Heuristic-based Approaches for CP-Nets in Negotiation *

Reyhan Aydoğan[1], Tim Baarslag[2], Koen V. Hindriks[2], Catholijn M. Jonker[2], Pınar Yolum[1]

[1]Boğaziçi University, Bebek, Istanbul, Turkey
{reyhan.aydogan, pinar.yolum}@boun.edu.tr
[2]Delft University of Technology, Delft, The Netherlands
{T.Baarslag, k.v.hindriks, c.m.jonker}@tudelft.nl

ABSTRACT

CP-Nets have proven to be an effective representation for capturing preferences. However, their use in multiagent negotiation is not straightforward. The main reason for this is that CP-Nets capture partial ordering of preferences, whereas negotiating agents are required to compare any two outcomes based on the request and offers. This makes it necessary for agents to generate total orders from their CP-Nets. We have previously proposed a heuristic to generate total orders from a given CP-Net. This paper proposes another heuristic based on Borda count, applies it in negotiation, and compares its performance with the previous heuristic.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent Systems

General Terms

Algorithms, Experimentation

Keywords

Negotiation, Qualitative Preferences

1. INTRODUCTION

Modeling users' preferences is an inevitable part of automated negotiation tools. While reasoning on and representing the user's preferences, there are several issues to be taken into account. One, outcome space grows exponentially with the number of attributes and their possible values. It may be infeasible to ask a user to order or rank all outcomes when the outcome space is very large. Two, the user may have difficulty in assessing her preferences in a quantitative way [**?**]. Representing someone's preferences with numerical values is an arduous task for a human. Three, it is difficult to find a mathematical model for representing some preferences such as conditional preferences in which there are preferential dependencies between attributes. Therefore, it is more effective and intuitive to use a qualitative preference model.

Although it is desired for users to express their preferences qualitatively, most of the current negotiation strategies [**?**, **?**, **?**, **?**, **?**] work with quantitative preferences. Hence, to use qualitative preferences in negotiation, it is necessary to estimate quantitative preferences from qualitative preferences. Accordingly, this paper is about estimation of quantitative preferences from qualitative preferences. That is, we propose heuristics to allow agents to have a qualitative preference model, while their negotiation strategy employs quantitative information. In order to do so, we start from a qualitative preference representation, namely CP-Nets. CP-Nets allow representation of conditional preferences and tolerate partial ordering. We extend the GENIUS negotiation framework [**?**] to allow elicitation of acyclic CP-Net preferences. Then, we apply our heuristics to generate utility-based information from the quantitative representation of the preferences.

The goal of this study is to compare the performance of the selected heuristics in a realistic negotiation setting. Thus, we compare the performance of agents when they apply heuristics on their users' qualitative preferences in the form of CP-Nets and negotiate with estimated utilities versus when they have their users' real total preference orderings in the form of UCP-Nets and negotiate with real utilities. To accomplish this, users were asked to create their preference profiles both quantitatively (UCP-Nets) and qualitatively (CP-Nets), using the GENIUS interface for an apartment renting domain. The given UCP-Nets serve as ground truth. The agents apply heuristics on the given CP-Net and then negotiate with the resulting estimated utilities. Each negotiation outcome is evaluated based on the given UCP-Net, which is not only consistent with the CP-Net but also provides a total ordering of outcomes rather than a partial ordering.

The rest of this paper is organized as follows: Section 2 and Section 3 gives an introduction on CP-Nets and UCP-Nets, respectively. Section 4 explains the heuristics that we propose to be used with CP-Nets. Section 5 explains our experimental setup, metrics, and results. Finally, Section 6 discusses our work.

2. CP-NETS

Conditional preference networks (CP-nets) is a graphical model for representing qualitative preferences in a compact way [**?**]. In CP-nets, each node represents an attribute (issue) and each edge denotes preferential dependency between nodes. Here, if there is an edge from X to Y, X is called "parent node" and Y is called "child node". The preference on child nodes depends on their parent nodes' values. To express conditional preferences, each node is associated with a conditional preference table (CPT), which represents a total order on possible values of that node with respect to its parents' values.

Consider apartment renting domain in Example 1 and CP-net depicted in Figure 1. According to this CP-net, the user's preference on parking area depends on neighborhood. CPT for *Parking Area* shows that the user prefers an apartment having a parking area when the neighborhood is either *Kadikoy* or *Kartal*. However, she prefers an apartment not having a parking area when it is at *Etiler*. Note that in CP-nets, each preference statement is interpreted under "everything else being equal" interpretation. The statement, "*Etiler* is preferred over *Kadikoy* for neighborhood", means that if all other attributes such as price and parking area are the same, an

*This research is supported by Boğaziçi University Research Fund under grant BAP5694.

