S}, \mathbf{A}_1, \mathbf{A}_2, T, R, H \rangle$ in which

- I is a set of agents.
- $\mathbf{S} = \mathbf{S_1} \times \mathbf{S_2}$ is a set of states. S_i indicates the set of states of agent i's task. S_i includes each agent's local observations about agent i's task. A DEC-MDP has a finite horizon H, and each state $s_i^h \in \mathbf{S}_i$ is associated with a time step $h \leq H$.
- $\mathbf{A}_1$ and $\mathbf{A}_2$ are the actions of agents 1 and 2, respectively.
- T is a transition function. $T\big(s^{h+1} \mid s^h, (a_1, a_2)\big)$ is the probability of transitioning from state $s^h = (s_1^h, s_2^h)$ to $s^{h+1} = (s_1^{h+1}, s_2^{h+1})$ given action pair (a_1, a_2).
- R is a global reward function. $R(s^h)$ is the joint reward function obtained by the agents at state s^h [1].
- A Dec-MDP also includes a finite set of observations and their associated likelihoods. The combined observations constitute the state space of the Dec-MDP. For simplicity, we do not represent these observations explicitly and assume that the observation function is captured by the transition function.

A policy for Agent 1, π_1, is a mapping from the local observations of Agent 1 about the joint state (i.e., observations about all agents' tasks) to an action for agent 1 (and similarly for Agent 2). We denote the action assigned by policy π_1 based on Agent 1's local observations as $\pi_1(s^h)$ as s^h includes Agent 1's local observations about the joint state. A joint policy $\pi = (\pi_1, \pi_2)$ is a pair consisting of a local policy for each agent. The expected value $V^\pi(s^h)$ for the joint policy is defined as

$$V^\pi(s^h) = V^{(\pi_1,\pi_2)}(s^h) = R(s^h) + \sum_{s^{h+1} \in \mathbf{S}} T(s^{h+1} \mid s^h, (\pi_1(s^h), \pi_2(s^h))) \cdot V^{(\pi_1,\pi_2)}(s^{h+1}) \tag{1}$$

An optimal joint policy $\pi^*(s^h) = (\pi_1^*(s^h), \pi_2^*(s^h))$ maximizes Equation 1. An optimal policy for a Dec-MDP determines when it is beneficial for agents to act individually or jointly. It initiates coordination among agents if doing so improves the collaborative utility. Although DEC-MDPs can capture decentralized decision making processes, the complexity of finding optimal solutions to Dec-MDPs is known to be NEXP-complete [Papadimitriou and Tsitsiklis, 1987], so that Dec-MDPs may not be a practical approach to solve such problems.

[1] It is also possible to make the reward function depend on the joint actions. The two forms of representation are equivalent.

3.2 The NED-MDP Approach

A collaborative activity typically includes both individual and joint actions. When participants of the collaboration are not interacting together, each participant only needs to reason about its individual tasks, and their activities are independent from each other. Participants need to reason about the collaborative activity only to determine when and how to coordinate, communicate with each other, support each other during potential breakdowns, and similar issues involving all of the participants. We model such collaborative activities as nearly-decomposable Markov Decision Processes.

A Nearly Decomposable Markov Decision Process (NED-MDP) is a Dec-MDP in which the state space $\mathbf{S}$ can be factored into individual state spaces $\mathbf{S}_1, \mathbf{S}_2$ for both agents that satisfy the following conditions:

1. there exists an individual transition function for Agent 1, $T_1\big(s_1^{h+1} \mid s_1^h, (a_1, a_2)\big)$ that represents the probability of reaching an individual state s_1^{h+1} given s_1^h and joint actions (a_1, a_2) (and similarly for Agent 2),

2. there exists an individual reward function for Agent 1, $R_1(s_1^h)$, mapping state s_1^h and action a_1 to a real number (and similarly for Agent 2),

3. the joint transition function T can be represented as the product of agents' individual transition functions,

$$T\big((s_1^{h+1}, s_2^{h+1}) \mid (s_1^h, s_2^h), (a_1, a_2)\big) = T_1\big(s_1^{h+1} \mid s_1^h, (a_1, a_2)\big) \cdot T_2\big(s_2^{h+1} \mid s_2^h, (a_1, a_2)\big) \tag{2}$$

4. the joint reward function R can be represented as the aggregate of the individual reward functions R_1 and R_2,

$$R(s_1^h, s_2^h) = R_1(s_1^h) + R_2(s_2^h) \tag{3}$$

5. the action set $\mathbf{A}_1$ for Agent 1 can be factored to a set of independent actions $\mathbf{A}_1^I$ and joint actions $\mathbf{A}_1^J$ (and similarly for Agent 2). Both agents coordinate to perform joint actions. When the agents are acting dependently ($a_1 \in \mathbf{A}_1^J$ and $a_2 \in \mathbf{A}_2^J$), the transition function for each agent depends on the joint action pair. When the agents are acting independently ($a_1 \in \mathbf{A}_1^I$ or $a_2 \in \mathbf{A}_2^I$), the transition function for each agent is independent of the other's action.

$$T_1(s_1^{h+1} \mid s_1^h, (a_1, a_2)) = T_1(s_1^{h+1} \mid s_1^h, a_1) \qquad \text{if } a_1 \in \mathbf{A}_1^I, a_2 \in \mathbf{A}_2^I \tag{4}$$

   (and similarly for Agent 2)

6. the set of joint actions for the group, $\mathbf{A}^J$, is the Cartesian product of $\mathbf{A}_1^J$ and $\mathbf{A}_2^J$, containing all pairs of joint actions agents can perform in coordination.

For example, in figure 1, the person's individual actions focus on driving and the agent's individual actions focus on searching for commerce opportunities. The joint actions include the agent's suggestion of a commerce opportunity to the person, and the person's response to this suggestion.

The joint policy for an NED-MDP is a mapping from the joint state space to a joint action for every time step, just as in a DEC-MDP. The following theorem states that the value of a joint policy is the sum of the values of individual policies of agents 1 and 2.

Theorem 1. *Let π be the joint policy of the two agents in an NED-MDP, and let π_1 and π_2 be the individual policies for Agent 1 and 2, respectively. The value of the joint policy π in Equation 1 is the aggregate of the values of the individual policies for agents 1 and 2:*

$$V^{\pi}(s_1^h, s_2^h) = V_1^{\pi}(s_1^h, s_2^h) + V_2^{\pi}(s_1^h, s_2^h) \tag{5}$$

where the value for Agent 1 of choosing the policy π in state $s^h = (s_1^h, s_2^h)$ is

$$V_1^{\pi}(s_1^h, s_2^h) = R_1(s_1^h) + \sum_{(s_1^{h+1}, s_2^{h+1})} T\big((s_1^{h+1}, s_2^{h+1}) \mid (s_1^h, s_2^h), \pi(s_1^h, s_2^h)\big) \cdot V_1^{\pi}(s_1^{h+1}, s_2^{h+1}) \tag{6}$$

(and similarly for Agent 2).

The proof is presented in the Appendix.

Corollary 2. *An optimal policy for initial states (s_1^0, s_2^0) maximizes Equation 5 as given below,*

$$\pi^*(s_1^0, s_2^0) = \arg\max_{(a_1,a_2)} (R_1(s_1^0) + R_2(s_2^0) + \begin{cases} \sum_{(s_1^1, s_2^1)} T_1\big(s_1^1 \mid s_1^0, (a_1, a_2)\big) \cdot T_2(s_2^1 \mid s_2^0, (a_1, a_2)) \\ \cdot (V_1^{\pi}(s_1^1, s_2^1) + V_2^{\pi}(s_1^1, s_2^1))) & \text{if } (a_1, a_2) \in \mathbf{A}^J \\ \\ \sum_{(s_1^1, s_2^1)} T_1(s_1^1 \mid s_1^0, a_1) \cdot T_2(s_2^1 \mid s_2^0, a_2) \\ \cdot (V_1^{\pi}(s_1^1, s_2^1) + V_2^{\pi}(s_1^1, s_2^1))) & \text{otherwise} \end{cases} \tag{7}$$

3.3 Modeling Interruption Management as a Nearly-Decomposable Decision-Making Problem

This section describes an experimental setting for studying an interruption management problem that is modeled as a NED-MDP. The "interruption game" [Kamar and Grosz, 2007] involves two players, referred to as the "principal" and the "agent". Each player needs to complete an individual task but the players' scores depend on each other's performance.

The game is played on a board of 6x6 squares. Each player is allocated a starting position and a goal position on the board. The game comprises a fixed, known number of rounds. Players advance on the board by moving to an adjacent square. The players' goals move stochastically on the board according to a Gaussian probability distribution centered at the current position of the goal.[2] Players earn 10 points in the game each time they move to the square on which their assigned goal is located. After a goal is reached, the goals are reassigned to random positions on the board. Players can see their positions and the goal location of the other player, but they differ

[2]The movement of the goal is restricted in that it does not move closer to the position of the player.

in their ability to see their own goal location: The principal can see the location of its goal throughout the game, while the agent can see the location of its goal at the onset of the game, but not during consecutive rounds.

At each round, both players can move their icon one position on the board. In addition, the agent can choose to interrupt the principal and request the current location of its goal. The principal is free to accept or reject an interruption request. If the principal rejects the interruption request, the players continue moving. If the interruption is accepted by the principal player, the location of the agent's goal in the current round (but not in consecutive rounds) is automatically revealed to the agent. There is a joint cost for revealing this information to the agent, in that neither participant will be able to move for one round. When making a decision whether to accept an interruption request, the principal can also observe the belief of the agent about the location of its goal.

The game is collaborative in that the score for each player depends on a combination of its own performance and the performance of the other player. The players share a joint score function that is the cumulative score of both players. An interruption is potentially beneficial for the individual performance of the agent, who can use this information to direct its movement, but it may affect the principal's performance negatively. When the agent deliberates about whether to ask for information, or when the principal deliberates about whether to accept an interruption to reveal the agent's goal, the players need to weigh the trade-offs associated with the potential benefit to the agent player with the detriment to their individual performance in the game. The success of both players in the game depends on the agent's ability to estimate the collaborative value of interrupting at any point in the game and use that information to choose wisely when to interrupt the principal.

Figure 2 shows the game GUI from the perspective of each participant and illustrates the nearly-decomposable structure of this group activity. The right-hand side of the figure shows the individual task of the principal, getting the *me* icon to the goal G_{me}. It also shows the location of the agent (the *smiley* icon) and the agent's goal G_{smiley}. The degree to which a square is shaded represents the agent's uncertainty about its goal location, giving the person information about the agent's belief state. Dark shaded squares imply higher certainty about the goal's location. The left-hand side of the figure shows the individual task of the agent, showing only the location of the agent and the goal of the principal.

The interruption game is analogous to the types of interactions that occur in collaborative settings involving a mixed network of computer agents and people. For example, the principal player in the interruption game may correspond to the user of a collaborative system for writing an academic paper, and the agent to a the collaborative assistant responsible for obtaining bibliographical data [Babaian et al., 2002].[3] While participants share a common goal of completing a document, each of them works independently to complete its individual task, such as composing paragraphs or searching for bibliographical information. This aspect is represented in the interruption game by assigning an individual goal for each player. The movement of these goals on the board corresponds to the dynamic nature of these tasks. For example, the user may not know what to write next, and the system may have uncertainty about search results. The agent's lack of information about its own goal location in the game corresponds to the uncertainty of a system about the preferences and intentions of its user, such as which bibliographical information to include in the paper. The ability to query the user for keywords and to choose among different bibliographies provides the system with valuable guidance and direction. It may, however, impede the performance of both participants on their individual tasks, because the system needs to suspend its search for bibliographical data when it queries the user, and the user may be distracted by the query.

