

Exploiting Preferences for Minimal Credential Disclosure in Policy-Driven Trust Negotiations

Philipp Kärger, Daniel Olmedilla, and Wolf-Tilo Balke

L3S Research Center and Leibniz University of Hannover, Hannover, Germany
{kaerger,olmedilla,balke}@L3S.de

Abstract. Business processes in open distributed environments such as the Web force users to interact with other parties be it users or companies even if they have never had any common transaction in the past. Policy-driven trust negotiations emerged in order to address these situations. But although many policy languages and protocols have been defined, the problem of deciding which credential disclosure set to choose from those that possibly make a negotiation succeed is still subject of research. This paper explores the use of qualitative preferences in order to solve the problem and exploits the recently introduced notions of amalgamated and incremented preferences in order to allow for automated decisions which negotiations are preferred by the user. Our solution eases the task for the user of selection among all possible negotiations by removing irrelevant alternatives and it may even automatize negotiations that otherwise would require manual intervention.

1 Introduction

Open distributed environments such as the Web allow users to interact even if they have never had any common transaction in the past. However, in this situation, traditional access control mechanisms do not apply anymore. Identity-based solutions are of no use when deciding whether to interact or to disclose information to a stranger. In order to address this situation, so-called attribute-based access control mechanisms have emerged and among them, trust negotiation [22] has emerged as a flexible yet powerful solution. Trust negotiation permits both parties willing to interact to specify statements (aka. policies) with requirements to be satisfied by the other party and establishes an iterative process in which this information is disclosed. Typically, each step in the negotiation may include the disclosure of a requested credential and therefore increases the level of trust between the parties towards the success of the negotiation.

Many languages to specify access control policies for trust negotiation have been lately created, e.g., [14,4,16], focusing on different application scenarios and providing different expressivity. However, although they are able to find successful negotiations if they exist, they do not provide the user with means to decide in advance which one to choose in case there are several alternatives.

In this paper we address the problem from a user perspective: our approach provides the user with control over the negotiations her trust agent performs

by exploiting preferences among credential disclosures such as “I prefer to disclose my e-mail address before I disclose my telephone number”. By doing this a personalized view to the negotiation process is given, and, further, we improve security since nothing is disclosed which is not preferred by the user, i.e., which would have been of higher sensitivity to the user. We build on new theoretical research results from preference handling which is a very promising field in database retrieval. We further extend these results in order to provide a theoretical basis of how to use preference constructs in order to personalize the negotiation process.

The rest of the paper is organized as follows: Section 2 introduces a running example and motivates the need for a user-based selection among the different alternatives a negotiation may raise. Section 3 compares our approach to existing related work. An introduction to preference theory is provided in Section 4. Section 5 extends current state of the art in preference handling to address disclosure set selection and demonstrates our approach using our running scenario as an example. Section 6 presents our implementation and experiments. Finally, Section 7 concludes the paper and outlines further work to be performed.

2 Credential-Disclosure Selection During a Negotiation

In a trust negotiation two entities specify policies stating the requirements the other party must satisfy to get access to a resource. Without losing generality in the following we will assume these requirements to be specific signed attributes which we also call credentials throughout the paper. The two policies must be matched in order to find out whether a negotiation exists that satisfies both parties’ policies and leads to the provision of the initially desired resource. In this paper we assume that at least one party’s policies are public. Considering both parties’ policies private increases the complexity of the problem and is left for future work since it creates new problems such as suboptimal decisions allowing for malicious entities to exploit this situation (see Section 7).

2.1 Alice Negotiates with a Book Store

This section presents a running example scenario to highlight some of the problems to be addressed during a negotiation and will be used throughout the paper to illustrate our work.