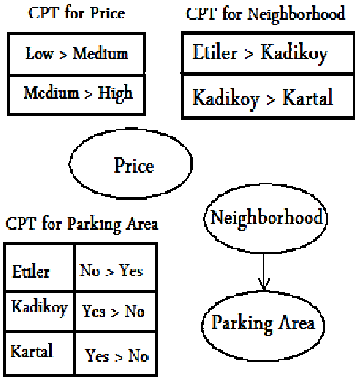

Figure 1: A sample CP-net for apartment renting domain

apartment at *Etiler* is preferred over an apartment at *Kadikoy*.

EXAMPLE 1. *For simplicity, we have only three attributes in our apartment renting domain: Price, Neighborhood and Parking Area. There are three neighborhoods: Etiler, Kadikoy and Kartal whereas the valid values for the price are categorized as High, Medium and Low. A parking area may exist or not. Thus, the domain for parking area has two values: Yes and No.*

In acyclic CP-Nets, there are only one best outcome so it is straightforward to determine the best outcome by answering the "outcome optimization query". From ancestors to descendants, the most desired value for each attribute is chosen in order to get the best offer. However, we need to check whether there exists an *improving flip* sequence from one outcome to another (and vice versa) to answer dominance queries (whether an outcome would be preferred over another). An improving flip is changing the value of a single attribute with a more desired value by using CPT for the attribute. If we cannot reach one outcome from another and vice versa via improving flip sequences, we cannot compare these two outcomes. The fact that we may not be able to compare some outcomes is the challenge of using CP-Nets in negotiation.

3. UCP-NETS

Boutilier *et al.* propose a graphical preference model namely UCP-nets [**?**] by combining a well-known qualitative preference representation, CP-Nets with generalized additive models. Hence, UCP-nets are able to represent preferences in a quantitative way rather than representing simply preference ordering. GAI-models [**?**] perform dominance queries (whether an outcome would be preferred over another) straightforwardly whereas CP-Nets perform outcome optimization queries (maximal outcome) straightforwardly. Therefore, it is claimed that by the combination of both models, UCP-Net becomes more powerful [**?**].

Figure 2 shows a sample UCP-Net, which is consistent with CP-Net seen in Figure 1. Similar to CP-nets, we firstly specify preferential dependency among attributes. Instead of specifying a total preference ordering over the values of each attribute according to their parents' values (conditions), we assign a real value (utility) for all values of each attribute by taking conditions into account. For instance, when neighborhood is *Etiler*, the value of having a parking area (*Yes*) is specified as 0.8. Notice that the real values are not restricted to be between zero and one in this model. Utility function *u(X_1, X_2,...X_n)* is represented in Equation 1 where X_i is the i^{th} attribute of outcome, U_i denotes parents of X_i and $f_i(X_i, U_i)$ represents a factor. For example, our sample UCP-Net

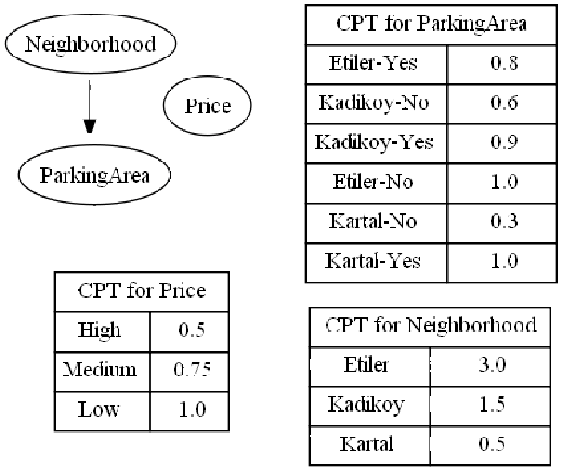

Figure 2: Sample UCP-net

involves three factors f_1(Neighborhood), f_2(Price) and f_3(Parking Area, Neighborhood). The utility of an outcome is estimated as the sum of these factors. For example, the utility of (*No, Etiler, High*) is equal to 4.5 $(= 1.0 + 3.0 + 0.5)$.

$$u(X_1, X_2, ...X_n) = \sum_i f_i(X_i, U_i) \quad (1)$$

In CP-Nets, it is implicitly induced that an ancestor has higher priority over its descendants. Note that this property constitutes a key role in UCP-nets in which each attribute should dominate its children. When we assign utilities, we need to ensure that each node dominates its children. There are several ways of verifying whether the constructed graph is a valid UCP-net (whether it satisfies the CP-relations among attributes).