[3] Interruptions are excluded from the original system described in Babaian et al. [2002] to prevent initiating interruptions at inappropriate times, since the system lacked a well-founded or formal approach for managing interruptions.

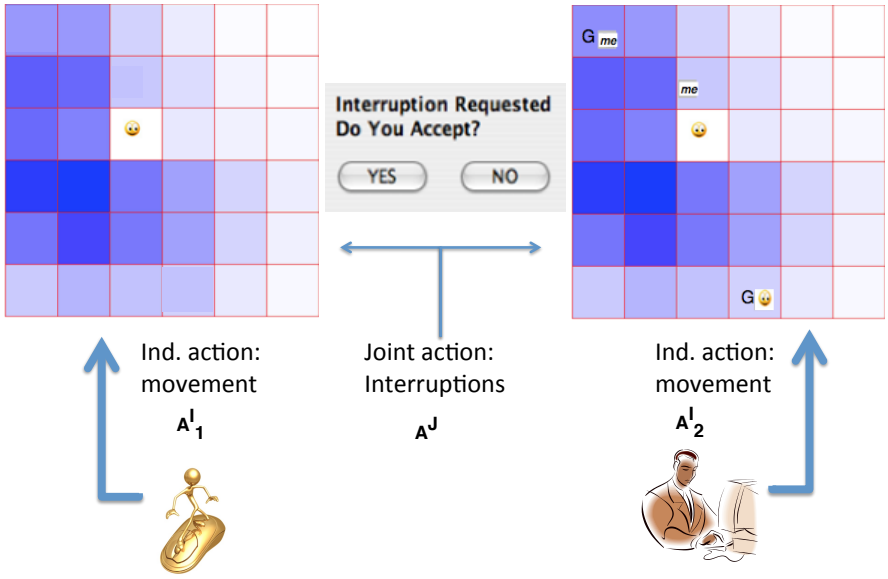

Figure 2: Individual and joint actions in the interruption game.

This dynamic cost of interruption represents the costs incurred by both people and computer agents due to task switching and task recovery for initiating and responding to an interruption.

We used the interruption game, rather than a real-world domain or application, because it is the simplest setting in which to study the factors that influence interruption management in collaborative settings. The rules of the game are easy for people to learn, and provide incentives for players to reach their goals as quickly as possible. Despite its simplicity, computing the value of an interruption request in the game is challenging. The agent's uncertainty about the location of its own goal increases over time, and its performance depends on successfully querying the principal and obtaining the correct position of its goal. A straightforward computation of the value of an interruption request requires the agent to consider all possible combinations of the future actions and the possible transition states of both agent and person. Because a player's future state is determined by the joint action taken by players, the game is not transition independent. Both actions need to be optimized together, significantly increasing the computational complexity. Kamar and Grosz [2007] have shown that the interruption game can be modeled as a Dec-MDP, and the complexity of finding optimal solutions to Dec-MDPs is known to be NEXP-complete [Papadimitriou and Tsitsiklis, 1987]. Furthermore, existing models with lower complexity (such as the ones described in Section 2) cannot be used to model the interruption game because the value of an interruption needs to consider all possible combinations of agents' future actions and the assumptions made by these models are not satisfied in the interruption game.

We now show that the interruption game can be defined as a Nearly Decomposable Decision-Making problem. A NED-MDP can formalize an interruption game in which the agent and the person players have different observations about the world. For easiness of presentation, we formalize below a version of the interruption game in which both players cannot observe the goal position of the agent but the position is revealed when an interruption is established. In Section 4, we will show that the nearly decomposable representation of the interruption game can significantly reduce the computation that is required to compute the value of an interruption request in the game.

- The state space $\mathbf{S}$ is factored into two components $\mathbf{S_P} \times \mathbf{S_A}$.

- $\mathbf{S}_P$ is the set of states for the principal and includes $\langle p_P^h, g_P^h \rangle$, where $p_P^h, g_P^h \in B$ are the positions of the principal and its goal at round h, respectively, and B denotes the set of board positions.

- $\mathbf{S}_A$ is the set of states for the agent and includes $\langle p_A^h, b^h \rangle$, where $p_A^h \in B$ is the position of the agent and b^h is the agent's belief over its goal positions at round h. The term $P(g_A^{h+1} \mid p_A^{h+1}, g_A^h)$ represents the probability of the goal position moving from g_A^h to g_A^{h+1} when the player is in position p_A^{h+1}. This distribution is updated at each time step to reflect the agent's changing belief about the position of its goal, as shown below:

$$\forall g_A^{h+1} \in B,\ b^{h+1}(g_A^{h+1}) = \sum_{g_A^h \in B} b^h(g_A^h) \cdot P(g_A^{h+1} \mid p_A^{h+1}, g_A^h) \tag{8}$$

  Intuitively, as the game progresses, the uncertainty of the agent about its goal location will increase if it does not receive information about its goal from the principal. Although the belief state is continuous, we only need to consider a finite number of states, those that are reachable with non-zero probability.

- The set of actions $\mathbf{A}_P$ for the principal include its movement actions on the board and whether to accept an interruption request from the agent. The set of actions $\mathbf{A}_A$ for the agent include its movement actions on the board and whether to initiate an interruption request. The set of independent actions $\mathbf{A}_P^I$ for the principal agent is a subset of $\mathbf{A}_P$, and contains its possible movement actions on the board. The set of independent actions $\mathbf{A}_A^I$ for the agent is similar. $\mathbf{A}^J$ includes a single joint action pair, $\langle$interrupt, accept$\rangle$. Because all participants have the same observations about the state of the world, they will not miscoordinate given that both are solving the same NED-MDP. Therefore the action pair $\langle$interrupt, reject$\rangle$ is excluded from this representation. (When players have different observations, the $\langle$interrupt, reject$\rangle$ action pair must be included.) In practice, participants may not use the same model to reason about the game (such as when playing with people). In this case miscoordinations are possible, because the principal may reject the agent's interruptions even though they have the same observations. (We studied this behavior empirically in Section 6).[4]

- The transition function for the principal is denoted as $T_P\big(s_P^{h+1} \mid s_P^h, (a_P, a_A)\big)$ where $s_P^h, s_P^{h+1} \in \mathbf{S}_P$ represent states at rounds h and $h+1$ for the principal; a_P represent an action for the principal; a_A denotes an action for the agent. If the joint action constituents (a_P, a_A) are individual actions (i.e., movement on the board), then we have $a_P \in \mathbf{A}_P^I, a_A \in \mathbf{A}_A^I$ and by Equation 4 the transition function for each participant depends solely on its own actions. In this case, the transition function for the principal advances location and goal from its position in turn h to its position in turn $h+1$, as defined by the probability distribution over locations on the board.

$$T_P\big(s_P^{h+1} \mid s_P^h, (a_P, a_A)\big) = P(g_P^{h+1} \mid p_P^{h+1}, g_P^h) \tag{9}$$

  where $p_P^{h+1} = p_P^h + a_P$. The transition function for the agent assigns probability 1 to state s_A^{h+1}, which includes its position on the board p_A^{h+1} (the location of the agent after it performs a movement action

[4]Note that since there is no cost in this game for interrupting but only for accepted interruptions, if there were such an action, the agent might interrupt indefinitely until the person accepts.

a_A^h), and b^{h+1}, the distribution over the agent's goal updated using its prior belief state and the updated position of the agent. T_A is defined as given below:

$$T_A\big(s_A^{h+1} \mid s_A^h, (a_P, a_A)\big) = \begin{cases} 1 & \text{if } p_A^{h+1} = p_A^h + a_A, s_A^{h+1} = \langle p_A^{h+1}, b^{h+1}\rangle \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

where b^{h+1} is updated from b^h with respect to p_A^{h+1} using Equation 8.

If (a_P, a_A) represent joint actions of the principal and agent, we have $(a_P \in \mathbf{A}_P^J, a_A \in \mathbf{A}_A^J)$. If (a_P, a_A) represent an interruption request by the agent that is accepted by the principal, then the principal does not move at turn h since it accepted an interruption request (i.e., $p_P^{h+1} = p_P^h$). In this case the transition function for the principal is as follows:

$$T_P\big(s_P^{h+1} \mid s_P^h, (a_P, a_A)\big) = P(g_P^{h+1} \mid p_P^{h+1}, g_P^h) \tag{11}$$

where $p_P^{h+1} = p_P^h$.

$$T_A\big(s_A^{h+1} \mid s_A^h, (a_P, a_A)\big) = \begin{cases} 1 & \text{if } p_A^{h+1} = p_A^h, b^h(g_A^h) = 1, s_A^{h+1} = \langle p_A^{h+1}, b^{h+1}\rangle \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

That is, the agent does not move at round h and has knowledge about its true goal location, and b^{h+1} is updated using Equation 8.

- The reward function is defined as $R(s^h) = R_P(s_P^h) + R_A(s_A^h)$ where the reward function for the principal, R_P, equals 1 when the principal reaches its goal, as shown below,

$$R_P(s_P^h) = \begin{cases} 1 & \text{if } p_P^h = g_P^h \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

In contrast to the principal, the agent has partial information about the state of the world, and cannot observe its goal. Therefore it uses its belief state about the location of the goal as an approximation to the reward. The reward function $R_A(s_A^h)$ for the agent is thus defined as the likelihood of the agent's goal being at the same position as the agent according to its belief state.

$$R_A(s_A^h) = b^h(p_A^h) \tag{14}$$

An optimal joint policy for a NED-MDP $\pi^* = (\pi_P^*, \pi_A^*)$ is a tuple composed of local policies for each agent. For the principal this policy is a mapping from $\mathbf{S}_P \times \mathbf{S}_A$ to $\mathbf{A}_P$, and the policy of the agent is a mapping from $\mathbf{S}_P \times \mathbf{S}_A$ to $\mathbf{A}_\mathbf{A}$. Thus, the joint policy for the principal and the agent will specify (1) the movement actions on the board, and (2) the times in which it is optimal for the agent to interrupt the principal.

4 Computing Coordination Strategies using NED-MDPs

In this section we present an algorithm that uses the NED-MDP model to efficiently compute information exchange strategies for agents. The algorithm, NED-DECOP (the DECOuPling algorithm for NEarly Decom-

posable Processes) decouples an NED-MDP into individual models that are linked to coordinate agents' joint actions. The algorithm *NED-DECOP* exploits the special structure of nearly decomposable planning problems to overcome the intractability of existing approaches for solving these problems. Given the nearly decomposable structure a NED-MDP, and that agents act individually more often than they act jointly, agents only consider their local state when optimizing their individual actions. Importantly, the algorithm makes the simplifying assumption that agents do not reason about the possible discrepancies in each other's models that may result from differences in their observations. In the interruption game, this means that the agent will not consider the possible discrepancy in the observations of the person and the agent. This significantly facilitates the computation of the agents' policies, because it will not have to consider the possible discrepancy in the observations of the person and the agent. Throughout this section, the states for both participants (S_1 and S_2) are composed of the agent's observations about these tasks (the participant that is assumed to be using the algorithm to make its decisions).

