8-2010

Effect of human biases on human-agent teams

Praveen PARUCHURI
Carnegie Mellon University

Pradeep Reddy VARAKANTHAM
Singapore Management University, pradeepv@smu.edu.sg

Paul Scerri
Carnegie Mellon University

DOI: https://doi.org/10.1109/WI-IAT.2010.104

Effect of human biases on human-agent teams

Praveen Paruchuri, Pradeep Varakantham*, Katia Sycara, Paul Scerri
Robotics Institute, Carnegie Mellon University
paruchur,katia,pscerri@cs.cmu.edu
*School of Information Systems, Singapore Management University
pradeepv@smu.ed.sg

***Abstract*—As human-agent teams get increasingly deployed in the real-world, agent designers need to take into account that humans and agents have different abilities to specify preferences. In this paper, we focus on how human biases in specifying preferences for resources impacts the performance of large, heterogeneous teams. In particular, we model the inclination of humans to simplify their preference functions and to exaggerate their utility for desired resources, and show the effect of these biases on the team performance. We demonstrate this on two different problems, which are representative of many resource allocation problems addressed in literature. In both these problems, the agents and humans optimize their constraints in a distributed manner. This paper makes two key contributions: (a) Proves theoretical properties of the algorithm used (named DSA) for solving distributed constraint optimization problems, which ensures robustness against human biases; and (b) Empirically illustrates that the effect of human biases on team performance for different problem settings and for varying team sizes is not significant. Both our theoretical and empirical studies support the fact that the solutions provided by DSA for mid to large sized teams are very robust to the common types of human biases.**

I. Introduction

A range of exciting proposed applications will involve many humans and agents working along side each other to achieve a complex objective. Domains for such applications include search and rescue [5], disaster response [12], military applications [1] and commerce [3]. Researchers envision automating the allocation of shared resources using algorithms such as distributed constraint optimization algorithms (DCOPs) [7]. For example, access to satellites, robots, computation or space may be automatically assigned using DCOPs. Due to the computational load and communication intensity of these algorithms, humans in the team will need to communicate their preferences and utilities to a proxy that executes the algorithm on their behalf. If there are many resources these preferences may be communicated incompletely or approximately. However, agents participating in the allocation process will be able to precisely and completely specify their preferences for all resources. The question addressed by this paper is what happens to the quality of the overall resource allocation when human and agent preference specifications differ in this way.

Preference elicitation is known to be a difficult problem that takes a lot of time and effort for humans [2]. When the preferences correspond to utilities, e.g., the value to the team to assign a particular robot to a particular human, a considerable computational burden is placed on the human since the utility of many resources must be computed by planning for each set of resources they might get and compare against other possible resource allocations. Many well known human biases will come into play, either consciously or sub-consciously, when reporting the utility of a resource. In this paper, we use a combination of empirical and theoretical analysis to understand the impact of these biases using a DCOP algorithm.

Using two commonly known models of human biases [4], we performed an extensive empirical evaluation of overall utility for a generic resource allocation problem and a generic event scheduling problems. These are benchmark problems that have been well-studied in literature [13], [6]. The experiments measured the difference between the utility of the solution for accurate preference values and the utility of the same solution with biased preference values. We have employed the Distributed Stochastic Algorithm (DSA) [14] in this paper, because it is the only algorithm that scales to the problems of interest [11] in this paper.

A key observation from our experiments was that even for extreme biases, there was a minor change in the utility on average. While the actual solution changed substantially, the utility of the changed solution was slightly less than the original utility. There was no consistent advantage or disadvantage to team members that had biased utilities, suggesting that humans would neither gain nor lose by not describing their preferences for each resource precisely. In addition, we were able to prove that the solution provided by DSA is dependent on the local ranking of preferences and not on the absolute preference values of the agent. This makes DSA a particularly attractive option for use in human-agent systems, because it removes the need for humans to accurately calibrate their utility estimates with the overall team.

In an attempt to understand why the human biases led to relatively minor impact on the overall utility, we looked more closely at the resource allocation and distributed event scheduling problems. We observed that for large problems, involving 100 or more agents, many agents were allocated one of their top few preferred resources/events. This means that provided the humans specify their top options, there is no significant impact on the overall utility. Moreover, these larger problems had many solutions that were of similar overall utility, and thus, inaccurate specification of preferences simply pushed the solution between generally good ones rather than towards bad ones.

II. Problems and Models

In this section, we provide a brief background of the motivating problems, models used to represent the problems and the approach employed to solve the models. Specifically, we describe two generic distributed resource allocation problems from literature [13], [6], [8] and then explain the DCOP framework used to represent these problems. We then formally represent the biases introduced by humans in specifying the preference function within the utility function of DCOP and finally present the DSA algorithm which is used to solve the DCOP models. In both these domains, the constraint functions represent both hard constraints (avoid scheduling one meeting in two different time slots etc.) and soft constraints (prefer to meet in mornings rather than afternoons etc).