One method for verifying UCP-nets, is to compute the values of *Maxspan* and *Minspan* for variables and check whether Equation 2 is satisfied or not. In this equation X and Y are attributes and Y_i are the children of X. Note that Minspan(X) is the minimum difference between the utilities of each possible values of X for the given parent value. If this is the case, we can say that a given graph is a valid UCP-net. For example, in our sample UCP-net *Neighborhood* is a parent of *Parking Area*. We estimate the value of *Maxspan(Parking Area)* the way that we evaluate the maximum difference between the utilities of the values of Parking Area for each possible value of its parent [Etiler: 0.2 $(1 - 0.8)$, Kadikoy: 0.3 $(0.9 - 0.6)$ and Kartal: 0.7 $(1 - 0.3)$] and choose the maximum among them (Maxspan=0.7). To be able to get a valid UCP-net satisfying the CP-relations, the difference between the utilities of each possible value of *Neighborhood* should be at least 0.7. In our example, it is equal to 1.0 $(1.5 - 0.5)$, which is higher than 0.75. Thus, this graph is a valid UCP-net:

$$Minspan(X) >= \sum_i Maxspan(Y_i). \quad (2)$$

Since we use normalized utility values between zero and one in our negotiation setting, the utility of outcome is divided by the utility of the best outcome (whose utility is the highest). For this case, the utility of best outcome is 5 (=1+1+3). Thus, the normalized utility of (*No, Kartal, High*) would be $0.26 (= 1.3/5)$.

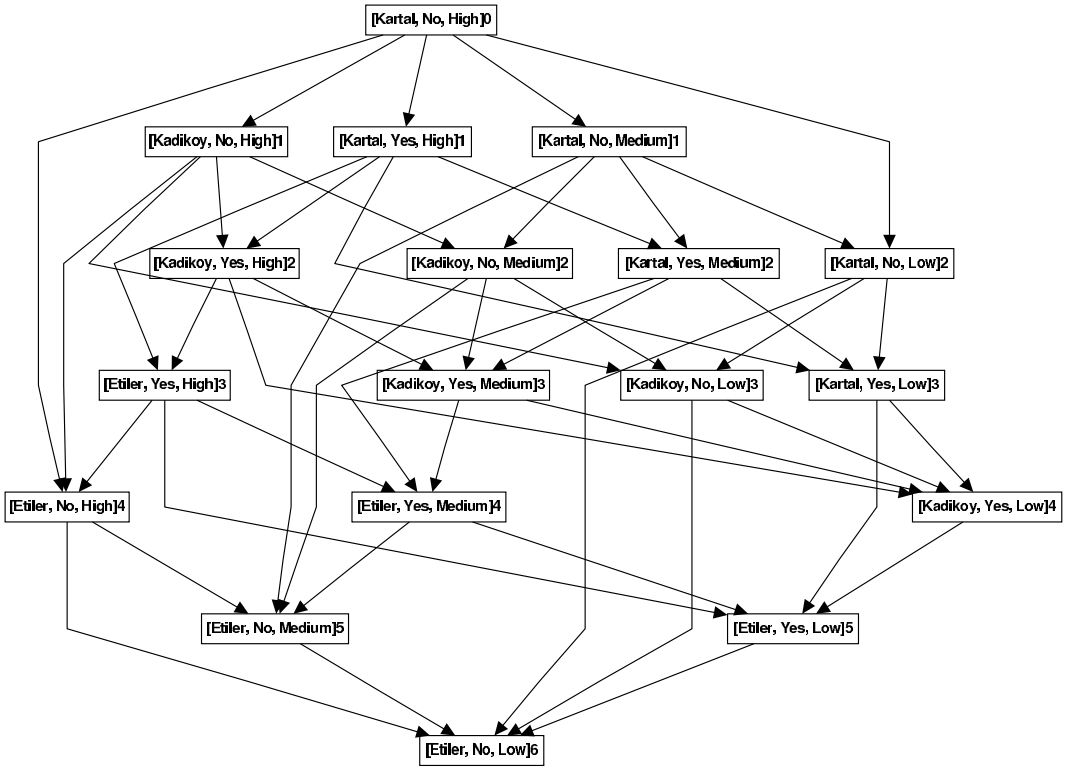

Figure 3: Induced preference graph from the CP-Net in Figure 1.

4. PROPOSED HEURISTICS

Most of the negotiation strategies [**?**, **?**, **?**, **?**, **?**, **?**] work with quantitative preferences such as *utility functions*. However, it is desired for users to express their preferences qualitatively. Thus, we propose heuristics to use acyclic CP-Nets (a qualitative preference model) in negotiation while agents still negotiate with their strategies using quantitative information, *utility* (a real value between zero and one). To do this, we generate artificial utilities from a given CP-Net by applying our heuristics.

In our framework, a preference graph is induced from a given CP-Net while eliciting a user's preferences in the form of CP-Net. In this preference graph, each node denotes a possible outcome and each edge represents an improving flip. Note that the direction of edges are ordered from less desired to more desired services. Therefore, the least desired (worst) outcome will be placed at the top of preference graph (root node) whereas the leaf node holds the best outcome. For intermediate nodes, we only compare the nodes having a path from others. The nodes having no path to each other cannot be compared under "everything else being equal" interpretation.