There are two building blocks to the NED-DECOP algorithm, a *query* model and a *coordination* model. The query model is used to compute the optimal policy for a single agent given a set of strategies that fix when the agent acts individually and when it acts jointly with another agent. The coordination model uses the query model to compute the joint value (values combined for both agents) for a given sequence of individual and joint actions.

We begin with definitions of several terms used in specifying the models. Given an agent m, $\mathbf{A}_m^I$ is the set of individual actions for m. $\mathbf{A}$ is the expanded set of actions that includes agents' individual action sets and the set of joint action pairs. Given two agents this set is defined below,

$$\mathbf{A} = \mathbf{A}_1^I \cup \mathbf{A}_2^I \cup \mathbf{A}^J \tag{15}$$

The function $f_t(a_i)$ denotes the action *type* of $a_i \in A$. The type of a_i equals constant I if $a_i \in \mathbf{A}_m^I$ for agent m (i.e., a_i is an individual action of agent m). The nearly-decomposable structure of NED-MDPs means that we don't need to distinguish among individual actions to optimize coordination decisions, given that individual actions can be optimized independently for each agent. The type of a_i equals J_i if $a_i \in \mathbf{A}^J$ (i.e. a_i is a particular joint action J_i).

$$f_t(a) = \begin{cases} I & \text{if } a_i \in \mathbf{A}_l^I \\ J_i & \text{if } a_i \in \mathbf{A}^J \end{cases} \tag{16}$$

In the interruption game modeled in the previous section, the set $\mathbf{A}_l^I$ include players' movement actions, while the set $\mathbf{A}^J$ includes only the single joint action J_1 which is ⟨interrupt, accept⟩.

The NED-DECOP algorithm abstracts away from agents' specific actions and reasons about action types, rather than actions. We therefore use the notion of a coordination type sequence to represent a set of action types. Formally, a *coordination type sequence* $\mathbf{C}^{h,H-1} = \{t^h, t^{h+1}, \ldots, t^{H-1}\}$ is an ordered set of action types for rounds h through $H-1$, where H is the horizon. The agents take actions in rounds 0 through $H-1$. The type $t^k \in \{I, J_1, \cdots, J_{|A^J|}\}$ denotes the type of action at round k. Superscripts are used to refer to the round number in which an action is taken. We abbreviate and use the term "coordination sequence" for the rest of the paper. We say the actual action sequence $\{a_l^h, \ldots, a_l^{H-1}\}$ *agrees* with type sequence $\mathbf{C}^{h,H-1}$ if $f_t(a_l^k)$ is equal to t^k for all rounds $h \leq k \leq H-1$. In other words, the action types in the coordination sequence at each round $h, \ldots, H-1$ match the types in the action sequence. For example, the individual action "move left" at

round 2, the joint action ⟨interrupt, accept⟩ at round 3, and the individual action "move right" at round 4 agree with the type sequence $\mathbf{C}^{2,4} = \{I, J_1, I\}$.

The query model computes the value for one agent, whom we denote "Agent1" of doing a set of actions that agree with a particular type sequence. To this end, the type sequences are embedded in the state space of the query model, as specified in Definition 3. In actual model construction, as we explain later in this section, the choice of which agent to assign as the query model agent affects the complexity of the algorithm.

Definition 3. The query model for Agent1 is the MDP $\langle \mathbf{S}_q, \mathbf{A}_q, T_q, R_q \rangle$, where

- $\mathbf{S}_q$ is a finite set of states that are derived from the states representing Agent1's task $\mathbf{S}_1$ by appending to each $s_1^h \in \mathbf{S}_1$ a type sequence from round h to $H-1$. For all possible type sequences $\mathbf{C}^{h,H-1} \in \{I, J_1, \cdots, J_{|A^J|}\}^{H-h}$ and all states for Agent1's task $s_1^h \in \mathbf{S}_1$ there exists a state $s_q^h \in \mathbf{S}_q$ such that $s_q^h = s_1^h \cup \mathbf{C}^{h,H-1}$.
- The set of actions A_q is the combination of the individual actions for Agent1 and the set of joint actions, i.e., A_q is defined as $A_q = A_1^I \cup A^J$.
- The transition function also simply extends coordination-sequence information as follows

$$T_q(s_q^{h+1} \mid s_q^h, a_q) = T_1(s_1^{h+1} \mid s_1^h, a_q) \tag{17}$$

  where $a_q \in A_q$ and $s_q^{h+1} = s_1^{h+1} \cup \mathbf{C}^{h+1,H-1}$ and $\mathbf{C}^{h+1,H-1} = \mathbf{C}^{h,H-1} \backslash \{t^h\}$.
- The reward function $R_q(s_q^h) = R_1(s_1^h)$, where $s_q^h = s_1^h \cup \mathbf{C}^{h,H-1}$.

The optimal policy for the query model, denoted π_q^*, specifies the best action for this agent to take in round h under the constraint that its future actions in rounds $h+1, \ldots, H-1$ agree with the type sequence $\mathbf{C}^{h,H-1}$. The term $V^{\pi_q^*}(s_q^h)$ represents the individual value for Agent1 when using the optimal policy, i.e.,

$$V^{\pi_q^*}(s_q^h) = \max_{a_q : f_t(a_q) = t^h} [R_q(s_q^h) + \Sigma_{s_q^{h+1}} T_q(s_q^{h+1} \mid s_q^h, a_q) \cdot V^{\pi_q^*}(s_q^{h+1})] \tag{18}$$

The policy π_q^* that maximizes the value function of a given query model is

$$\pi_q^*(s_q^h) = \arg\max_{a_q : f_t(a_q) = t^h} \Sigma_{s_q^{h+1}} T_q(s_q^{h+1} \mid s_q^h, a_q) \cdot V^{\pi_q^*}(s_q^{h+1}) \tag{19}$$

Definition 4. The coordination model for Agent2 is the MDP $\langle \mathbf{S_c}, \mathbf{A_c}, T_c, R_c \rangle$ which is specified as follows:

- $\mathbf{S_c}$ is a finite set of states derived from $\mathbf{S_2}$ by appending to each state $s_2^h \in S_2$ a type sequence from round 0 to $h-1$ and the initial state s_1^0 of Agent1's task. For all states $s_2^h \in S_2$ and $s_1^0 \in S_1$, and $\mathbf{C}^{0,h-1} \in \{I, J_1, \cdots, J_{|A^J|}\}^h$ there exists a coordination state $s_c^h \in \mathbf{S}_C$, $s_c^h = s_2^h \cup \mathbf{C}^{0,h-1} \cup s_1^o$. The sequence $\mathbf{C}^{0,h-1}$ represents the types of actions that led to the state s_c^h from the initial state s_c^0.
- The set of actions A_c is the combination of the individual actions for Agent2 and the set of joint actions, $A_c = A_2^I \cup A^J$.
- The transition function is defined as $T_c(s_c^{h+1} \mid s_c^h, a_c) = T_2(s_2^{h+1} \mid s_2^h, a_c)$, where $a_c \in A_c$, $s_c^{h+1} = s_2^{h+1} \cup \mathbf{C}^{0,h} \cup s_1^0$ and $\mathbf{C}^{0,h} = \mathbf{C}^{0,h-1} \cup \{f_t(a_c)\}$.

- The reward function is $R_c(s_c^h) = R_2(s_2^h)$, where $s_c^h = s_2^h \cup \mathbf{C}^{0,h-1} \cup s_1^0$.

The value function for the coordination model is denoted $V^{\pi_c^*}$. For any given coordination sequence $\mathbf{C}^{0,H-1}$, the value function queries the query model to obtain the value for Agent1 that is associated with the best actions which agree with the action sequence $\mathbf{C}^{0,H-1}$. Thus, $V^{\pi_c^*}$ represents the joint values of Agent1 and Agent2 for any coordination sequence.

$$V^{\pi_c^*}(s_c^h) = \begin{cases} \max_{a \in A_c}(R_c(s_c^h) + \Sigma_{s_c^{h+1}} T_c(s_c^{h+1} \mid s_c^h, a) \cdot V^{\pi_c^*}(s_c^{h+1})) & \text{if h<H} \\ R_c(s_c^h) + V^{\pi_q^*}(s_1^o \cup \mathbf{C}^{0,H-1}) & \text{if } s_c^h = s_2^H \cup \mathbf{C}^{0,H-1} \cup s_1^o \end{cases} \tag{20}$$

The action that maximizes this value determines whether agents are better off acting jointly rather than acting individually. This is key for computing the value of an interruption request in the interruption game.

For any given state of the coordination model, the policy π_c^* selects the action of the coordination model that maximizes $V^{\pi_c^*}$. This action identifies the coordination strategy for both agents, and optimizes Agent2's action accordingly.

$$\pi_c^*(s_c^h) = \arg\max_{a \in A_c}(\Sigma_{s_c^{h+1}} T_c(s_c^{h+1} \mid s_c^h, a) \cdot V^{\pi_c^*}(s_c^{h+1})) \tag{21}$$

4.1 The NED-DECOP algorithm

The input to the NED-DECOP algorithm is a NED-MDP, the initial states for the query and coordination model, and an assignment of agents to the query and coordination model (indicating which is Agent1 and which is Agent2 in Definitions 3 and 4). The output of the algorithm is the action type the agents should follow in the current round. This decision determines whether agents should act individually or jointly, and which joint action they should take if they are acting jointly. We describe later how we deal with deciding about individual actions.

The NED-DECOP algorithm decouples the collaborative decision-making problem in the NED-MDP into individual problems that are represented in the query and coordination models. Each model contains the state space, reward and transition function for one of the agents, that agent's individual actions, as well as the joint action pairs for both agents. The algorithm comprises the following steps:

1. Generate a query model for the agent in the role of Agent1 according to Definition 3.

2. Generate a coordination model for the agent in the role of Agent2 according to Definition 4.

3. For the given initial state s_1^0 of Agent1 and for all possible type sequences $\mathbf{C}^{0,H-1}$, assign $s_q^0 = s_1^0 \cup \mathbf{C}^{0,H-1}$. Use the query model to recursively compute, for all reachable states from rounds 0 to H, the policy for Agent1, π_q^*, that agrees with $\mathbf{C}^{0,H-1}$ according to Equation 19, and the value of that policy $V^{\pi_q^*}$, according to Equation 18.

4. For the given initial state s_2^0 of Agent2 and the empty type sequence $\{\}$, assign $s_c^0 = s_2^0 \cup \{\} \cup s_1^0$. Recursively call the value function $V^{\pi_c^*}(s_c^0)$ of the coordination model (Equation 20) for all reachable states until reaching the horizon H. When the horizon H is reached, Equation 20 calls the query model to retrieve the policy for Agent1 given the type sequence that is embedded in the coordinate model state s_c^H and the value that is associated with the policy.

5. Output the action $\pi_c^*(s_c^0)$, which maximizes the joint value function in Equation 20 for the initial state s_c^0.

6. Update initial states s_c^0 and s_q^0 for the query and coordination models, given agents' observations at the current round. Repeat the computation starting at step 3 for the next round.

As we have stated in the beginning of this section, the NED-DECOP algorithm does not reason about possible differences in agents' models. Thus, the NED-DECOP algorithm does not allow for agents to pick an action pair that indicates one agent is to act jointly and the other individually. Such an action pair is suboptimal, because it would result in mis-coordination. For instance, consider two robots that may either choose the joint action pair of meeting at the local farmers market to shop for groceries together (because the produce is too heavy for one to carry alone), or take individual actions to buying groceries at the supermarket closest to their individual places of work and who share such relevant information as each other's schedule in a collaborative setting. The action pair consisting of one agent going to the farmer's market and the other agent going to the supermarket is suboptimal, and thus it would never be chosen by these agents.