A. Problem 1: Discrete Resource Allocation

Resource allocation problems [13] represent bipartite graph matching which is representative of an entire class of matching problems, specifically ones in coding theory, task to capability matching and others. In this domain, resources need to be allocated to a group of agents and humans, E based on their preferences. For ease of explanation, we assume only one resource is allocated to one agent or human (Extension to a multi-resource problem involves enumerating all the possible valid resource combinations that can be allocated to each agent). We model this domain as a DCOP as follows:

Each agent/human has a variable, represented as $e \in E$. The values that the variable can take correspond to the resource allocated to the agent. This domain for variable e is specified as D_e and belongs to the set $\{0, 1, 2, \cdots, R\}$. The utility for e when a resource, $r_e \in D_e$, is allocated to e is

$$u_e(r_e) = L * I_{r_e \neq 0} * (1 - \prod_{\tilde{e} \in E, \tilde{e} \neq e} I_{r_e \neq r_{\tilde{e}}}) + p_e(r_e), \quad (1)$$

where $p_e(r_e)$ is the preference value of agent e for resource r (soft constraint) and equals 0 when r_e is 0. L is a large negative number that represents the penalty for allocating one resource to two different entities (hard constraint) and

$$\begin{aligned} I_{condition} &= 1, if condition = true \\ &= 0, otherwise \end{aligned}$$

B. Problem 2: Distributed Event Scheduling

Distributed event scheduling introduced by [6] captures problems such as meeting scheduling, sensor scheduling in sensor networks and other distributed scheduling domains. Here, we explain this problem using meeting scheduling, however, it extends directly to other distributed event scheduling domains. M meetings need to be scheduled for humans and agents (acting on behalf of humans). A meeting, m can require multiple humans, E_m and a human could be part of multiple meetings, M_e. Furthermore, each human/agent, e has preference values over the time slots, $p_e(t)$ and meetings, $p_e(m)$. For a variable, v, these preferences are specified as $p_e^v(t)$ and $p_e^v(m)$. Given this information, the goal is to compute a schedule which maximizes the utility of the team. This domain is modeled as a DCOP by [6] as follows:

(a) Each agent/human, e has multiple variables, M_e, corresponding to all the meetings where e is required.

(b) The values that each variable can take, correspond to any of the time slots where the meeting can be scheduled. This domain for variable $m_e \in M_e$ is specified as D_m^e and belongs to the set $\{0, 1, 2, \cdots, T\}$, where T is the set of time slots available in which a meeting can be scheduled.

(c) If a meeting, m for e is scheduled at $t_e^m \in D_m^e$, then the utility for e is defined as

$$\begin{aligned} u_e^m(t_e^m) = L * (1 - &\prod_{\tilde{m} \in M_e, \tilde{m} \neq m, t_e^m \neq 0, t_e^{\tilde{m}} \neq 0} I_{t_e^{\tilde{m}} \neq t_e^m}) * \\ &(1 - \prod_{\tilde{e} \in E_m, e \neq \tilde{e}} I_{t_e^m = t_{\tilde{e}}^m}) + (p_e^m(m) + p_e^m(t_e^m)) \quad (2) \end{aligned}$$

Here, L is a large negative number that represents a penalty for scheduling the two meetings at the same time slot or for not scheduling a meeting at the same time slot for all participants. $(p_e^m(m) + p_e^m(t_e^m))$ represents the agent/human preferences and the sum equals zero when t_e^m is zero.

C. Distributed Constraint Optimization

DCOP [7] is a popular framework for representing coordination in multiagent systems. A DCOP consists of n variables, $v_1, v_2, ..., v_n$. Variable v_i can take on any value from the discrete finite domain D_i. The goal is to find an assignment, A of values to variables, such that the sum over a set of binary constraints and associated payoff or utility functions, $u_{i,j} : D_i \times D_j \rightarrow N$, is maximized. Formally, we maximize $\sum_{v_i, v_j} u_{i,j}(d_i, d_j)$, where $d_i \in D_i, d_j \in D_j$ and $\{v_i \leftarrow d_i, v_j \leftarrow d_j\} \in A$.

D. Bias 1: Simplification of preferences

As shown in [4], humans tend to simplify preference values when faced with problems where multiple factors need to be considered. One popular way of simplifying preferences is thresholding. For example in meeting scheduling, it is natural for a human to specify his preferences as 1:00-5:00 are "good" and rest of the times are "bad", instead of specifying a different preference value for each time slot even if they do actually have different preferences. Formally, this involves approximating the preference function over a variable v, $p_e^v()$ as a function that has zero corresponding to all but the top "k" values in its range. In general, it could be a "multi-step" step function and defined as follows (with ordering of thresholds given as $thres_1 > thres_2 > \cdots$):

$$\begin{aligned} \tilde{p_e^v}(d) &= \max_{\hat{d}} p_e^v(\hat{d}), \quad if \quad p_e^v(d) > thres_1 \\ &= thres_1, \quad if \quad thres_2 \leq p_e^v(d) \leq thres_1 \\ &= \cdots\cdots \end{aligned}$$

Instead of a threshold, humans may simply limit themselves to specifying some number of top preferences. Specifically, we consider the scenario where humans specify preferences for only the most important resources while ignoring the rest. The modified function for this preference simplification is defined as:

$$\begin{aligned} \tilde{p_e^v}(d) &= p_e^v(d), \text{ if } d \in maxK(p_e^v) \\ &= 0, \text{ otherwise} \end{aligned}$$

E. Bias 2: Preference Exaggeration

This bias of humans arises due to exaggeration of the importance of certain features over others. To be precise, we represent cases where humans subconsciously exaggerate their preferences. For example it would be very natural for a person to believe her/his task was critical to overall team success and therefore set the utility of them getting access to the best resources to be high. This involves increasing the preference function value of a variable v, $p_e^v()$ for the top preferred value assignments. The modified function for this type of preference exaggeration is defined as:

$\tilde{p_e^v}(d) = \mathcal{S} * p_e^v(d) + \mathcal{A}$, if $d \in maxK(p_e^v())$
$= 0$, otherwise

$maxK(p_e^v())$ provides the set containing the top k values in the range of the function $p_e^v()$, $\mathcal{S}$ and $\mathcal{A}$ are scaling and addition factors of exaggeration.