A user named Alice wants to buy a book in an on-line book store. For committing the transaction the store requires the user to register and also to specify the payment information in order to be charged with the price of the book. The store accepts several registration possibilities involving the disclosure of several different credentials. See Figure 1 for the policies of the book store.¹ Registration can

¹ For our policy examples depicted in Figure 1 and 2 we use similar notation as in [24] and include the properties of the language there defined such as monotonicity and use of propositional symbols to represent credentials. In order to improve readability of the policy, we have abused notation by creating the symbols *pregistration* and *ppayment* which must be understood as a placeholder for the right side of the clause.

$$\begin{aligned}
p_{purchase} &\leftarrow p_{register} \wedge p_{payment} \\
p_{register} &\leftarrow (c_{name} \wedge c_{bdate} \wedge (c_{email} \vee c_{pcode})) \vee \\
&\quad c_{id} \vee \\
&\quad c_{passport} \vee \\
&\quad (c_{name} \vee c_{email}) \wedge c_{id} \\
p_{payment} &\leftarrow (c_{bank_name} \wedge c_{bank_account}) \vee \\
&\quad (c_{credit_card} \wedge c_{pin}) \\
c_{bbb} &\leftarrow true \\
c_{osc} &\leftarrow true
\end{aligned}$$

Fig. 1. The book store's policies

$$\begin{aligned}
c_{name} &\leftarrow true \\
c_{birthdate} &\leftarrow c_{bbb} \\
c_{telephone} &\leftarrow c_{bbb} \\
c_{email} &\leftarrow c_{bbb} \\
c_{post_code} &\leftarrow c_{bbb} \\
c_{id} &\leftarrow c_{bbb} \\
c_{passport} &\leftarrow c_{bbb} \\
c_{bank_name} &\leftarrow c_{bbb} \wedge c_{osc} \\
c_{bank_account} &\leftarrow c_{bbb} \wedge c_{osc} \\
c_{credit_card} &\leftarrow c_{bbb} \wedge c_{osc} \\
c_{pin} &\leftarrow c_{bbb} \wedge c_{osc}
\end{aligned}$$

Fig. 2. Alice's policies

be achieved by some sort of identification of the user; this is possible by specifying the name, the date of birth² and the user's country code or her e-mail address (from which the country of registration can be extracted). Identification can also be achieved via the personal ID card number or the passport number. In addition, registration is automatically performed, if either name or e-mail address, and an ID card number are provided³. Regarding payment, the book store offers two options: either bank account information together with the bank's name or a credit card number together with a PIN must be provided. As stated before, we assume that these policies are public for any requester asking to buy a book.

Alice does not mind to share her name with anyone, but she specifies some conditions before other information about her is disclosed (see Figure 2). She is willing to provide her general personal information to any entity that is certified by the Better Business Bureau (BBB) seal program that guarantees that its members will not misuse or redistribute disclosed information. However, in order to disclose her bank account or credit card, an additional Online Security Certificate (OSC) must be disclosed, ensuring that the store website is secured and no-one else will be able to retrieve her information during the transaction.

2.2 Selecting among Possible Alternative Negotiation Paths

By matching the policies of both, the book store and Alice, one can find all possible negotiation paths, that is, all the credential disclosure sets that will make the negotiation succeed. How to extract such a negotiation path is described in [24,7]. The matching of our two policies return several possible negotiation

² Note that in some cases it is possible to apply zero-knowledge proofs (e.g., [15]) in order to avoid disclosure of information (e.g., whether a user is older than 18). This is orthogonal to our approach since we deal with situations where the actual value *must* be retrieved (e.g., name).

³ Although in this simple example it may be evident that this last possibility of registration overlaps with other policies, it may not be so evident for more complex policies or policy languages, or when the whole policy is specified by more than one single administrator.

Table 1. Disclosure sets for a successful negotiation between the book store and Alice

	1	2	3	4	5	6	7	8	9	10	11
	name	bdate	telephone	e-mail	postcode	id	passport	bank name	bank account	credit card	pin
S_1	x	x	x					x	x		
S_2	x	x	x							x	x
S_3	x	x		x				x	x		
S_4	x	x		x						x	x
S_5						x	x	x			
S_6						x				x	x
S_7							x	x	x		
S_8							x			x	x
S_9	x					x	x	x			
S_{10}	x					x				x	x
S_{11}			x	x	x	x	x				
S_{12}			x	x						x	x