We illustrate this algorithm in Figure 3, which shows part of the coordination model, which has two individual actions(part a) and the query model, which has three individual actions (part b). Nodes in the tree in Figure 3(a) are labeled with the states of the coordination model. Similarly, nodes in the tree in Figure 3(b) are labeled with the states of the query model. For simplicity, we do not represent all state transitions. The root node of the tree in Figure 3(a), labeled s_2^0, represents the initial state of the coordination model, which includes an empty type sequence. Its left-most child node $s_2^0 \cup \{I\}$ contains the type sequence in which Agent2 chooses an individual action I at the first round. The leaf of this tree at the bottom right includes a type sequence $\{I, J_1, I, J_1\}$, which spans the horizon H. When it reaches this leaf node, the algorithm calls the query model to compute the best policy for Agent1 given that its actions agree with the type sequence embedded in the state that labels this leaf node, as shown in Figure 3(b). The joint value to Agent1 and Agent2 for this type sequence is then propagated up the tree of the coordination model. The actions shown in bold from the root of the tree to the leaves represent the actions chosen by the algorithm to maximize the joint value function.

In addition to optimizing the coordination strategy for both agents, the algorithm also optimizes the action of Agent2. Thus if Agent2 is the agent using the algorithm, the output of the algorithm will specify the actions that Agent2 should take. If Agent1 is using the algorithm, then an additional optimization step is needed to compute its individual actions. However, the algorithm has already determined whether agents are acting individually or when and how they are acting jointly for each state. Thus Agent1 need only focus on optimizing its individual task, which significantly reduces the complexity of computing its policy.

The NED-DECOP algorithm has three advantages over alternative approaches. First, it reasons about the effect of an information exchange actions on future opportunities for coordination, i.e., it is not simply myopic. For example, in the interruption game, using this algorithm will allow the agent to decide not to interrupt the principal at a given round because it expects that generating an interruption in a future round will be more beneficial to the group. Second, the algorithm outputs a coordination strategy that maximizes the value of information exchange actions to the group, rather than considering only a single agent and the benefit to that agent. Third, it does so efficiently by decomposing the group decision-making problem into the query and coordination models.

The complexity of the NED-DECOP algorithm is $(k+1)^H \times \Omega + \Theta$, where Ω is the complexity of solving the query model, Θ is the complexity of solving the coordination model, $k = |A^J|$ is the number of joint actions and H is the horizon. The term $(k+1)^H \times \Omega$ results from the exponential number of queries made to the query model each time the coordination model is solved. Since the query model is computed exponentially

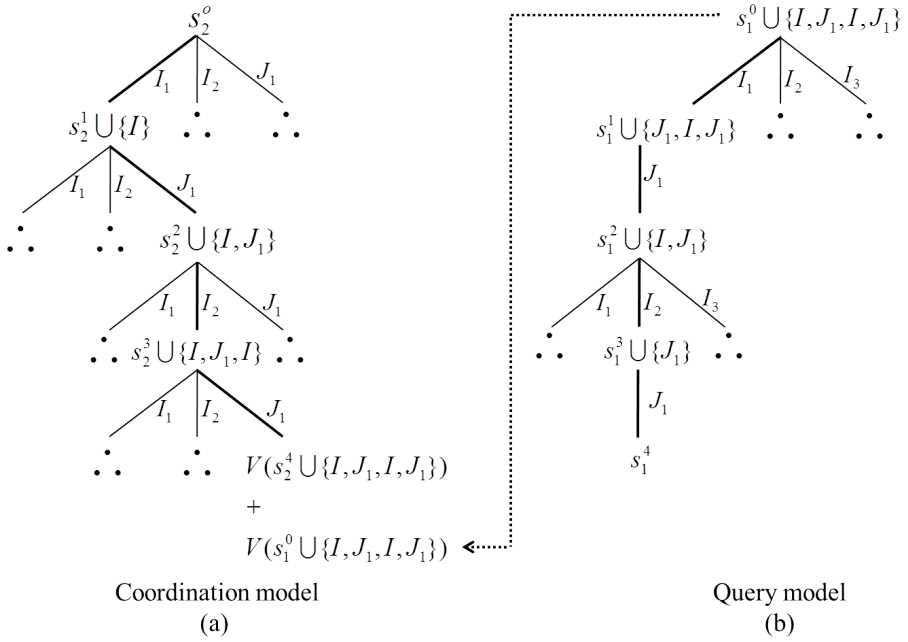

Figure 3: An illustration of the solution algorithm for NED-MDPs, showing (a) the coordination model (left), and (b) the query model (right).

many times, the complexity of the query model Ω is the largest factor contributing to the complexity of the NED-DECOP algorithm. Thus, it is important that the query model be used to represent the simpler of the two agents' individual problems. If one (or both) of the individual problems is a fully observable MDP, the query model can be solved in polynomial time. In this case the complexity of the NED-DECOP algorithm is exponential in the horizon, and the algorithm is able to achieve exponential savings over the general DEC-MDP approach, which is doubly exponential in the horizon.

Figure 4 compares the expected running time of the NED-DECOP algorithm with an exact solution algorithm for computing the best action to take at a randomly selected initial state of the interruption game for varying board sizes and horizons. The exact solution algorithm used in this comparison, called Forward-Sweep Policy Computation (FSPC), performs dynamic programming to generate a policy iteratively from end states to the initial state to optimize the best action to take at a given state [Oliehoek, 2012]. Since this algorithm needs to evaluate joint policy pairs, the complexity of the FSPC algorithm is doubly exponential in the horizon. As shown by the figure, the running time of the NED-DECOP algorithm is exponentially less than the complexity of the exact solution algorithm for the different game instances we sampled.

Lastly, we describe how the NED-DECOP algorithm implements two tenets of foundational collaborative decision making models [Bratman, 1992, Grosz and Kraus, 1996]. First, an agent needs to reason about the effect of joint actions on the performance of its partner's task. This is realized by optimizing a coordination type sequence for the collaboration rather than optimizing individual actions. Second, these foundational decision making models do not require an agent to know the details of the model of its partner. This is realized in that the algorithm does not reason about the possible discrepancies in agents' observations about the state of the world. Doing so enables decoupling the decision-making problems and making the algorithm efficient. In practice, this

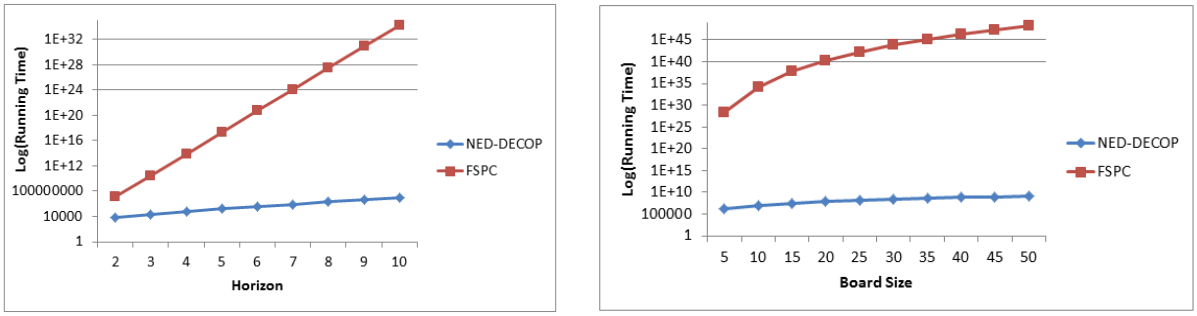

Figure 4: Comparing NED-DECOP and FSPC performance

assumption may not always hold. For example, in the interruption game, the person may observe the location of a goal that the agent can not observe. Reasoning about the differences in participants' observations would exponentially increase the complexity of the algorithm. However, if the assumption made by the algorithm about agents models being identical holds in a given setting, the decisions of the algorithm about when and how to coordinate are optimal.

4.2 Possible Extensions to the NED-DECOP Algorithm

This section describes three extensions to the NED-DECOP algorithm to allow for more nuanced models of information exchange. The first addresses a different way to reason about the information that is available to agents when they make decisions; the second shows how to extract strategies for all group participants; the third deals with adding agents to the group. Each of these extensions are straightforward.

First, participants in a collaborative activity typically have different information about the world. For example, in the interruption game, the principal player is able to observe the agent's goal but the agent is not able to observe its own goal. As a result, the principal may assign a different value to an interruption request than the agent and an interruption generated by the agent player may not be seen as having positive benefit by the person. An interruption requested by the agent may be rejected. To address this discrepancy, the NED-MDP model given in Section 3.3 can be updated to include joint action pair $\langle$interrupt, reject$\rangle$ to reason about the fact that the agent's goal position is observable to the principal player. Consequently, an additional transition can be incorporated into the coordination model that maps the agent's belief about the world to a set of fully observable states, representing the principal's perspective, and information sets can be added to represent the set of states that are indistinguishable to the agent. In brief, the NED-DECOP algorithm can be extended to use this modified coordination model to compute an expected value for each of the agent's action by probabilistically transitioning to each of the fully observable states. This extension is one of the ways an agent may reason about information discrepancy in a collaborative activity. Reasoning about this discrepancy formally in a multi-agent decision making problem may significantly increase the complexity of finding a solution, and the complexity of the NED-DECOP algorithm may be affected by such extensions. However, compared to general solution algorithms, the NED-DECOP algorithm would still be more efficient in generating coordination strategies for nearly-decomposable problems, because it decouples coordination from acting individually.

Second, if strategies for both agents were desired as the outcome of the NED-DECOP algorithm, the action space for the coordination model would explicitly include all joint action pairs (rather than just their types),

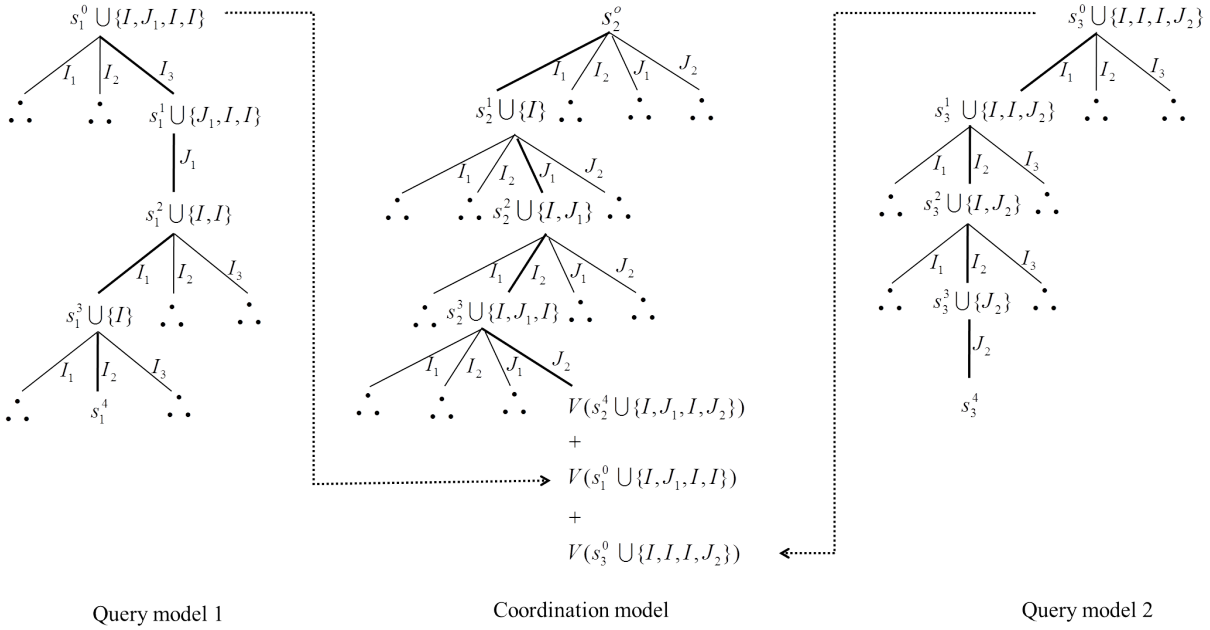

Figure 5: Solving NED-MDP for 1-Many Interactions

and the type function would distinguish between possible individual actions as well as pairs of joint actions. Although this increases the complexity of the algorithm, the increase is not exponential in the size of the state space. It only increases the branching factor of the search tree to the number of all possible action pairs.