Consider the preference graph in Figure 3 induced from the CP-Net in Figure 1. Here, the node (*Yes, Etiler, Low*) represents a low-priced apartment at *Etiler* having a parking area. It is seen that there is an edge from (*No, Kartal, High*) to (*No,Kartal, Medium*). It can be inferred that an apartment with a medium price at *Kartal* not having a parking area is preferred over an apartment with a high price at *Kartal* not having a parking area.

Before negotiating, the agent applies one of the proposed heuristics and uses the estimated utilities produced by a chosen heuristic. The proposed heuristics are:

- Depth Heuristic (Section 4.1)
- Borda Scoring Heuristic (Section 4.2)

4.1 Depth Heuristic (DH)

We have previously proposed an approach based on capturing the depth of an outcome in preference graph [**?**] but in that study *depth* is used by the proposed negotiation strategy – it is not independent from the negotiation strategy. However, in this study we use the concept of *depth* to produce estimated utilities of outcomes regardless of negotiation strategy. That is, the agent using this heuristic is able to apply any negotiation strategy (working with utility values).

Depth of an outcome node in a preference graph indicates how far it is from the worst choice; in other words the highest distance from the worst outcome. It is intuitive to say that the better (more preferred) a service is, the further from the worst outcome it is. Depth of an outcome node is estimated as the length of the longest path from the root node. Note hat the root node in the preference graph represents the worst choice whereas the leaf node denotes the best outcome.

The intuition here is that we know that if there is an edge from x to y, we ensure that y is preferred over x and the depth of y is higher than that of x. According to this approach, the higher the depth of a service, the more likely it is to be preferred by the user. Further, if two outcomes are at the same depth, it is assumed that these services are equally preferred by the user. We apply Equation 3 to estimate the utility values between zero and one. In short, the depth of a given outcome is divided by the depth of the preference graph (the highest depth) to obtain estimated utility of that outcome. For example, if we have a preference graph with a depth of 6 in Figure 3, an outcome whose depth is equal to 3 will have

utility of $0.5(= 3/6)$ according to this approach.

$$U(x) = \frac{Depth(x, PG)}{Depth(PG)} \quad (3)$$

4.2 Borda Scoring Heuristic

CP-Nets orders outcomes partially in which we cannot compare some outcomes. If we would have a total linear ordering of outcomes, we would be able to compare all of the outcomes with each other. And we know that there are a plenty of linear orderings consistent with the partial preference ordering induced from a given CP-Net. One of these linear orderings may reflect the user's real preference orderings. Thus, this heuristic is based on finding all possible linear extensions of a given partial preference ordering and selecting one of the most suitable linear extensions.

At that point, the question "How do we choose one of the most suitable linear orderings?" arises immediately. One possible answer may be to take the benefit from voting theory and apply a voting procedure to get one candidate linear ordering. For this purpose, we estimate all linear extensions of a given partial preference ordering induced from a preference graph and apply a voting procedure called "Borda Rule" [**?**] in order to obtain one of the most suitable linear orderings.

According to Borda Rule, we score outcomes according to their position in the ordering. Let assume that we have *m* alternatives ordered as $< o_1, o_2 ... o_m >$ where o_{i+1} is preferred over o_i. When we score the outcomes, each outcome will get a point of its position minus one, which means that o_i will get $i - 1$. The sum of points namely *Borda count* represents the aggregation of existing alternative orderings. To illustrate this, consider we have three orderings such as $< x, y, z >$, $< z, x, y >$, $< x, z, y >$ where x, y and z are possible outcomes. Borda count of x would be equal to one $(= 0 + 1 + 0)$ whereas that of y would be five $(= 1 + 2 + 2)$. In this approach, Borda count of each outcome over all possible linear extensions will reflect how much that outcome is preferred. Thus, we will estimate utilities based on the estimated Borda counts.

This heuristic uses Varol Rotem Algorithm [**?**] to find all possible linear extensions of a given partially ordered outcomes (*poset*). This algorithms takes one linear ordering consisting with the given preference graph and produces all possible linear orderings. It needs $O(e(P))$ or $O(n \cdot e(P))$ time depending on which linear ordering is used at the beginning [**?**].

One of the problems with this approach is that the number of all possible linear extensions of a given partial ordering may be so huge that this technique may become impractical because of high complexity. In order to reduce the complexity, we partition the preference graph and apply Borda Rule to all possible linear extensions of each subpartition.