paths and there exist 12 different credential disclosure sets (see Table 1) leading to a successful negotiation. Therefore, Alice has to select among 12 different possibilities that would all make the negotiation succeed. However, not all of them may be necessarily equally desirable to her. In fact, Alice may have several preferences concerning the disclosure of her credentials: she might for example prefer to disclose her ID card number instead of her passport number and she may prefer to provide her bank account instead of paying via credit card. This information is not given by her policies. In fact, the policies specify that access to resources is granted if certain conditions are satisfied but not how to decide which credential to disclose in case only k out of n conditions are required. Alice's trust agent would have to ask Alice to decide which of all the 12 alternatives she prefers to disclose. And as soon as complex policies come into play, she may be easily overloaded with too many options. Furthermore, many of these options are already overruled by other ones so the user does not even need to consider them. Therefore, Alice's preferences shall be exploited in order to rule out suboptimal negotiations. The following requirements sum up what we consider necessary to be taken into account when providing a solution to Alice:

Total vs. partial orders. It may be difficult for Alice to define a total order preference for all her credentials. First, it is time consuming, and second it may be impossible to say whether a frequent flyer card is more or less sensitive than a video club card. Moreover, it is useless to specify such a preference since it is unlikely that they will be given as an alternative to each other. Therefore, it should be possible to reason over Alice's preferences even if only a partial ordering among her credentials is available.

Preferences among more than one credential. Generally, preferences among disclosure sets of credentials (and not only among single credentials) should be allowed, too. For instance, it could be preferred to disclose the e-mail address instead of the date of birth together with the postal code (since postal code and date of birth are considered a quasi-identifier [21]).

Conditional Preferences. Contrarily to this preference, in case the date of birth is not disclosed, Alice may strongly prefer to disclose her postal code instead of her e-mail address. However, if she also has to disclose her date of birth, she would switch her preference and prefer to disclose her e-mail instead of her post code because the latter together with her date of birth is a quasi-identifier. Even more general, preferences may depend on other situational attributes such as the party the user is negotiating with. Therefore, a preference-based approach should allow for conditional preferences such as “This preference only holds if my date of birth is disclosed, too. In all other cases, I have the opposite preference.”.

Quantitative vs. qualitative preferences. It may be clear that Alice considers her ID card number less sensitive than her passport number. However, quantifying the sensitivity of a credential is difficult (e.g., $s_{id} = 10$, $s_{passport} = 11$ or $s_{id} = 10$, $s_{passport} = 51$), especially when having a large number of credentials. Furthermore, the aggregation of this quantification to sets of credentials is even more difficult: calculating the cost of disclosing *two* (or more) credentials using arbitrary quantitative aggregation methods (as they are used in [7]) is difficult to understand by users (assigning sensitivity 11 or 51 to $s_{passport}$ may have a great difference later on). Therefore, *qualitative* preferences among credentials should be allowed.

Dynamic generation of preferences. It may be impossible for Alice to provide in advance all preferences required to automatically choose a single negotiation path. In case more than one disclosure set that fulfills the negotiation remains, it is straightforward to ask Alice to deliver more preferences. However, if it is required to ask Alice, a set of possible preferences to additionally decide upon should be shown to her and her answer should be (optionally) recorded in order to avoid the same request for future negotiations. Therewith, the system should incrementally build up a knowledge base about her preferences.

Selecting the Optimum. Finally, any solution provided to Alice has to meet a trivial but very important requirement, i.e., to reduce the number of negotiations by strictly following the users preferences: any procedure should ensure that no preferred alternative is ruled out and no suboptimal disclosure set should be contained in the selected alternatives.

3 Related Work

The work in [24] introduces generic strategies, giving the user the possibility to generally specify a negotiation’s behaviour, such as termination criteria or when to disclose an unlocked credential (speeding up the negotiation with less or more

cautious strategies). However, the problem of personalization in terms of which successful negotiation path to choose among several alternatives has not been addressed.