The third extension is illustrated in Figure 5. This figure shows how to generalize the algorithm and incorporate multiple query models to make coordination decisions in settings in which one agent can coordinate with multiple others. These one-to-many interactions may occur in settings where a single user is interacting with multiple computer processes, or a computer process may request information from multiple users. In this case, the coordination model represents an agent that is providing resources to several partners on request. An individual query model is generated for each partner agent. Given that agent i is the resource agent, and agent j is one of its partners, the set of joint actions is $A^J_{q_j} = A^J_i \times A^J_j$. The set of joint actions for the coordination model is $A^J_c = \bigcup_{j \neq i} A^J_i \times A^J_j$. Thus, the branching factor of the coordination model grows linearly with respect to the number of agents the coordination agent can interact with.

As in the case of the 2-agent algorithm, the complexity of the one-to-many extension of NED-MDP is bounded by the complexity of solving the most "expensive" query model. It does not grow exponentially with the number of agents in the model. Thus it is significantly more efficient than traditional collaborative planning approaches for which complexity is exponential in the number of agents. However, these properties of NED-MDP models do not generalize to model many-many interactions among agents (i.e., multiple resources are connected to multiple partners) when both agents need to track each other's models.

4.3 Estimating the Value of Interruption in the Interruption Game

Section 3.3 showed how to use NED-MDPs to model the interruption game and Section 4.1 showed how to generate information exchange strategies in the game using the NED-DECOP algorithm. In this section we

show how to estimate the value to agents that is associated with generating interruption requests in the game. Recall that (s_P^h, s_A^h) are the state of the principal's and agent's tasks based on the agent's observations about the game. The agent's estimate of the expected utility to the group with generating an interruption request, denoted $EU_A^I(s_P^h, s_A^h)$, is defined as follows:

$$EU_A^I(s_p^h, s_A^h) = \sum_{g_A^h} b^h(g_A^h) \cdot V^{\pi_c^*}(s_c^{h+1}) \tag{22}$$

where $s_c^{h+1} = \langle s_A^{h+1} \cup \langle \text{interrupt, accept} \rangle \cup s_p^h \rangle$, and $s_A^{h+1} = \langle p_A^h, b^{h+1} \rangle$. The term $V^{\pi_c^*}(s_c^{h+1})$ is computed in the NED-DECOP algorithm of Section 4.1 in Step (4). The term b^{h+1} refers to the belief state of the agent in round $h+1$, in which probability 1.0 is given to its true goal location g_A^h and updated using Equation 8 to reflect the stochastic movement of the goal in turn h.

If the agent decides not to generate an interruption request, it chooses the individual action that maximizes the joint value function. In this case, the agent's estimate of the value to the group that is associated for acting individually is denoted $EU_A^{NI}(s_P^h, s_A^h)$ and is defined as follows:

$$EU_A^{NI}(s_P^h, s_A^h) = max_{a_A^h \in A_A^I} V^{\pi_c^*}(s_c^{h+1}) \tag{23}$$

where $s_c^{h+1} = \langle s_A^{h+1} \cup \{\text{Individual Action}\} \cup s_P^h \rangle$, and $s_A^{h+1} = \langle p_A^{h+1}, b^{h+1} \rangle$. The term $p_A^{h+1} = \langle p_A^h + a_A^h \rangle$ refers to the position of the agent after taking the individual action a_A^h, and b^{h+1} refers to the updated belief state of the agent at round $h+1$.

The agent's estimate of the expected value of an interruption request to the group, denoted $EBI(s_P^h, s_A^h)$, is the difference between EU_A^I and EU_A^{NI}.

$$EBI(s_P^h, s_A^h) = EU_A^I(s_P^h, s_A^h) - EU_A^{NI}(s_P^h, s_A^h) \tag{24}$$

Any interruption request that is generated by the algorithm has positive $EBI(s_P^h, s_A^h)$ value as computed above.

In contrast to the agent, the principal has full information about the game, including the positions and goals of all participants. Therefore the principal can better estimate the value of an interruption at a given state of the game without having uncertainty about the information the agent is missing. The *Actual Benefit of an Interruption* (ABI) term is equal to the actual value to the group that is associated with an interruption request given that there is no uncertainty about the information that is provided following the interruption (i.e., the true location of the agent's goal on the board), under the assumption that both players follow the policy computed by the agent in following rounds of the game. This term is the difference between the principal's estimate of the joint value when an interruption is established ($EU_P^I(s_P^h, s_A^h)$), and respectively, when an interruption is not established ($EU_P^{NI}(s_P^h, s_A^h)$), given the joint state (s_P^h, s_A^h).

$$ABI(s_P^h, s_A^h) = EU_P^I(s_P^h, s_A^h) - EU_P^{NI}(s_P^h, s_A^h) \tag{25}$$

Under the assumption that players will not miscoordinate in future rounds of the game, it is beneficial for the person to accept an interruption request, if the interruption is associated with a positive ABI value.

The following equations show how to compute the EU_P^I and EU_P^{NI} values when using the NED-MDP that is described in Section 3.3. The term $EU_P^I(s_P^h, s_A^h)$ is the principal's estimation of the value that is associated

with successfully interrupting the principal player at round h and updating the agent's belief state about the true location of its goal at round h.

$$EU_P^I(s_p^h, s_A^h) = V^{\pi_c^*}(s_c^{h+1}) \tag{26}$$

where $s_c^{h+1} = \langle s_A^{h+1} \cup \langle \text{interrupt, accept} \rangle \cup s_p^h \rangle$, and $s_A^{h+1} = \langle p_A^h, b^{h+1} \rangle$. The term b^{h+1} refers to the belief state of the agent in round $h+1$, in which probability 1.0 is given to its true goal location g_A^h and updated using Equation 8 to reflect the stochastic movement of the goal in turn h.

The term $EU_P^{NI}(s_P^h, s_A^h)$ is the principal's estimation of the value to the group that is associated for not having an interruption at the current round, i.e.,

$$EU_P^{NI}(s_P^h, s_A^h) = max_{a_A^h \in A_A^I} V^{\pi_c^*}(s_c^{h+1}) \tag{27}$$

where $s_c^{h+1} = \langle s_A^{h+1} \cup \{\text{Individual Action}\} \cup s_P^h \rangle$. Here, $s_A^{h+1} = \langle p_A^{h+1}, b^{h+1} \rangle$, $p_A^{h+1} = \langle p_A^h + a_A^h \rangle$ refers to the position of the agent after taking the individual action a_A^h, and b^{h+1} refers to the updated belief state of the agent at round $h+1$.

Because the agent cannot observe its goal without interrupting the principal, not every interruption initiated by the agent is truly beneficial to the players. For example, it may be the case that an interruption request is generated by the agent, because a joint state is associated with a positive EBI value, while the actual benefit of this interruption, i.e., the ABI value, is negative. In settings in which future miscoordiantions are not possible, it is rational for the principal player to accept interruptions with positive ABI values as computed above and to reject interruptions with negative ABI values. The following section visits an empirical setting in which this condition holds (i.e., no future miscoordinations possible) in the calculation of ABI values. It studies the extent to which people deviate from these predictions when playing the role of principal players, and it identifies the factors that affect their actual behavior.

5 Empirical Methodology

This section describes an experiment in which an agent played the interruption game with people. Our purpose was to measure the extent to which different factors in the game, such as the collaborative benefit of interruption (ABI), the timing of interruptions and the perceived partner type, influence people's perception of the value of interruptions. The version of the game we used in our experiments had 10 rounds in total (i.e., the game ended after the principal and the agent acted 10 times including movements on the game board and interruptions), where, for instance, the players reach round 5 after acting 4 times in the game. A total of 26 subjects participated in the study. The subjects were between ages of 19 and 46 and were given a 20 minute tutorial of the game. Subjects played between 25 and 35 games each, and were compensated in a manner that was proportional to their total performance. People assumed the principal role. Each game proceeded as described in Section 3.3. The agent could not observe its own goal location, but could initiate one interruption to acquire the correct location of its goal from the principal. To minimize the effect of people's past interactions on their actions (e.g., regret from rejecting or accepting an interruption), the interruption game was constrained to allow one interruption request per game.[5] Because no further interruptions are allowed, miscoordination of agents' actions in future

[5]In effect, we computed Equation 20 using the algorithm presented by Kamar and Grosz [2007]. For the case where there is only one interruption, this algorithm gives the same result as the NED-DECOP algorithm.

PP:Computer	Round 3	Round 5	Round 7
ABI -1.5	0.16	0.16	0.41
ABI 1.0	0.27	0.7	0.81
ABI 3.5	0.91	0.97	0.79
ABI 6.0	0.91	0.95	0.95
PP:Person	Round 3	Round 5	Round 7
ABI -1.5	0	0.11	0.44
ABI 1.0	0.72	0.94	1
ABI 3.5	0.91	0.94	0.88
ABI 6.0	1	0.88	1

Table 1: Acceptance Rate of Interruptions

rounds of the game will not occur. Thus, it is beneficial to the collaborative activity to accept any interruption request with a positive ABI value as computed in Section 4.3.

Interruptions were generated by the computer agent at different points in the game with varying actual benefit, round number and the subjects were given different perceived partner types. We measured people's responses to these requests given the game conditions at the time of interruptions, which included the number of turns left to play, the positions of both players on the board, and the agent's belief about the location of its goal. To investigate the way subject responses change with respect to the (perceived) benefit of interruptions, the game scenarios were varied to have different ABI values.

To investigate the effect of the timing of an interruption on the subjects' likelihood of acceptance, we varied the round of the game at which an interruption was initiated. We hypothesized that people's responses to interruptions agree more with the fully-rational baseline at later rounds of the game as the complexity of decision-making decreases. We also expected that the type of the agent player (whether a computer or human) would affect the way people respond to interruption requests. For this reason, in some of the games, subjects were told they would be interacting with other people. However, subjects were always paired with an agent.[6]

Subjects were given randomly generated game scenarios that varied the actual benefit of interruption (ABI) over four types of values: -1.5 (small loss), 1.0 (small gain), 3.5 (medium gain), 6.0 (large gain). The rounds in the game in which interruptions occurred varied from the beginning of a game (round 3), to the middle of a game (round 5), to the end of a game (round 7). There were 540 game instances played when the perceived agent was a computer and 228 data points when the perceived agent was a person.