Another variant of this bias is where the humans exaggerate the preference for values which are not part of the $maxK$ set. This exaggeration in bias of unimportant meetings/resources could be arise due to selfish manipulation or lack of information about the problem domain. We model this by adding a random Gaussian noise to the original preference values (X represents the Gaussian noise).

$\tilde{p_e^v}(d) = \mathcal{S} * p_e^v(d) + \mathcal{A}$, if $d \in maxK(p_e^v() + X)$
$= 0$, otherwise

III. Algorithm for solving DCOPs

Distributed Stochastic Algorithm (DSA) is the algorithm that we employ for solving DCOPs. It should be noted that DSA is the only algorithm that can realistically scale to the type of problems considered in this paper. There are three key steps to DSA: (a) Every variable starts with a random assignment, d; (b) Each variable calculates its current local utility and the overall best utility (over all possible assignments) given the current assignments of the neighboring agents. It then computes the gain for moving to the best assignment as the difference of best utility and the current local utility. (c) If gain greater than 0, the variable updates to the best assignment with a probability p.

A. Key properties of DSA

In this section, we prove a key property of DSA for the type of problems introduced earlier (Problems 1 & 2). For ease of explanation in the proof, we provide the general structure of utility functions for the DCOP variables given the types of DCOPs mentioned in the Problems and Models section:

$$u_e^v(d) = L * I_{\neg concur} + p_e^v(d), \tag{3}$$
$$\text{where } L << -1 * \max_{d_1} p_e^v(d_1)$$

Proposition 1. *For DCOP problems 1 and 2, the solution provided by DSA is reliant on ranking of preference values and not on the actual preference values.*

Proof. To prove the proposition, we create a new preference function for all variables v, $\hat{p}_e^v()$ by modifying the original preference function, $p_e^v()$, i.e. $p_e^v(d_1)$ is modified to $\hat{p}_e^v(d_1)$, $p_e^v(d_2)$ to $\hat{p}_e^v(d_2)$ and so on. Ordering of preference values in the new preference function, $\hat{p}_e^v()$, is preserved by using the following constraint.

$$\begin{aligned}\forall d_1, d_2 \in D_e^v, \hat{p}_e^v(d_1) &> \hat{p}_e^v(d_2), if \quad p_e^v(d_1) > p_e^v(d_2)\\ &< \hat{p}_e^v(d_2), if \quad p_e^v(d_1) < p_e^v(d_2)\\ &= \hat{p}_e^v(d_2), if \quad p_e^v(d_1) = p_e^v(d_2)\end{aligned} \tag{4}$$

Now, we show that the solution provided by DSA will be the same for both preference functions, given identical initial random seed. For this, we show that in every update phase of DSA, the updates made at each variable are the same.

Since we have identical initial random seeds, the neighbors for agent e in both cases (original and modified preferences) initially will be the same. Without loss of generality, we assume that the preference value ordering for a variable v is $p_e^v(d_1) \geq p_e^v(d_2) \geq \ldots \geq p_e^v(d_l)$. There are two cases of importance for each variable during its assignment update phase (step (c) of algorithm) :

(a) Firstly, we consider the case where current value assignment, d at agent e is not in agreement with the neighboring agents, i.e. the first part of Equation 3 is equal to L. For instance, in task allocation, this happens if the same resource is allocated to two different agents. Similarly, this happens in meeting scheduling, if two meetings are scheduled at the same time or a meeting is scheduled in two different time slots for two humans. Since L is a large negative number and all preferences values (both original, $p_e^v(d)$ and modified, $\hat{p}_e^v(d)$) are greater than or equal to zero (zero for not scheduling a meeting or allocating a resource), the gain is less than zero in both cases. Thus in both cases, the assignment for variable v of e will need to be be changed to a value which is in concurrence with the neighbors (i.e. first part of utility equations is equal to zero) and has the highest preference value amongst all values which are in concurrence with their neighbors. We define this sub set of values that are in concurrence with neighbors as

$CD_e^v = \{d | u_e^v(d) \geq 0\}$ and $\mathcal{CD}_e^v = \{\hat{d} | \hat{u}_e^v(\hat{d}) \geq 0\}$.

Since the variable assignments are the same for both cases before the value update phase, $CD_e^v = \mathcal{CD}_e^v$. Given this, the value d to which variable v should be updated in the DCOP with original preference function is given by:

$d = \arg\max_{\tilde{d} \in CD_e^v} u_e^v(\tilde{d})$

Since $u_e^v(\hat{d}) >= 0$, from Equation 1, $d = \arg\max_{\tilde{d} \in CD_e^v} p_e^v(\tilde{d})$

From Equation 4, $d = \arg\max_{\tilde{d} \in \mathcal{CD}_e^v} \hat{p}_e^v(\tilde{d})$

Therefore, $d = \hat{d}$, where $\hat{d}$ is the value to which variable v should be updated in the DCOP with modified preference function.

(b) Secondly, we consider the case where current assignment, d at agent e is in agreement with neighboring agents, i.e. the first part of the utility equation is equal to 0. Therefore, the new assignment should be the one with the highest p_e^v or $\hat{p}_e^v$. As we saw in (a) above, this leads to both cases (original and modified DCOPs) having the same assignment.

Therefore, after the first assignment update phase (step (c)), the new assignments are identical for the original and modified DCOP problems. By continuing the above analysis until convergence, the solutions provided by DSA remain similar for both DCOPs. ■

Although we are unable to prove analytically, we illustrate experimentally in the future sections that DSA is robust even to cases where there are disruptions to the preference order.