How do we partition the preference graph? We know that the root node holds the worst outcome while the leaf node holds the best outcome. Thus, we need to find an ordering for the outcomes within the intermediate nodes. We partition this part in such a way that each subpartition can involve at most n, predefined number of outcomes. For this purpose, n can be taken as 10 or 15 according to the size of the preference graph. Note that we choose 10 in this study. Figure 4 shows how we partition the preference graph induced from the CP-Net in Figure 1.

After applying Borda rule to each partition, we normalize Borda counts in a way that Borda count of each outcome will be between zero and one. To do this, we can divide Borda count of each outcome in that partition by the maximum Borda count. Consider we have four outcomes (x, y, w and z) in a partition and their corresponding Borda counts are $25, 10, 20, 30$ respectively. The normalized Borda counts would be $25/30$, $10/30$, $20/30$, $30/30$.

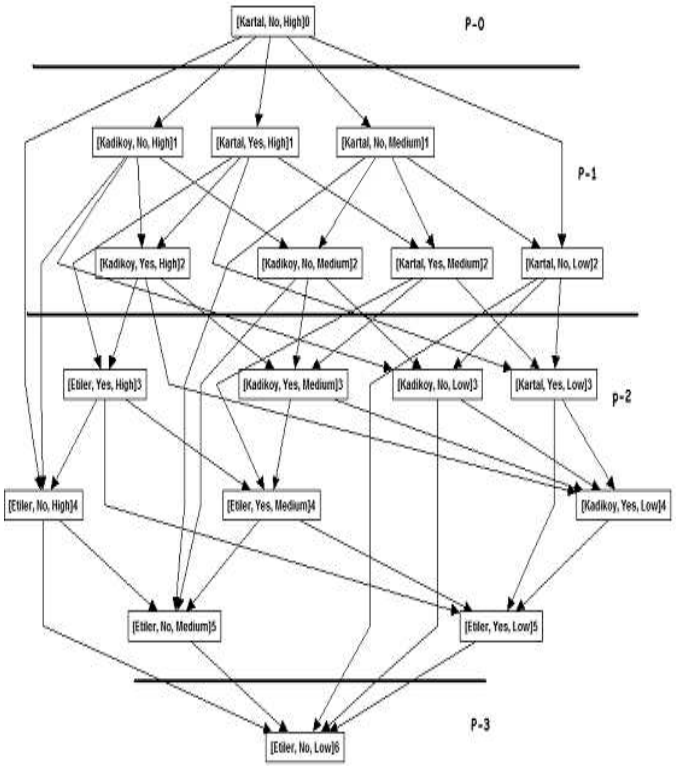

Figure 4: Partitioning Preference graph in Figure 3.

$$U(x, p_i) = U_{max}(p_{i-1}) + \frac{S_{p_i}}{N} * BRCount(x, p_i) \quad (4)$$

Another issue pertains to using these normalized Borda counts in order to estimate final utilities. We distribute the utilities by taking the number of outcomes at each partition into account. To achieve this, we apply the formula in Equation 4 where $U(x, p_i)$ denotes the utility of outcome x in the i^{th} partition, $U_{max}(i-1)$ denotes the utility of outcome whose utility is maximum in the previous partition $(i - 1)$, N denotes the number of possible outcomes, S_{p_i} denotes the number of outcomes in i^{th} partition and $BRCount(x, p_i)$ denotes normalized Borda count of the outcome x. Note that $U_{max}(p_0)$,the utility of worst outcome (root node in the preference graph), is equal to $1/N$. Furthermore, the utility of the best outcome (leaf node in the preference graph) is equal to one according to this approach.

Table 1 shows the estimated utilities for each heuristic when we have the CP-net in Figure 1 and the last column of this table shows utilities estimated by the UCP-net in Figure 2, which is consistent with the given CP-net.

5. EXPERIMENTS

To evaluate the proposed heuristics, we extend GENIUS [**?**], which is a platform for bilateral negotiation. Our extension enables an agent to elicit user's preferences in the form of CP-Nets and to use utilities estimated by chosen heuristic while negotiating with another agent. In this setting, the platform also keeps the user's total ordering of outcomes in the form of UCP-Nets and evaluates each negotiation outcome for that agent based on the given UCP-Net. The given UCP-Net is consistent with the given CP-net. In our

Table 1: Estimated Utilities for Each Heuristic and UCP-net

Service	DH	BSH	UCP-net
[Etiler, No, Low]	1.0	1.0	1.0
[Etiler, No,Medium]	0.83	0.92	0.95
[Etiler, Yes, Low]	0.83	0.94	0.96
[Etiler, Yes, Medium]	0.67	0.74	0.91
[Kadikoy, Yes, Low]	0.67	0.82	0.68
[Etiler, No, High]	0.67	0.72	0.90
[Kadikoy, No, Low]	0.5	0.60	0.62
[Kadikoy, Yes, Medium]	0.5	0.54	0.63
[Etiler, Yes, High]	0.5	0.52	0.86
[Kartal, Yes, Low]	0.5	0.60	0.5
[Kadikoy, No, Medium]	0.33	0.43	0.57
[Kadikoy, Yes, High]	0.33	0.44	0.58
[Kartal, Yes, Medium]	0.33	0.43	0.45
[Kartal, No, Low]	0.33	0.39	0.36
[Kartal, No, Medium]	0.17	0.13	0.31
[Kartal, Yes, High]	0.17	0.17	0.40
[Kadikoy, No, High]	0.17	0.17	0.52
[Kartal, No,High]	0.0	0.06	0.26