5.1 Results and Discussion

The results presented in this section are all statistically significant using the pairwise sign test. This test takes as input paired samples from two distributions and tests whether there is a difference in medians between the distributions [Conover, 1999]. The effect of a single factor is measured by considering all matched pairs of data points that differ only in that factor (e.g., (ABI=1.0, level=3, PP=person) vs (ABI=3.5, level=3, PP=person)). For example, to determine the effect of ABI, we compared all data points with different ABI values but fixed game level and perceived partner type.

The optimal policy for the principal player is to accept an interruption if its associated benefit (ABI) is positive and to reject otherwise. Table 1 shows interruption acceptance rates for different rounds and ABI

[6]This misinformation was approved by the committee for the use of human subjects at Harvard University.

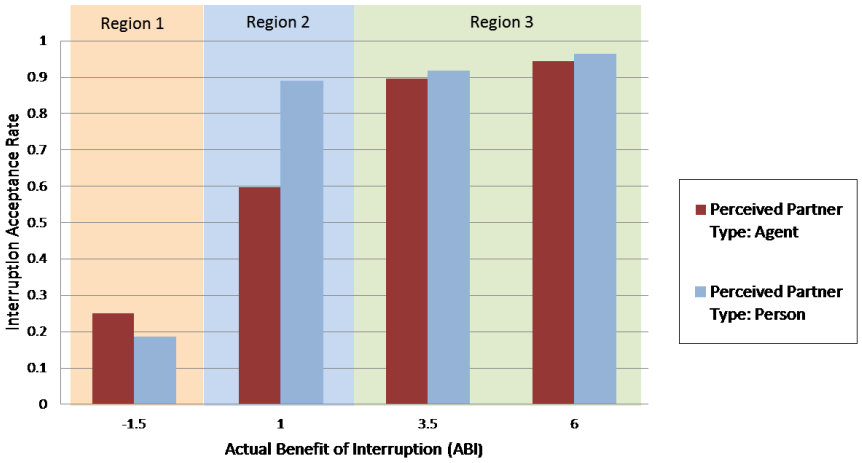

Figure 6: Effect of interruption benefit and perceived partner type on interruption acceptance rate

values across game instances. It is divided by perceived partner type (person or agent). As the results of Table 1 show, the utility of an interruption is the major factor influencing the probability that an interruption will be accepted by a person. The interruption acceptance rate increases significantly as the benefit of interruption rises from -1.5 to 1.0 ($p < e^{-20}$, α=0.001) and from 1.0 to 3.5 (p=0.0013, α=0.01). However the rise from 3.5 to 6.0 does not further improve the acceptance rate. Thus, these results confirm that people are successful at perceiving interruption benefits above a threshold (in this case 3.5). Furthermore, when an interruption is costly for the collaboration, people are significantly more likely to reject the interruption. However, subjects varied in their responses to interruptions offering slightly positive gains (i.e., ABI=1.0), indicating the difficulty of estimating the benefit of interruption when its usefulness is not clear. These results indicate an asymmetry in the way subjects perceive ABI in that interruptions offering small losses are clearly rejected, but subjects do not respond to interruptions with small gains so uniformly.

Figure 6 summarizes the acceptance rates of interruption as a function of the actual benefit of interruption and perceived partner type (person vs. computer). We divide the figure into three regions of interruption benefits: small losses (Region 1), small gains (Region 2), and large gains (Region 3). The analysis shows that for losses (Region 1) and for large gains (Region 3), changing the perceived partner type does not affect the likelihood that the interruption will be accepted. In contrast, for interruptions offering small gains (Region 2), the acceptance rate is significantly larger if the perceived partner type is a person ($p = 3 \times 10^{-5}$, $\alpha = 0.001$).

The results above show that when the benefit of interruption is straightforward, people do not care whether the initiator of the interruption is a person or a computer. However, for those cases in which the benefit of interruption is ambiguous, people are more likely to accept interruptions that originate from other people. These results align with recent studies showing that mutual cooperation is more difficult to achieve in human-computer settings as compared to settings involving people exclusively [Rilling et al., 2004].

Figure 7 shows the effect of interruption timing (the round of the game) on people's acceptance rates for interruptions of ambiguous benefit. We expected that interruptions occurring late in the game (i.e., with fewer rounds left to play) will be more likely to be accepted when they incur positive benefit, and rejected when incurring a loss. However, as shown by the Figure, as the game round increases, so does the acceptance rate for interruptions of both small losses (ABI -1.5) and small gains (ABI 1.0). There is a significant increase

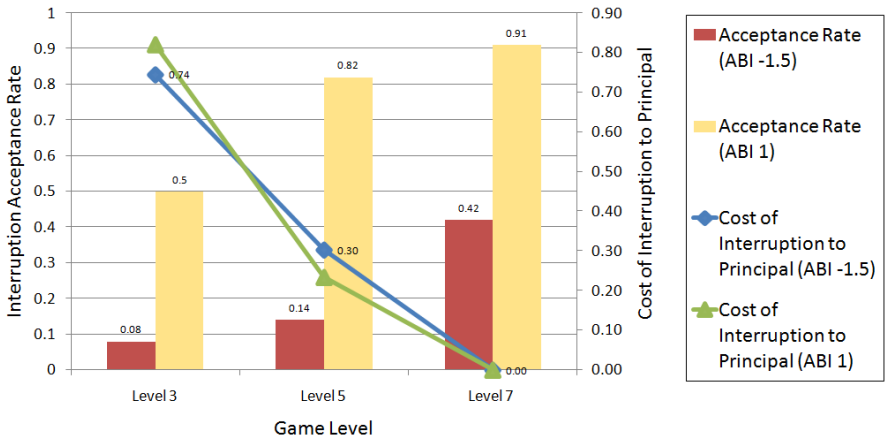

Figure 7: Correlation of interruption acceptance rate with the cost of interruption to subjects (-ABI_P) for ambigous benefit.

in the acceptance rate when game round increases from 3 to 5 (p=0.002, α=0.01) and from 5 to 7 (p$< 10^{-6}$, α=0.001).

Given that our hypothesis about the effect of the game round on the interruption acceptance rate, we considered the influence of the following factors on people's decisions to accept or reject interruption requests: the actual benefit of an interruption to the person (to the person's individual task performance), and the actual benefit of an interruption to the agent (to the agent's task performance).

The interruption game is collaborative; players share a joint reward function. However, an interruption offers different benefits (or costs) to the players' individual task performances, which suggests that the cost of an interruption to the principal's individual task ($-ABI_P$) may explain the correlation between the acceptance rate and the game round for interruptions of small gains and losses. We explored this hypothesis on the dataset collected from the interruption game.

We refer to ABI_P and $-ABI_P$ as the benefit and cost of an interruption to the principal, respectively. As shown in Figure 7, the cost of interruption to the principal ($-ABI_P$) decreases as game round increases. Thus, for interruptions of ambiguous benefit, we found that the acceptance rate is negatively correlated with the cost of interruption to the principal. In addition, the benefit of the interruption to the principal (ABI_P) was a better predictor of the acceptance rate than ABI_A, the benefit of interruption to the agent (logistic regression $SE = 0.05, R^2 = 0.19, p < 0.001$). Thus, human subjects tend to overestimate the importance of their own benefit from interruptions as compared to the benefit for the agent. Consequently, the benefit of an interruption to the person may be weighted more in people's decision making model than the benefit of the interruption, and people may be more likely to accept an interruption with high ABI_P among interruptions with identical benefit. This finding resonates with research in behavioral economics that has repeatedly demonstrated that people care more about their own outcome in non-collaborative settings. Our own research generalizes this trend to collaborative settings. It was suprising to us that people cared more about their own task even with a collaborative activity, despite the lack of material benefit. A possible explanation is people's possible lack of trust of computer agents, which has been documented in the behavioral sciences literature (see for example, van Wissen et al. [2012]). Another possible explanation for people valuing their individually assigned tasks

more is the "endowment effect" in which people overestimate the value of objects and tasks that are assigned to them [Kahneman et al., 1990].

The study in Section 6 further investigates these conjectures by applying learning techniques on the data to better understand the correlation of acceptance rate with the cost of interruption to the person. This conjecture is supported by several responses to post-experiment survey questions regarding people's strategies for accepting interruptions. Representative answers include:

> (1) If the agent was in the totally wrong direction and I had several moves left, I would allow the interruption. (2) I always wanted the sure thing for myself. (3) If the collaborator was way off in knowing and had enough moves to likely catch it after I told the location, I accepted. If it compromised my ability to get my goal, I declined.

6 Learning People's Responses

As the work presented in the last section demonstrates, if the benefit of information exchange actions is not clearly positive or negative (a large gain or a loss) then people's behavior diverges from the purely rational behavior as predicted by the NED-DECOP algorithm. In this section, we deploy machine learning to build a model able to predict people's responses to information exchange actions. These learned models can then be used to predict which interruptions are likely to be accepted by people. The NED-DECOP algorithm plays a key role in learning; it computes the value of the features used in the learning.

We compared two machine learning algorithms, naive Bayes and perceptrons. These particular algorithms have been shown to do well for small training sets, and their assumptions are reasonable ones for the interruption game setting. The Naive Bayes algorithm assumes that all features are independent of each other given the class. For example, in the interruption game, given that it is known whether an interruption is accepted, the distance between the principal's position and its goal and the distance between the agent's position and its goal are independent. The perceptron algorithm provides a weighting over the feature set.[7] This allows the algorithm to learn the extent to which people care about such factors as their own benefit and the other player's benefit from interruption requests. After learning, we evaluated the classifiers' predictions as compared to people's actual responses.

Two types of classifiers were used: a *general* classifier which was trained on data from multiple participants; and, a *personalized* classifier which was trained on a particular person's data. Given people's diverse responses in the interruption game, as reported in the last section, we expected that if there were sufficient training data, personalized classifiers would outperform general classifiers when predicting an individual persons behavior in the game.

For each classifier we used two types of features. *Domain-dependent features* represent characteristics of the interruption game. They include the following features: the perceived participant type (whether computer or human); the current round h; the Manhattan distance between the principal's position p_P^h and its goal g_P^h; the Manhattan distance between the agent's position g_A^h and its goal g_A^h; the expected distance between the agent's location and its goal location according to its belief b^h. *Domain-independent features* are general attributes relating to agents' information exchange actions. These include the actual benefit of the interruption (ABI) and

[7] We employed a variant of the perceptron algorithm that uses a soft-margin technique which was been shown to be robust to noise [Khardon and Wachman, 2007].

Model	Accuracy
General NB (domain features)	0.80
General NB (full features)	**0.85**
Personalized NB (domain features)	0.75
Personalized NB (full features)	0.77
Majority Rule	0.71
ABI Rule	0.82

Table 2: Comparison of Naive Bayes classifiers with baseline classifiers

Model	Accuracy
General Perceptron (domain features)	0.81
General Perceptron (full features)	**0.84**
Personalized Perceptron (domain features)	0.75
Personalized Perceptron (full features)	0.77
Majority Rule	0.71
ABI Rule	0.82
General Perceptron (individual benefits)	0.82
Personalized Perceptron (individual benefits)	**0.84**

Table 3: Comparison of Perceptron classifiers with baseline classifiers

its two components: the actual benefit to the principal (ABI_P); and the actual benefit to the agent (ABI_A). The values of these features for a particular game are computed by the NED-DECOP algorithm.