IV. Impact of Approximations

In this section, we examine the impact of the human biases introduced earlier as biases 1 and 2. Firstly, in domains where the ordering of preference values remains the same as the original preferences after human biases, according to Proposition 1 the impact is zero. Since, we have already proved this theoretically, we will not provide further empirical results to illustrate this. Our empirical analysis will be primarily focused on showing that even if there are disruptions to the preference order (due to human biases), DSA provides solutions that are close to the optimal solution obtained with the original preference function. All our experiments were performed on the generic problems of distributed resource allocation and event scheduling that were described earlier.

Subscript	Description
Hch	Humans CHanging preferences
Bch	Both humans and agents CHanging preferences
$EHch$	Enhanced Hch where zero valued preferences are considered as valid.
$EBch$	Enhanced Bch where zero valued preferences are considered as valid.

TABLE I
DESCRIPTION OF THE PREFERENCE APPROXIMATIONS

A. Preference Approximations and Algorithms

We considered four different types of preference approximations in our experiments, described briefly in Table I: Hch represents the problem where humans change their preferences using biases 1 and 2 (explained earlier) and Bch represents the case where both humans and agents change their preferences using biases 1 and 2. The third approximation, EHch enhances Hch, where zero valued preferences (agent gets a reward of zero for that preference) are considered valid and would account for cases where preferences for a resource/time/event were decreased to zero from a positive value. Similarly, the fourth approximation, EBch enhances Bch.

With respect to algorithms, we provide comparisons between the Global optimal algorithm (represented using G) and the DSA algorithm (represented using DSA) on the various categories of preference approximations. For ease of explanation, we will henceforth represent the preference approximation as a subscript of the algorithm. For instance, solving the Hch preference approximation using G (global optimal algorithm) is represented as $GHch$.

B. Simulation setup

We now describe the experimental setup for our problem domains. We borrowed problem settings from the literature [1] and extended them to larger problems of upto 500 agents and hundreds of resources (meetings). It should be noted that it is not possible to provide results over the entire domain space of the generic problems presented earlier. Therefore, we experimented with many different randomly sampled team sizes along with many different constraints. Furthermore, to eliminate the dependence of results on the DCOP constraint values, we averaged over 500 randomly generated problems (using Uniform and Gaussian distributions). Although our results may not be guaranteed to hold over the entire domain space of problems, they identify a very interesting pattern observed in many commonly used problems.

Table II summarizes the simulation setup for the discrete resource allocation domain. The first column in the table lists the variables involved: (a) Number of agents ($|E|$); (b) Number of resources (R); (c) Number of top choices reported by the human (corresponds to Bias 1, expressed as $maxK$); and (d) Exaggeration factor (corresponds to Bias 2, expressed using S, A and X).

For each setting of the number of agents, the number of resources range from a low value to a high value (ex: resource range <4,20>). Bias 1 is represented by the number of top choices reported by the human and is varied from 0 to 3. Bias 2 is expressed using the exaggeration factors $\{S, A, X\}$. S was one of 1, 5 or 10, A was set to 0 in all our experiments while X was set to 0 or 25%. We bound the effect of the exaggeration factor by setting a threshold on the amount of utility an agent can claim for each resource. The total number of settings for this simulation is 96 and each setting was tested on 500 randomly generated examples leading to a total of 48000 example problems for each combination of algorithm and preference approximation type (total 10 combinations, 2 algorithms and 5 approximation types). In both this and the next domain, the percentage of humans in the team is 50%.

Similarly, Table III summarizes the simulation setup details for the distributed event scheduling domain. The variables involved in this domain are the domain size (Number of agents/humans E, meetings M, maximum number of agents per meeting E_m, number of meetings per agent M_e and time slots t), the number of top choices reported (Bias 1 captured using $maxK$) and the exaggeration factor (Bias 2 expressed using the terms S, A and X). In total, there were 60 different settings with 500 randomly generated examples tested for each setting leading to a total of 30000 tested examples for each combination of algorithm and preference approximation. Computer simulation time of nearly 90 hours was involved in running all the experiments.

For both these problem domains, we experimented with both Uniform and Gaussian distributions to generate preference values for each agent across various resources or meetings. In the following sections, we present graphs obtained using the Gaussian distribution for resource allocation and Uniform distribution for event scheduling (avoided others due to space constraints). To avoid any bias in problem generation, each agent had different distribution parameters for its preference values. We performed the following experiments:

- In section V, we study the effect of bias 1 on the solution quality.
- In section VI, we study the effect of bias 2 on the solution quality.
- In section VII, we study the effect of problem size on the solution quality.
- In section VIII, we show experimentally that it may not be easy to exploit DCOPs through a detailed study of the resource assignments and analysis of the agent/human subgroups.
- Finally, in section IX, we perform experiments that show that large problems are in general insensitive and hence may be a good explanation for the results we observed in our previous experiments.

V. EFFECT OF BIAS 1 ON SOLUTION QUALITY

Our first set of experiments investigate the effect of Bias 1, i.e., varying $maxK$. Figure 1 shows two graphs corresponding to the DSA algorithm (similar results were observed with Global algorithm). Graph (a) shows results for the resource

[1]We borrowed the examples from the DCOP repository at $http://teamcore.usc.edu/dcop/$.

Parameter	Variable	Brief description	Possible values
Number of agents	$\|E\|$	Four possible values	$\{5, 20, 100, 300\}$
Number of resources	R	<Low, High> resource settings	$< 4, 20 >, < 12, 50 >, < 70, 500 >, < 100, 600 >$
Bias 1	$maxK$	Four possible settings	$\{0, 1, 2, 3\}$
Bias 2	$\{S, A, X\}$	S and X varied. A set to zero	$S \in \{1, 5, 10\}$, X is 0 unless specified.