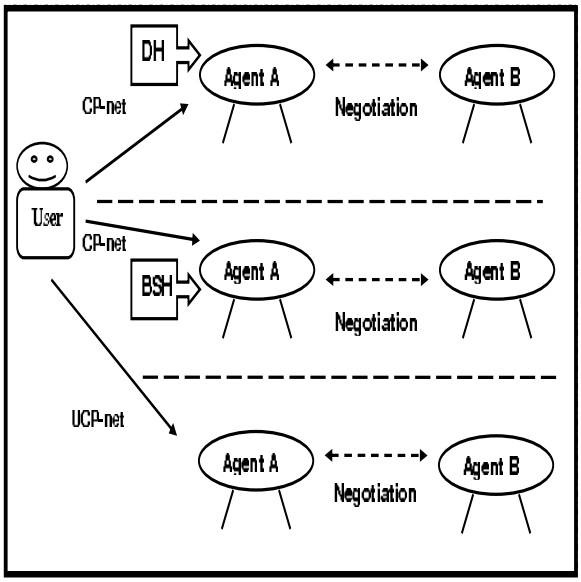

Figure 5: Experiment Set-up for Comparison of Heuristics

experiments, the UCP-Net serves as ground truth. After an agent negotiates using its CP-Net, we evaluate its performance as if we knew the correct total ordering (UCP-Net).

In order to compare the performance of the heuristics, we investigate three test cases depicted in Figure 5. In each test case, two agents *Agent A* and *Agent B* negotiate with each other. We fix both agents' negotiation strategies so that *Agent A* negotiates with the same *Agent B* (having same preference profile and strategy). In the first case, *Agent A* has a CP-net and applies Depth Heuristic (DH) to derive the estimated utilities. During the negotiation, the agent will act on according to these estimated utilities. In the second case, *Agent A* has the same CP-Net with the first case but it applies Borda Scoring Heuristic (BSH) to estimate utilities which will be used in negotiation. In the last case, *Agent A* has its user's real total preference orderings in the form of UCP-Net (consistent with the CP-net and able to compare all outcomes). Thus, it uses the real utilities during the negotiation. Consequently, we are able to observe what the agent gets at the end of negotiation when it applies heuristics on partial preference information (CP-Net) versus when it has total preference information (UCP-Net).

In our experiments, each agent uses a concession based strategy in which the agent starts with the outcome having the highest utility and concedes over time. It also remembers the best counter offer that is made by the opponent agent. If the utility of the current counter offer is higher than or equal to the utility of agent's previous offer, then the agent will accept the offer. The agent will take the best counter offer of its opponent into account while generating its offer. If the utility of the current offer is lower than that of the best counter offer, the agent will take the opponent's best counter offer.

Since the opponent agent (*Agent B*)'s preference profile has a significant impact on negotiation outcome, we generate 50 different preference profiles for *Agent B*. That is, the same *Agent A* will negotiate with 50 different *Agent B*s. Note that *Agent B*'s preferences are represented with a linear additive utility function in this experiment. Another factor having an influence on negotiation outcome in this setting is UCP-Net of the user. Different UCP-Nets mean different ordering of outcomes (different preference profiles for *Agent A*), so represent different users. Thus, we generate four different UCP-Nets for *Agent A* consistent with the given CP-net—four different users having the same CP-net. As a result, both agents will negotiate 200 times (4 different users of *Agent A* * 50 different *Agent B*) and the performance of the proposed approached will be performed over these 200 negotiations.

We define two evaluation criteria for comparison:

- Sum of outcome utilities for *Agent A* over 50 negotiations (Section 5.1)
- Number of times that the agent using a heuristic negotiates at least as well as the agent having UCP-Net (Section 5.2)

Furthermore, we investigate the performance of the heuristics in negotiation from a different point of view by taking the structure of CP-Nets into account. For this purpose, we generate three different CP-Nets: *CPNet*-1, *CPNet*-2 and *CPNet*-3. *CPNet*-1 involves one dependency such as preference of *parking area* depends on *neighborhood* whereas *CPNet*-2 involves two dependencies such as both preferences of *parking area* and *price* depend on *neighborhood.* There are not any dependencies between attributes in *CPNet*-3. For each CP-Net, we generate four different UCP-Nets consistent with them and perform the experiments mentioned above.