We trained classifiers that used only domain-dependent features, as well as classifiers that used both domain-dependent and domain-independent features. We refer to the latter as using the "full feature set". We studied whether using the domain-independent features improved performance for the classifiers.

Table 2 and Table 3 summarize the performance of the Naive Bayes and Perceptron classifiers respectively. We measured the accuracy of all classifiers in predicting people's responses using ten-fold cross-validation. All reported results were significant using non-parametric paired tests. The tables also show the performance of two baseline strategies: accepting an interruption when the ABI measure is positive (referred as the ABI rule) and choosing the response in the training set that is in the majority (referred as the majority rule).

As shown in the tables, the general classifier using the full feature set performed significantly better than the other classifiers for all measures ($p = 0.012, \alpha = 0.05$). The ABI rule did better than (or as well as) all the other classifiers using domain-dependent features, not taking into account the full general classifiers.

The personalized classifiers were outperformed by both the ABI rule and the general classifiers. A possible reason for this result is the data scarcity for individual subjects; we collected an average of 27 game instances for each subject. It may be difficult to learn a personal perceptron over the full feature set.

Due to the negative results for learning personal classifiers over a large feature set, we focus on learning personal classifiers over a small feature space. Based on results in the social preference literature in Behavioral Economics [Camerer, 2003], we conjectured that people differ in the importance they gave to their value from an interruption request and the importance that they gave to the other participant. To examine this conjecture, we generated general and personal perceptron classifiers trained only on the individual benefit from an interruption request for each participant (ABI_p and ABI_A). The results are presented in Table 3. The personalized classifier trained over ABI_P and ABI_A was able to perform as well as the general perceptron trained over the full feature

Model	Accuracy
ABI Rule	0.82
General Naive Bayes(full features)	0.85
Personal Perceptron (individual utilities)	0.84
Mixture	**0.88**

Table 4: Comparison of the Mixture model with the best performing models.

set.

Lastly, we found that the difference in performance (measured as compared to using the ABI rule) of the best general classifier versus the best personalized classifier was not consistent across subjects. The best general classifier (i.e., the naive Bayesian algorithm with full features) improved classification accuracy for 42% of the subjects in comparison to the ABI rule, whereas the best personal classifier (the perceptron algorithm with individual utilities) improved performance for only 27% of the subjects. These differences suggest trying a hybrid approach or a *mixture model* that chooses whether to apply the personalized or general classifier on a single data point based on their performance on the remaining dataset. We applied leave-one-out cross validation on the dataset. We compare the accuracy of the mixture model with other classifiers in Table 4. As shown by the table, the mixture model was able to perform significantly better than the best Naive Bayes and Perceptron classifiers.

7 Deploying NED-MDPs in agent design

In the previous sections, we showed how NED-MDPs, the NED-DECOP algorithm, and machine learning could be used to build models for information exchange for the interruption game. NED-MDPs, the algorithm and the methodology apply more broadly. We illustrate this generalization to collaborative settings in which people and computers coordinate by providing a brief description of their use for the example from Section 3.1 of a collaboration between a person writing an academic paper, and a computer agent assisting the person by autonomously obtaining bibliographic data. These two participants share the common goal of completing a document, but each of them works largely independently on individual tasks such as composing paragraphs or searching for bibliographical information.

To build a decision-making model for this domain, a designer would need to specify the actions, rewards and transition functions for the agent, the person and their joint efforts. For example, the individual actions of the person include writing the paper, and the coordination actions for the person include supplying keywords to the computer and selecting the appropriate bibliographical information from the available options. The individual actions of the computer include searching for citation information on the web, and the coordination actions for the computer represent when to ask the user for keywords, what keywords to ask for, and choosing which bibliographical options to display to the user. The reward function for the person's task reflects the person's satisfaction from the quality of paragraphs, while the reward for the computer agent's task reflects the agent's success in retrieving the appropriate bibliographical information. The transition functions of the person and the agent represent the dynamic nature of their individual tasks. For example, the user may not know what to write next, and the system may have uncertainty about search results. The transition functions of the person and the agent have a nearly-decomposable structure in that their individual progress is independent of each other's task

and action when they are not exchanging information. On the other hand, a communication action may impede the performance of both participants because the system needs to suspend its search for bibliographical data when it queries the user, and the user may be distracted by the query. Once actions, rewards and transitions are specified, the designer can construct the NED-MDP for the domain.

Using this NED-MDP, the NED-DECOP algorithm can compute baseline strategies for the group (agent and principal). Following these strategies, the computer agent could, for example, request keywords from the user as well as to ask to choose citations from a list. Similarly, the algorithm can be used to compute the benefit of coordination actions to the group from the person's perspective (e.g., ABI) and individual components of this benefit to the group participants (e.g., ABI_P and ABI_A).

To complete the agent design, a designer would undertake empirical studies in which the agent used the baseline strategies to coordinate with a representative sample of users, and people's responses to these requests would be used to train classifiers to distinguish interruptions that are likely to be accepted by the people. The domain independent features for the classifiers would include the benefits of a coordination action for individuals and for the group as computed from the person's perspective (e.g., ABI_P, ABI_A and ABI), just as in the interruption game. The domain dependent features might include such characteristics as the length of the text, the number of citation keywords. The resulting learned model provides the designer with a system component that can be used to identify useful opportunities for information exchange.

8 Conclusion and Future Work

This paper demonstrates that decomposition, which has been shown repeatedly to be important for reducing the complexity of AI reasoning systems [Simon, 1962], can play a significant role in overcoming the general intractability of collaborative decision-making in the context of information-exchange. It introduces the concept of a nearly decomposable decision-making problem and demonstrates that decisions about information exchange between people and computer systems can be modeled in this way when their collaborative activity is composed largely of individual tasks with a need to coordinate only occasionally. Although information exchange is not frequent, it remains crucial for their joint performance on the shared activity. Even though it is infrequent, reasoning about the utility of information exchange actions on the collaboration may be intractable with general approaches.

The paper defines NED-MDPs, a novel model for formalizing information exchange, which can be used to represent the nearly decomposable structure of these decision-making problems. It presents a novel algorithm, which exploits the special structure of NED-MDPs, for efficiently computing agents' strategies for information exchange. The efficacy of the model and algorithm are demonstrated on an interruption management problem, a special case of an information exchange task. The paper also introduces a new methodology for generating coordination strategies for agents, one which proceeds from a fully rational baseline model of decision-making to comparing the behavior this model engenders with people's responses to it and then, using machine learning, constructs predictive models of their responses.

These results may be extended in several ways: First, the algorithm model can be extended to account for the discrepancy in the information that agents have about the world. To address the added complexity implied by this extension, sampling methods like Monte-Carlo tree search might be used to improve the efficiency of the algorithm. Such techniques also allow the algorithm to be applied to a broader range of collaborative set-

tings [Kocsis and Szepesvári, 2006]. Agents can be designed based on iterative application of the methodology; such agents would revise their coordination strategies based on their interactions with people. These agents could be developed not only for the interruption game, but also for other collaborative settings involving information exchange. The techniques and approaches presented in this paper suggest that analyzing problems to find natural places for problem decomposition might be used to reduce complexity and lead to agents better able to coordinate with people in a range of decision-making domains.

9 Appendix

Proof of Thoerem 1 (by induction).

Proof. ***Basis:*** At $h = H$, the end of the time horizon is reached. We apply the original Dec-MDP value function given in Equation 1:

$$\begin{aligned} V^{\pi}(s_1^H, s_2^H) &= R(s_1^H, s_2^H) \\ &\textit{We apply the reward independence property presented in Equation 3 :} \\ &= R_1(s_1^H) + R_2(s_2^H) \\ &= V_1^{\pi}(s_1^H, s_2^H) + V_2^{\pi}(s_1^H, s_2^H) \end{aligned}$$

Inductive step: For each $h+1 < H$, we assume that:

$$V^{\pi}(s_1^{h+1}, s_2^{h+1}) = V_1^{\pi}(s_1^{h+1}, s_2^{h+1}) + V_2^{\pi}(s_1^{h+1}, s_2^{h+1}) \tag{28}$$

We generate the value function for time step h by applying Equation 1 from Section 3.1:

$$\begin{aligned} V^{\pi}(s_1^h, s_2^h) &= R(s_1^h, s_2^h) + \sum_{(s_1^{h+1}, s_2^{h+1})} T\big((s_1^{h+1}, s_2^{h+1}) \mid (s_1^h, s_2^h), \pi(s_1^h, s_2^h)\big) \\ &\cdot V^{\pi}(s_1^{h+1}, s_2^{h+1}) \\ &\textit{We apply the semi-transition and reward independence properties of the NED-MDP} \\ &\textit{formalization given in Equations 2 and 3:} \\ &= R_1(s_1^h) + R_2(s_2^h) + \sum_{(s_1^{h+1}, s_2^{h+1})} T_1\big(s_1^{h+1} \mid s_1^h, \pi(s_1^h, s_2^h)\big) \cdot T_2\big(s_2^{h+1} \mid s_2^h, \pi(s_1^h, s_2^h)\big) \\ &\cdot (V_1^{\pi}(s_1^{h+1}, s_2^{h+1}) + V_2^{\pi}(s_1^{h+1}, s_2^{h+1})) \\ &= R_1(s_1^h) + \sum_{(s_1^{h+1}, s_2^{h+1})} T_1\big(s_1^{h+1} \mid s_1^h, \pi(s_1^h, s_2^h)\big) \cdot T_2\big(s_2^{h+1} \mid s_2^h, \pi(s_1^h, s_2^h)\big) \cdot V_1^{\pi}(s_1^{h+1}, s_2^{h+1}) \\ &+ R_2(s_2^h) + \sum_{(s_1^{h+1}, s_2^{h+1})} T_1\big(s_1^{h+1} \mid s_1^h, \pi(s_1^h, s_2^h)\big) \cdot T_2\big(s_2^{h+1} \mid s_2^h, \pi(s_1^h, s_2^h)\big) \cdot V_2^{\pi}(s_1^{h+1}, s_2^{h+1}) \\ &= V_1^{\pi}(s_1^h, s_2^h) + V_2^{\pi}(s_1^h, s_2^h) \end{aligned} \tag{29}$$

The derivation of the value function (Equation 6) is straightforward given that the value of the joint policy is aggregate of the values of individual policies, and that the transition function is semi-independent. □

10 Acknowledgments

The research reported in this paper was supported in part by grant IIS-0705406 from the U.S. National Science Foundation, Marie Curie grant 268362 and grant 1276/12 from the Israeli Science Foundation.

References

C. Amato, A. Carlin, and S. Zilberstein. Bounded dynamic programming for decentralized POMDPs. In *AAMAS 2007 workshop on multi-agent sequential decision making in uncertain domains*, 2007.

E. Arroyo and T. Selker. Self-adaptive multimodal-interruption interfaces. In *Proceedings International Conference on Intelligent User Interfaces*, pages 6–11, 2003.

D. Avrahami, J. Fogarty, and S.E. Hudson. Biases in human estimation of interruptibility: Effects and implications for practice. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, page 60. ACM, 2007.

T. Babaian, B.J. Grosz, and S.M. Shieber. A writer's collaborative assistant. In *Proceedings of the 7th International Conference on Intelligent User Interfaces*, pages 7–14. ACM New York, NY, USA, 2002.

M. Bazerman. *Judgment in Managerial Decision Making*. Wiley Publishers, 2001.

R. Becker, S. Zilberstein, V. Lesser, and C.V. Goldman. Transition-independent decentralized Markov decision processes. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 41–48. ACM New York, NY, USA, 2003.