TABLE II
SIMULATION SETUP DETAILS FOR DISCRETE RESOURCE ALLOCATION

Parameter	Variable	Brief description	Possible values
Domain size	$\{E, M, E_m, M_e, t\}$	Five possible settings	{5, 8, 4, 4, 6}, {20, 50, 7, 5, 12}, {100, 80, 8, 4, 12}, {250, 350, 8, 9, 20}, {500, 650, 15, 6, 20}
Bias 1	$maxK$	Four possible settings	$\{0, 1, 2, 3\}$
Bias 2	$\{S, A, X\}$	S and X are varied. A is set to zero	$S \in \{1, 5, 10\}$, X is 0 unless specified

TABLE III
SIMULATION SETUP DETAILS FOR DISTRIBUTED EVENT SCHEDULING

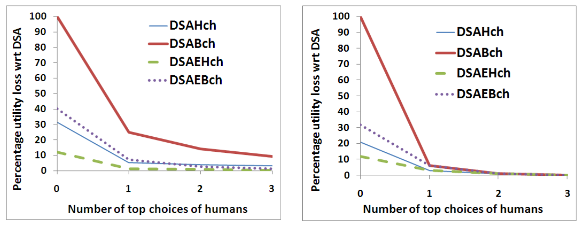
allocation domain while graph (b) shows the same results for event scheduling. Figure 1-a compares the performance of DSA, over the four preference approximations (of Table 1) and the original preferences. X-axis for the graph represents the number of top choices (varied from 0 to 3) with 100 agents, 70 resources and $S = 1$. Y-axis for the graph shows the averaged utility loss over 500 examples with respect to the DSA solution. The following conclusions can be drawn from Graph (a):

1) In the case where only humans report zero choices (Hch) the algorithm has a significant performance decrease with a utility loss of 32%.
2) When both agents and humans report zero choices (Bch) the utility loss becomes 100% since no resource is deemed as useful by the algorithm.
3) On the other hand, the enhanced versions perform better with utility losses of 12% when only humans report zero choices and 40% when both report 0 choices.
4) The scenario quickly changes when $maxK = 2$. In this case, the loss is 4% for Hch and 14% for BHch. The enhanced approximations (EHch and EBch) perform much better with utility losses of less than 3% when $maxK >= 2$.

Graph (b) of Figure 1 shows a similar experiment for the distributed event scheduling domain with similar conclusions. Thus, on both these generic domains, we are able to definitively show that humans need to report only their top few choices (>= 2) without worrying about the result. To measure the statistical significance of our results, we calculated the standard error. Given that we are plotting the mean percentage of utility loss, the standard error is the appropriate measurement to calculate the error bars. For clarity purposes, we avoided showing error bars in our graphs. Unless specified otherwise, in the rest of our experiments the standard error was less than 2% of the mean value i.e. if the mean is shown to be 100, the standard error is ± 2.

VI. EFFECT OF BIAS 2 ON SOLUTION QUALITY

While the above experiments support the fact that team performance does not get affected as long as humans can get their top few preferences correctly, the question arises as to what happens if these preferences get exaggerated. Firstly, we



(a) Resource Allocation (b) Event scheduling

Fig. 1. Utility loss % as a function of top choices reported

consider the effect of scaling factor and then we consider the effect of a random noise added to the preferences.

Figure 2 investigates the effect of increasing the exaggeration factor for the distributed event scheduling domain when there are 100 agents and $maxK = 2$. The X-axis of the figure shows 9 options, 3 options each for the team, agents and humans represented as T, A and H. Factor S was set to one of the 3 choices: 1, 5 and 10. Therefore, 1-T represents the effect of having an exaggeration factor of 1 on the team utility across the 4 preference approximations while 1-A and 1-H investigate the same effect on agents and humans. From the graph, we make the following key observations:

1) When $S = 1$, the average team utility loss is upto 2% for the four preference approximations. When $S = 5$ (for DSAHch and DSAEHch), there is a 4% team utility loss which increases negligibly thereafter to a 5% loss when $S = 10$.
2) The scenario is different for individual subgroups. The agents subgroup has an average utility loss of 2% for a factor of 1, which increases to 13% for a factor of 5 and 14% for a factor of 10. On the other hand, humans gain upto 5% of utility on an average for exaggeration factors of 5 and 10 and lose 2% for a factor of 1.
3) The scenario gets very interesting when agents also use the same biases as humans (DSABch, DSAEBch). In this case, the loss in team utility falls down to 1 and 2% from 4 and 5% respectively. A greater effect is seen in the utilities for agents and humans with agents cutting their utility losses from 13 and 14% to 2% while humans

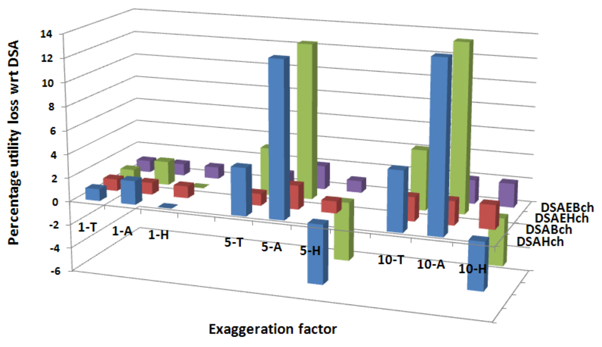


Fig. 2. Effect of increasing S in event scheduling domain

incur a loss of upto 2% in contrast to the gains made earlier.