5.1 Sum of Utilities for Agent A

Our first evaluation criterion is the sum of negotiation outcomes' utilities with respect to *Agent A* over 50 different negotiations with *Agent B*. Note that the utility of an outcome is a real value between zero and one. Table 2 shows these total utilities for three different CP-Nets and four different UCP-Nets consistent with each CP-Net. As expected *Agent A* using UCP-Net gets the highest score when it has a consistent UCP-Net with *CPNet*-1 and *CPNet*-3 since it negotiates with user's real preference orderings. Overall, the performance of the agent using BSH is quite close to that of the agent using UCP-Net (172 vs 179 and 171 vs. 172). For the case of *CPNet*-2 (having two dependencies), the score of BSH is approximately the same with the score of of UCP-net. Since *CPNet*-2 involves two dependencies (the user specifies her preferences in a more detailed way), the agent may get more information than the case of other CP-Nets (one dependency and no dependency). This

leads to better results. Note that the score of heuristics are the highest when they have *CPNet*-2.

Table 2: Sum of Outcome Utilities over 50 Negotiations for *Agent A*

AGENT A	DH	BSH	UCP-Net
CPNET-1 with UCPNet-1A	39.03	39.00	41.88
CPNET-1 with UCPNet-2A	38.27	40.73	43.66
CPNET-1 with UCPNet-3A	45.73	45.69	45.80
CPNET-1 with UCPNet-4A	46.88	46.94	47.29
Overall Sum (200 negotiations):	**169.91**	**172.36**	**178.63**
CPNET-2 with UCPNet-1B	39.93	41.66	41.70
CPNET-2 with UCPNet-2B	42.94	43.56	43.21
CPNET-2 with UCPNet-3B	46.15	46.76	46.75
CPNET-2 with UCPNet-4B	42.18	43.56	43.53
Overall Sum (200 negotiations):	**171.20**	**175.55**	**175.20**
CPNET-3 with UCPNet-1C	40.17	41.61	40.83
CPNET-3 with UCPNet-2C	41.58	42.50	45.64
CPNET-3 with UCPNet-3C	42.83	43.97	43.37
CPNET-3 with UCPNet-4C	42.36	43.36	42.64
Overall Sum (200 negotiations):	**166.94**	**171.44**	**172.48**

Moreover, it is seen that *Agent A*'s score while applying Borda Scoring Heuristic (BSH) is higher than the case in which it uses Depth Heuristic (DH) for all CP-Nets (based on overall sum over 200 negotiations). According to this criterion, BSH may be preferred over DH.

5.2 Number of Times as Well as UCP-Net

Our second evaluation criterion is the number of times that the agent that applies a heuristic on a given CP-Net negotiates at least as well as the agent having a UCP-Net. For each CP-Net and UCP-Net pairs, we compare the negotiation outcome for the agent using a heuristic with that for the agent having UCP-Net. If the utility of outcome for the agent using a heuristic is higher than or equal to the utility of outcome for the agent having UCP-Net, that agent receives one point. Since 50 different opponent agents (*Agent B*) negotiate with the same agent (*Agent A*), we count the number of times that *Agent A* using a heuristic is successful at least as the agent having UCP-Net over 50 negotiations.

According to Table 3, when *Agent A* uses *CPNet*-1 and applies DH, it negotiates at least as well as the agent having total preference ordering (UCP-Net) in 78 per cent of negotiations whereas BSH is successful at least as UCP-Net in 76 per cent of negotiations. Although the performance of BSH with respect to sum of utilities (Section 5.1) is better than the performance of DH, it negotiates as successfully as UCP-Net more than BSH for *CPNet*-1 (78 per cent versus 76 per cent). This stems from the fact that when BSH completes a negotiation better than DH, the difference between utilities of the outcomes is much higher than the case when DH negotiates better than BSH.

For *CPNet*-2 and *CPNet*-3, the agent using BSH negotiates successfully as the agent having UCP-Net more than the agent using DH. When agents have *CPNet*-2, it is seen that BSH beats DH. Note that in 89.5 per cent of negotiations DH negotiates at least as well as UCP-Net whereas 98.5 per cent of negotiations BSH performs at least as good as the UCP-Net.

6. DISCUSSION

We present two heuristics namely *Depth* and *Borda Scoring* to negotiate with CP-Nets and compare them in a realistic negotiation setting. According to our experimental results, it would be better to apply Borda Scoring heuristic in small domains since its performance is higher than the performance of Depth heuristic. However, we may prefer to use Depth Heuristic in large domains since its complexity is lower than Borda Scoring heuristic.