R. Becker, S. Zilberstein, and V. Lesser. Decentralized markov decision processes with event-driven interactions. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 302–309, 2004.

R. Becker, V. Lesser, and S. Zilberstein. Analyzing myopic approaches for multi-agent communication. In *IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, pages 550–557, 2005.

D.S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein. The complexity of decentralized control of Markov decision processes. *Mathematics of Operations Research*, pages 819–840, 2002.

A. Beynier and A.I. Mouaddib. A polynomial algorithm for decentralized Markov decision processes with temporal constraints. In *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 963–969. ACM New York, NY, USA, 2005.

C. Boutilier. Sequential optimality and coordination in multiagent systems. In *International Joint Conference on Artificial Intelligence*, volume 16, pages 478–485, 1999a.

C. Boutilier. Multiagent systems: Challenges and opportunities for decision-theoretic planning. *AI magazine*, 20(4):35, 1999b.

M.E. Bratman. Shared cooperative activity. *The Philosophical Review*, 101(2):327–341, 1992.

C. F. Camerer. *Behavioral Game Theory. Experiments in Strategic Interaction*, chapter 2. Princeton University Press, 2003.

W.J. Conover. *Practical nonparametric statistics*. Wiley New York, 1999.

M. Czerwinski, E. Cutrell, and E. Horvitz. Instant messaging and interruption: Influence of task type on performance. In *Proceedings of OZCHI*, pages 356–361, 2000.

L. Dabbish and R. Kraut. Awareness displays and interruptions in teams. In *Annual Meeting of the Academy of Management, Seattle*, 2003.

A. Falk and U. Fischbacher. A theory of reciprocity. *Games and Economic Behvior*, 54(2):293–315, 2006.

J. Fogarty, S.E. Hudson, C.G. Atkeson, D. Avrahami, J. Forlizzi, S. Kiesler, J.C. Lee, and J. Yang. Predicting human interruptibility with sensors. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 12(1): 119–146, 2005.

T. Gillie and D. Broadbent. What makes interruptions disruptive? A study of length, similarity, and complexity. *Psychological Research*, 50(4):243–250, 1989.

C.V. Goldman and S. Zilberstein. Communication-based decomposition mechanisms for decentralized MDPs. *Journal of Artificial Intelligence Research*, 32:169–202, 2008.

B.J. Grosz and S. Kraus. Collaborative plans for complex group action. *Artificial Intelligence*, 86(2):269–357, 1996.

E.A. Hansen, D.S. Bernstein, and S. Zilberstein. Dynamic programming for partially observable stochastic games. In *Proceedings of the National Conference on Artificial Intelligence*, pages 709–715. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2004.

H.M. Hodgetts and D.M. Jones. Interrupting problem solving: Effects of interruption position and complexity. In *Past Reflections, Future Directions: Proceedings of the 40th Australian Psychological Society Annual Conference, Melbourne, Australia*, pages 128–132, 2005.

E. Horvitz and J. Apacible. Learning and reasoning about interruption. In *Proceedings of the 5th International Conference on Multimodal Interfaces*, pages 20–27. ACM New York, NY, USA, 2003.

E. Horvitz, A. Jacobs, and D. Hovel. Attention-sensitive alerting. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, volume 99, pages 305–313, 1999.

E. Horvitz, P. Koch, C.M. Kadie, and A. Jacobs. Coordinate: Probabilistic forecasting of presence and availability. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 2002.

S. Hudson, J. Fogarty, C. Atkeson, D. Avrahami, J. Forlizzi, S. Kiesler, J. Lee, and J. Yang. Predicting human interruptibility with sensors: A wizard of Oz feasibility study. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 257–264. ACM New York, NY, USA, 2003.

S.T. Iqbal and B.P. Bailey. Leveraging characteristics of task structure to predict the cost of interruption. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, page 750. ACM, 2006.

D. Kahneman, J.L. Knetsch, and R.H. Thaler. Experimental tests of the endowment effect and the coase theorem. *Journal of political Economy*, pages 1325–1348, 1990.

E. Kamar and B.J. Grosz. Applying MDP approaches for estimating outcome of interaction in collaborative human-computer settings. *Multi-Agent Sequential Decision Making in Uncertain Domains*, pages 25–32, 2007.

E. Kamar, E. Horvitz, and C. Meek. Mobile opportunistic commerce: mechanisms, architecture, and application. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems*, 2008.

E. Kamar, Y. Gal, and B.J. Grosz. Modeling User Perception of Interaction Opportunities for Effective Teamwork. In *Proceedings of the 2009 International Conference on Computational Science and Engineering-Volume 04*, pages 271–277. IEEE Computer Society, 2009.

R. Khardon and G. Wachman. Noise tolerant variants of the perceptron algorithm. *The Journal of Machine Learning Research*, 8:227–248, 2007.

L. Kocsis and C. Szepesvári. Bandit based monte-carlo planning. *Machine Learning: ECML 2006*, pages 282–293, 2006.

R.E. Kraut and P. Attewell. Media use in a global corporation: Electronic mail and organizational knowledge. *Culture of the Internet*, pages 323–342, 1997.

S. LeeTiernan, E. Cutrell, M. Czerwinski, and H. Hoffman. Effective notification systems depend on user trust. In *Proceedings of Human-Computer Interaction–Interact*, 2001.

D.S. McCrickard, R. Catrambone, CM Chewar, and J.T. Stasko. Establishing tradeoffs that leverage attention for utility: Empirically evaluating information display in notification systems. *International Journal of Human-Computer Studies*, 58(5):547–582, 2003.

D. McFarlane. Comparison of four primary methods for coordinating the interruption of people in human-computer interaction. *Human-Computer Interaction*, 17(1):63–139, 2002.

F.S. Melo and M. Veloso. Decentralized mdps with sparse interactions. *Artificial Intelligence*, 175(11):1757–1789, 2011.

F.S. Melo, M.T.J. Spaan, and S. Witwicki. Exploiting sparse interactions for optimizing communication in dec-mdps. In *Seventh Annual Workshop on Multiagent Sequential Decision Making Under Uncertainty (MSDM-2012)*, page 40, 2012.

H. Mostafa and V. Lesser. Offline planning for communication by exploiting structured interactions in decentralized mdps. In *Web Intelligence and Intelligent Agent Technologies, 2009. WI-IAT'09. IEEE/WIC/ACM International Joint Conferences on*, volume 2, pages 193–200, 2009.

R. Nair, P. Varakantham, M. Tambe, and M. Yokoo. Networked distributed pomdps: a synthesis of distributed constraint optimization and pomdps. In *Proceedings of the 20th national conference on Artificial intelligence-Volume 1*, pages 133–139. AAAI Press, 2005.

B. O'Conaill and D. Frohlich. Timespace in the workplace: Dealing with interruptions. In *Conference on Human Factors in Computing Systems*, pages 262–263. ACM New York, NY, USA, 1995.

F.A. Oliehoek. Decentralized POMDPs. In *Reinforcement Learning: State of the Art, Adaptation, Learning, and Optimization*. Springer Berlin Heidelberg, 2012.

F.A. Oliehoek, M.T.J. Spaan, and N. Vlassis. Optimal and approximate q-value functions for decentralized POMDPs. *Journal of Artificial Intelligence Research*, 32(1):289–353, 2008a.

F.A. Oliehoek, M.T.J. Spaan, S. Whiteson, and N. Vlassis. Exploiting locality of interaction in factored Dec-POMDPs. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 517–524. International Foundation for Autonomous Agents and Multiagent Systems Richland, SC, 2008b.

C.H. Papadimitriou and J.N. Tsitsiklis. The complexity of Markov decision processes. *Mathematics of Operations Research*, pages 441–450, 1987.

D.V. Pynadath and M. Tambe. The communicative multiagent team decision problem: Analyzing teamwork theories and models. *Journal of Artificial Intelligence Research*, 16:389–423, 2002a.

D.V. Pynadath and M. Tambe. Multiagent teamwork: Analyzing the optimality and complexity of key theories and models. In *Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 2*, pages 873–880, 2002b.

Z. Rabinovich, C.V. Goldman, and J.S. Rosenschein. The complexity of multiagent systems: The price of silence. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1102–1103. ACM New York, NY, USA, 2003.

K. Renaud. Expediting rapid recovery from interruptions by providing a visualization of application activity. In *Proceedings of OZCHI*, pages 348–355, 2000.

J.K. Rilling, A.G. Sanfey, J.A. Aronson, L.E. Nystrom, and J.D. Cohen. The neural correlates of theory of mind within interpersonal interactions. *Neuroimage*, 22(4):1694–1703, 2004.

M. Roth, R. Simmons, and M. Veloso. Reasoning about joint beliefs for execution-time communication decisions. In *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 786–793. ACM New York, NY, USA, 2005.

M. Roth, R. Simmons, and M. Veloso. What to communicate? Execution-time decision in multi-agent POMDPs. In *The 8th International Symposium on Distributed Autonomous Robotic Systems*. Springer, 2006.

M. Roth, R. Simmons, and M. Veloso. Exploiting factored representations for decentralized execution in multi-agent teams. In *Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems*. ACM New York, NY, USA, 2007.

P. Rudman and M. Zajicek. Autonomous agent as helper-helpful or annoying? In *Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, pages 170–176. IEEE Computer Society, 2006.

P. Scerri, D. Pynadath, L. Johnson, P. Rosenbloom, M. Si, N. Schurr, and M. Tambe. A prototype infrastructure for distributed robot-agent-person teams. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 433–440. ACM New York, NY, USA, 2003.

S. Seuken and S. Zilberstein. Memory-bounded dynamic programming for DEC-POMDPs. In *International Joint Conference on Artificial Intelligence*, pages 2009–2016, 2007.

S. Seuken and S. Zilberstein. Formal models and algorithms for decentralized decision making under uncertainty. *Autonomous Agents and Multi-Agent Systems*, 17(2):190–250, 2008.

H.A. Simon. The architecture of complexity. *Proceedings of the American Philosophical Society*, 106(6): 467–482, 1962.

D. Szer, F. Charpillet, S. Zilberstein, et al. MAA*: A heuristic search algorithm for solving decentralized POMDPs. In *Proceedings of the twenty-first conference on uncertainty in artificial intelligence*, pages 576–583, 2005.

A. van Wissen, Y. Gal, B.A Kamphorst, and M.V Dignum. Human–agent teamwork in dynamic environments. *Computers in Human Behavior*, 28:23–33, 2012.

P. Varakantham, J. Kwak, M. Taylor, J. Marecki, P. Scerri, and M. Tambe. Exploiting coordination locales in distributed POMDPs via social model shaping. *Proc. of ICAPS*, 2009.

S. Voida and E.D. Mynatt. It feels better than filing: Everyday work experiences in an activity-based computing system. In *Proceedings of the 27th International Conference on Human Factors in Computing Systems*, pages 259–268. ACM, 2009.

S.J. Witwicki and E.H. Durfee. Influence-based policy abstraction for weakly-coupled dec-pomdps. In *Proceedings of the Twentieth International Conference on Automated Planning and Scheduling (ICAPS)*, 2010.

P. Xuan, V. Lesser, and S. Zilberstein. Communication decisions in multi-agent cooperation: Model and experiments. In *Proceedings of the Fifth International Conference on Autonomous Agents*, pages 616–623. ACM New York, NY, USA, 2001.