The overall conclusion from the graph is that exaggerating the preferences has little effect on the team utility on average. However, there are significant differences at the individual level. The same experiment was performed for the discrete resource allocation domain and it yielded similar results.

	5	20	100	250	500
$DSAEHch$	$3(2,5)$	$3(2,5)$	$4(2,6)$	$3(2,5)$	$3(2,4)$
$DSAEBch$	$7(8,6)$	$6(6,6)$	$7(7,7)$	$6(6,7)$	$6(6,6)$

TABLE IV
EFFECT OF GAUSSIAN NOISE ON TEAM UTILITIES.

In Table IV, we show the effects of adding Gaussian random noise (the second aspect of bias 2) on the event scheduling problem. All the results for this experiment were generated by setting X to 25% of the maximum preference value, S to 1 and $maxK$ to 2. The table shows the impact of this bias on the team utility as well as agent and human subteam utilities. The rows of the table represent the two different possibilities for the addition of Gaussian random noise while the columns of the table represent the various team sizes. Each cell in the table shows 3 utilities: Team utility (Agent subteam utility, Human subteam utility).

The key result of this experiment is that the addition of Gaussian random noise has a minor impact on the percentage of utility loss. This number stays constant and is about 3-4% across all the team sizes. The results also show that human utilities (loss of upto 6%) are affected slightly more than the agents (loss of 2%). While an intentional addition of Gaussian random noise to agent's preferences is not practical, we still tested it for consistency purposes. Our results (row 2 of table IV) indicate that the utility loss gets doubled i.e. 6% to 8%. This number while significant represents the extreme case where all the utilities reported are about 25% noisy. In essence, this illustrates the robusness of the algorithm to random noise in preference functions.

	5	20	100	250	500
$DSAEHch$	2.526	23.748	52.996	201.913	363.565
$DSAEBch$	2.716	26.352	61.006	221.904	393.57

TABLE V
NUMBER OF CHANGES IN ASSIGNMENTS

VII. EFFECT OF PROBLEM SIZE ON SOLUTION QUALITY

Our first experiment investigates the effect on solution quality as the number of agents and the resources available to the team vary for the discrete resource allocation domain. Figure 3 contains two graphs. The top graph shows the average percentage of utility lost over the four preference approximations and a random assignment when compared to the optimal assignment of G with original preferences, as the number of agents and resources increase. The bottom figure illustrates the corresponding results with respect to DSA. The x-axis for both these graphs shows the various combinations of number of agents/humans and resources available while the y-axis shows the percentage of utility loss. The parameter $maxK$ was fixed to 2 and S was set to 1. Following are our key observations from the two graphs:

1) The top graph shows that the biased solutions GHch and GBch are within 10% of the optimal solution and reduces to within 7% for the enhanced versions while the random assignment is at least 30% and up to 60% away from the optimal.
2) A similar trend as above is repeated in the bottom figure where three of the four biased DSA algorithms are within 5% of the DSA solution while the random assignment is atleast 20% and upto 50% away from the DSA solution.
3) The biased algorithm DSABch is sometimes upto 13% away from optimal, while the enhanced version (DSAE-Bch) brings down the loss to within 3%.

These observations clearly illustrate that for the discrete resource allocation problems, the modeled biases do not critically impact average team utility irrespective of the number of agents/humans and the number of resources. Note that across all the experiments, DSA had a maximum utility loss of upto 25% wrt the global optimum for the low resource setting, while the loss was less than 1% for the high resource setting.

VIII. INEXPLOITABILITY OF DCOPS

In this section we show that large biases at an individual agent or sub-group level may not translate to large changes in the average team utility. While Figure 2 illustrated this to some extent, we will establish this fact further. Our first experiment in this regard is to investigate the number of changes in assignments (of humans and agents) due to a biased version of the preference function for the event scheduling domain.

Table V shows the number of changes in assignments that happen across the various team sizes when $maxK = 2$ for DSAEHch and DSAEBch. Note that we did not present the results for DSAHch and DSABch since both compute the same policies as DSAEHch and DSAEBch (only the zero valued timeslots are not assigned). As the problem size increases the number of changes in the assignments increases substantially. From our previous experiments we already noted that there are no significant changes in team utilities (less than 5%) for both these preference approximations. Thus, it becomes harder to manipulate DCOPs with simple changes in assignments. Similar trends hold for the resource allocation domain as well.

In our next experiment, we analyze the effect of biases on the two subgroups i.e. agents and humans for the resource allocation domain. In particular, we investigate how the trends

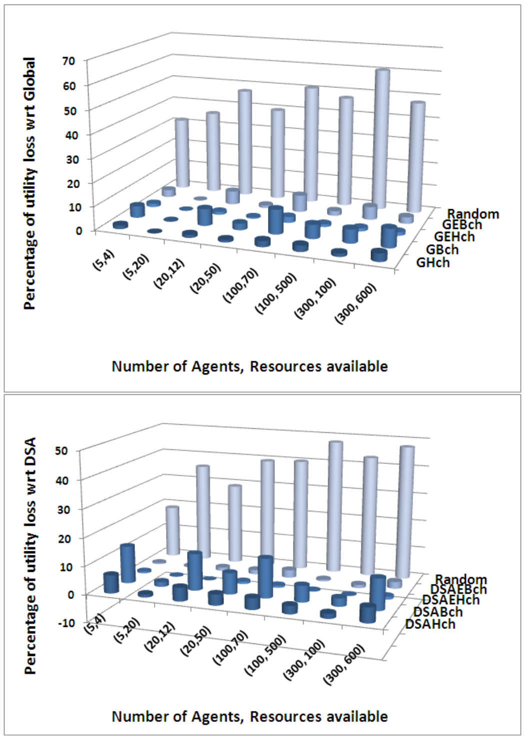


Fig. 3. Utility loss % as the number of (Agents, Resources) increase.