Table 3: Number of Times Heuristics Performs As Well As UCP-Nets

AGENT A	DH	BSH
CPNET-1 with UCPNet-1A	40	35
CPNET-1 with UCPNet-2A	26	35
CPNET-1 with UCPNet-3A	46	38
CPNET-1 with UCPNet-4A	44	44
Overall Sum (200 negotiations):	**156**	**152**
CPNET-2 with UCPNet-1B	43	49
CPNET-2 with UCPNet-2B	48	48
CPNET-2 with UCPNet-3B	44	50
CPNET-2 with UCPNet-4B	44	50
Overall Sum (200 negotiations):	**179**	**197**
CPNET-3 with UCPNet-1C	44	50
CPNET-3 with UCPNet-2C	27	31
CPNET-3 with UCPNet-3C	45	49
CPNET-3 with UCPNet-4C	47	47
Overall Sum (200 negotiations):	**163**	**177**

Li *et al.* study the problem of collective decision making with CP-Nets [**?**]. Their aim is to find a Pareto-optimal outcome when agents' preferences represented by CP-Nets. They firstly generate candidate outcomes to increase the computational efficiency instead of using the entire outcome space. Then each agent in the system ranks these candidate outcomes according to their own CP-Nets. For ranking an outcome, they use *the longest path between the optimal outcome and that outcome* in the induced preference graph. Thus, the minimum rank is desired for the agents. They choose the final outcome for the agents by minimizing the maximum rank of the agents. In contrast, we use *the longest path between the worst outcome and that outcome* to estimate the utilities with our depth heuristic. Furthermore, while they propose a procedure for collective decision making with the aim of choosing one outcome for multiple agents, we focus on estimating utility values of each outcome that will be used during the negotiation for an individual agent.

Rosi *et al.* extend CP-Nets to capture multiple agents' preferences and present *m*CP-Nets [**?**]. They propose several voting semantics to aggregate agents' qualitative preferences and to determine whether an outcome is preferred over another for those agents. One of the semantics provides a measure of quantifying the quality of outcomes with respect to the agent's preferences. According to this approach, the rank of an outcome is estimated by the length of the shortest sequence of worsening flips between that outcome and one of the optimal outcomes. Note that they use partial CP-Nets which may lead more than one optimal outcomes. It is worth noticing that Li *et al.* use the longest path instead of the smallest path in their study [**?**] while we use the longest sequence of improving flips between the worst outcome and that outcome in our depth heuristic to get the estimated utilities.

Son and Sakama propose to formalize negotiation processes by using logic programming [**?**]. Consistency restoring rules are used to represent each agent's knowledge. These rules are supported by a set of assumptions and ordered goals (preferences). Since an agent has incomplete knowledge about its opponent, it needs to reason on its own preferences under this uncertainty. Son and Sakama use

answer sets to generate offers and determine if an offer is acceptable. In our work, the agent reasons on its user's partial ordering of preferences to negotiate effectively.

Chalamish and Kraus presents an automated mediator for bilateral negotiations in which agents share their qualitative preferences only with the mediator [**?**]. For representing agents' qualitative preferences, an extension of CP-Nets, namely Weighted CP-Nets (WCP-Nets) are proposed and used. Note that their addition is the property of weights – the weighted importance table. After agents send their WCP-Nets to the mediator, it sorts all possible outcomes with respect to agents' preferences separately – resulting in two sorted list of outcomes. While sorting outcomes, the mediator uses an enhanced version of majority lexicographical (ML) ordering [**?**] with weights. In ML ordering [**?**], a hierarchy of variables in CP-Nets is taken into account to evaluate the outcome. To illustrate this, assume that we have three variables X, Y and Z and a CP-Net saying that $x_1 > x_0$, $x_1 : y_1 > y_0$, $x_0 : y_0 > y_1$, $x_1 : z_1 > z_0$ and $x_0 : z_0 > z_1$. X is in the first level while Y and Z are in the second level. According to ML ordering, the outcome $x_1y_0z_0$ dominates $x_0y_0z_0$ since the former outcome wins in X, which is in the first level (not need to check for the variables in the second level). The weighted version of ML presented by Chalamish and Kraus is based on comparison of weighted sum of the values that the outcome gets for the assignment of its variables. For instance, in case of $x_1 > x_0$ the outcome including x_1 gets one while the outcome including x_0 takes zero. To sum up, the mediator sorts outcomes with this metric and recommends pareto-optimal outcomes the agents to speed up the negotiation. While the mediator uses other agents' WCP-Nets to suggest pareto-optimal outcomes to the negotiating agents, in our study the agent tries to derive estimated utilities that will be employed by the agent's negotiation strategy from that agent's CP-Net. Furthermore, we use the induced preference graph covering possible sequences of improving flips while applying our heuristics and compare the performance of our heuristics with users' real preferences in a negotiation setting.