vary as the team sizes increase from 5 to 300 agents when $maxK = 2$ and $S = 1$. Figure 4 shows two graphs – the top graph shows the utility losses with respect to the DSA algorithm for the agents subgroup while the bottom one shows the same graph for the humans subgroup. The top graph shows that the un-enhanced versions have large variations for the agent subgroup. While the results imply that agents have better utilities by reporting all their true preferences, this may not be a general trend. The evidence for this comes from Figure 2 where agents lose by reporting all their preferences. The bottom graph shows that humans as a subgroup can sometimes perform badly if they are the only ones reporting their top choices. A similar analysis for the event scheduling domain revealed that humans may have small gains for the same scenario. This leads us to conclude that there may not be simple general techniques that can ensure guaranteed advantages in utility. Note that the standard error for this experiment was less than 4% across all the settings. While we do not present the graphs here due to space constraints, similar trends hold for the global versions too.

IX. Insensitivity of large problems

In this section we establish empirically that large problems are relatively insensitive to changes in assignments and could be a reason for the minor impact of the biases. To verify this claim we study the effect of making random swaps (i.e. switching allocation of one agent with another randomly chosen agent) in the DSA assignment on the team utility. Figure 5(a) shows the impact of performing random swaps on the DSA assignment in the resource allocation domain while Figure 5(b) shows a similar graph for the event scheduling domain. Both these figures show the effect on a small, medium and large

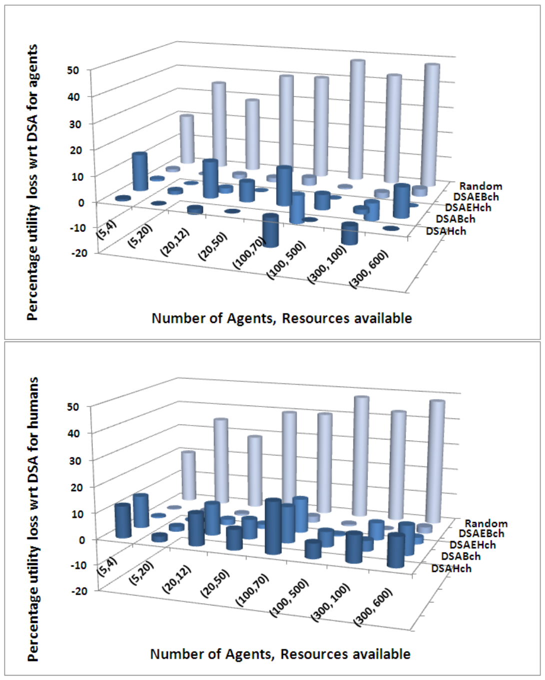
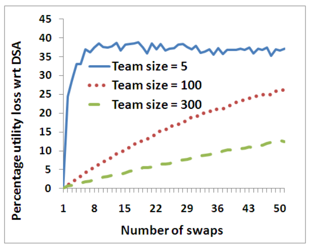
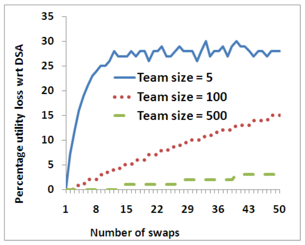


Fig. 4. Utility loss % for agents and humans.



(a) Resource Allocation



(b) Event scheduling

Fig. 5. Utility loss % as function of random swaps

team involving 5, 100 and 300 agents for the allocation domain for the high resource setting and 5, 100 and 500 agents for the scheduling domain. The x-axis of both these figures represent the number of random swaps performed (from 1 to 50) while the y-axis shows the percentage utility loss with respect to the DSA assignment, averaged over 500 runs. Both these graphs show that the solution space for larger problems is smooth and hence small random perturbations have little effect on overall utility (smaller slope). While the graphs presented show upto 50 swaps for presentation purposes, we tested upto 250 swaps and noticed that the lines remain smooth and each line stabilizes to a small interval.

In our next experiment, we study the utility loss patterns of the individual runs (as opposed to averaged statistics in all our previous experiments) for the two biases. The idea here is to find the sensitivity of the individual runs across the various problem sizes and maxK. Figure 6 shows the interval percentage of average utility loss w.r.t. the DSA algorithm on the x-axis. For example, a value of 1 on x-axis would

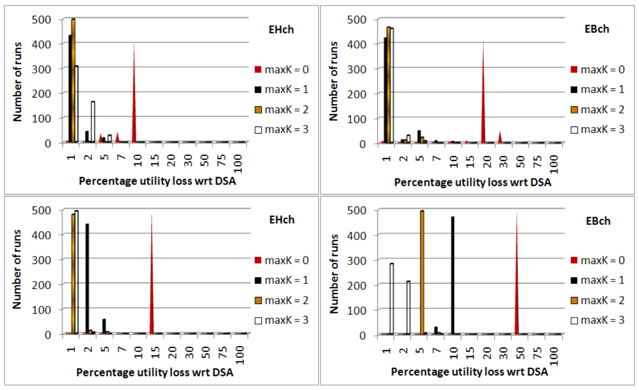


Fig. 6. Number of runs vs percentage of utility loss.

correspond to a utility loss between 0 to 1 percent while a value of 2 would mean a utility loss between 1 to 2 percent and so on. The y-axis shows the number of instances out of 500 runs that have the particular percentage of utility loss for the resource allocation domain. Row 1 corresponds to the smallest problem while row 2 corresponds to the largest problem from our problem set. Column 1 corresponds to the EHch algorithm while column 2 corresponds to EBch. The 4 bars in each figure correspond to the maxK values varying from 0 to 3. The graph shows that for the larger problem as maxK increases the peaks move more gradually to the left implying that specification of a single additional choice may not have as drastic effect in a large problem as opposed to a small problem. Similar trends hold for the meeting scheduling domain. Furthermore, the graph shows a neat classification of the various maxK instances into certain percentage bins i.e. most/all the runs for a given maxK have utility losses that fall into a single interval (or two sometimes) especially for the larger problem. This re-emphasizes the fact that the resource allocation problems modeled here are in general insensitive to realistic human biases and humans can safely specify their top few ($>= 2$) choices while having a small effect on the solution quality most of the times.

X. Related Work

This paper is relevant to three research threads, namely preference elicitation, human biases and distributed constraint optimization (DCOP). In preference elicitation, the key relevant point is that eliciting the correct preferences of humans is difficult on both humans and systems [2]. Due to this elicitation burden on humans, biases creep into the preference values specified by humans [4]. There has been a plethora of work that provide algorithms to solve event scheduling problems using DCOPs. In particular, optimal algorithms have been developed for sensor scheduling, meeting scheduling and task allocation problems [6], [7], [10]. We are interested in modeling mid to large teams and the optimal algorithms cannot scale to such problems and hence not applicable in our context. [9] analyzed the solution space of local optimal algorithms like Distributed Breakout Algorithm (DBA) and Distributed Stochastic Algorithm (DSA) when the coordination levels are changed. Although such an analysis is useful, we are unable to use it in our context due to the biases involved.

XI. Conclusions and Future Work

In this paper, a detailed empirical analysis of the impact of two types of human biases on the outcome of two classes of resource allocation problems using DCOPs was presented. The results showed that for medium and large sized problems, the impact on the overall team utility was relatively small. Moreover, humans were neither advantaged nor disadvantaged versus agents by their inability to precisely and completely specify utilities for all resources. This is exciting for the applicability of DCOPs to real world problems, because it shows that the algorithms will work without the high computational burden of complete, accurate preference elicitation. While we believe that the analysis may apply to a range of DCOP algorithms, we were able to prove that DSA is particularly convenient, since it only requires team members to locally rank preferences correctly.

A key reason why human biases had so little impact on overall utility was that for the large problem instances of interest, it turned out that there were many solutions with approximately the same utility. It is not clear that such *smoothness* will exist for a wider range of resource allocation problems. An immediate direction for future work is to look at other resource allocation problems and attempt to identify instances where large problems have a small number of solutions that are significantly better than other solutions.

Acknowledgments: This research is supported by MURI-7 award AFOSR FA955008-1-0356.

References

[1] T.M. Cioppa, T.W. Lucas, and S.M. Sanchez. Military applications of agent-based simulations. In *Proceedings of the 36th conference on Winter simulation*, page 180, 2004.
[2] Y. Guo, J.P. Muller, and C. Weinhardt. Elicitation of user preferences for multi-attribute negotiation. In *AAMAS*, page 1005, 2003.
[3] R.H. Guttman, A.G. Moukas, and P. Maes. Agent-mediated electronic commerce: A survey. *The Knowledge Engineering Review*, 13(02):147–159, 2001.
[4] J. Huber, D. Ariely, and G. Fischer. Expressing preferences in a principal-agent task: A comparison of choice, rating, and matching. *Organizational Behavior and Human Decision Processes*, 87(1):66–90, 2002.
[5] H. Kitano and S. Tadokoro. Robocup rescue: A grand challenge for multiagent and intelligent systems. *AI Magazine*, 22(1):39, 2001.
[6] R.T. Maheswaran, M. Tambe, E. Bowring, J.P. Pearce, and P. Varakantham. Taking DCOP to the real world: Efficient complete solutions for distributed multi-event scheduling. In *AAMAS*, pages 310–317, 2004.
[7] P.J. Modi, W.M. Shen, M. Tambe, and M. Yokoo. Adopt: Asynchronous distributed constraint optimization with quality guarantees. *Artificial Intelligence*, 161(1-2):149–180, 2005.
[8] P.J. Modi and M. Veloso. Multiagent meeting scheduling with rescheduling. *DCR*, 2004, 2004.
[9] J.P. Pearce, R.T. Maheswaran, and M. Tambe. Local Algorithms for Distributed Constraint Optimization in Dynamic, Anytime Environments. *AmbiAgents Workshop, AAMAS*, 2005.
[10] A. Petcu, B. Faltings, and R. Mailler. PC-DPOP: A new partial centralization algorithm for distributed optimization. In *IJCAI*, volume 7, pages 167–172, 2007.
[11] P. Scerri, A. Farinelli, S. Okamoto, and M. Tambe. Allocating Tasks in Extreme Teams. In *AAMAS*, 2005.
[12] R. Stranders, A. Farinelli, A. Rogers, and N.R. Jennings. Decentralised Coordination of Mobile Sensors Using the Max-Sum Algorithm. In *IJCAI*, 2009.
[13] M. Yokoo, O. Etzioni, T. Ishida, and N. Jennings. *Distributed constraint satisfaction: foundations of cooperation in multi-agent systems*. 2001.
[14] W. Zhang, G. Wang, Z. Xing, and L. Wittenburg. Distributed stochastic search and distributed breakout: properties, comparison and applications to constraint optimization problems in sensor networks. *Artificial Intelligence*, 161(1-2):55–87, 2005.