Defining preferences in the context of trust negotiation has been described in [7] by attaching weights to credentials and comparing accumulated costs of a negotiation path. However, using numbers to express preferences has several drawbacks (as described in the previous section). In particular, assigning numbers to all credentials is complex and unintuitive for users [11] and assuming the linear aggregation of weights as a measurement for composition of objects is in our opinion undesirable; for cases like the quasi-identifier example it may even yield a lack of security. Furthermore, assigning weights implies the preference order to be a total order which is too restrictive since it may not be possible to express a preference between every two credentials (cf. for instance [9]). The same drawbacks apply to the approach of [23] where a point-based trust management model is introduced enabling the users to quantitatively distinguish the sensitivity of their credentials by assigning a certain amount of points to each credential.

A similar approach is presented in [17] where preference elicitation for negotiating agents is discussed. Case studies concerning how to acquire knowledge about the user's preferences are described. However, similar to [7], preferences are quantitatively defined, utilizing satisfaction degrees to be defined by a user.

In [5], preferences are applied in the context of policy-driven network control. The authors introduce the policy language PDDL allowing for preference definitions between possible actions preventing a hazardous network situation that may be generated by a policy. Although this approach does not tackle trust negotiation, it is related to our work since it integrates the concept of preference orders into a policy-driven behavior control. However, PDDL restricts to preferences on which action to block and it further focuses solely on total orders. It provides a leveling approach for partial orders but this leveling does not work with all sets of partial order preferences (such as the one described in our running scenario and depicted in Figure 3 later in the paper—do post code and telephone or post code and date of birth belong to the same level?).

4 Specification of Preference Relations

In order to model Alice's preference for the disclosure of certain credentials, we will now introduce the notion of qualitative preferences. As we have motivated in the scenario, the selection of a suitable negotiation path according to a user's preferences is needed for a secure and satisfactory negotiation. Picking the 'right' negotiation path is basically similar to personalization techniques used in today's database research, like for instance retrieving the best object out of a database or a digital product catalog. The notion of retrieval preferences in the context of databases and information systems has been formalized, e.g., by Kießling [12] and Chomicki [8]. To describe users' preferences in a way

exploitable for selecting optimal objects, in the following we will rely on the preference query formalization proposed by Chomicki in [8]. In this extension to relational algebra, preferences are expressed on object level as binary relations over a set of objects O with certain attributes. In our case, the objects are the sets of possible negotiations and their attributes will be the different credentials that either have to be disclosed or not.

Definition 1 (Object-Level Preference). *Let $A = \{A_1, \dots, A_n\}$ be the set of available attributes of the elements in O , and U_i , $1 \leq i \leq n$ the respective domain of possible values of A_i . Then any binary relation \succ which is a subset of $(U_1 \times \dots \times U_n) \times (U_1 \times \dots \times U_n)$ is a qualitative preference relation over the object set O .*

Typically, preference relations are not directly defined on object level. In the area of database retrieval, users usually need to explicitly provide their preferences on the attribute values of each object attribute. For example, for the attribute “model of the car” a user needs to explicitly state that she prefers a Volkswagen to a Nissan. Therefore, preferences are rather stated with respect to the attribute values of each single attribute. Certain values are preferred over others, thus forming a partial order of attribute values:

Definition 2 (Attribute-Level Preference). *Let A be the set of available attributes of the elements in O and $A_i \in A$ an attribute in such set with U_i its respective domain of possible values. The attribute level relation \succ_i , which is a subset of $U_i \times U_i$, is a qualitative preference relation over the value set of A_i .*

The extension of an attribute level preference to respective object level preferences generally follows the well-known ceteris paribus semantics [18] (“all other things being equal”). The ceteris paribus condition states that a domination relationship between two objects can only be applied, if one object dominates the other in one attribute and the remaining attributes of both object show exactly the same values. That is, if $x_i \succ_i y_i$ and $x_j = y_j (\forall j \neq i)$ then x is preferred to y .

After defining preferences with respect to each attribute, objects have to be compared considering all attributes. Therefore it is needed to combine the attribute preferences and build up an object level preference. For such a composed preference, the combined attribute level preference relations are called *dimensions* of the resulting preference relation. Two multidimensional compositions are common [8]: *lexicographic composition* combines any two dimensions by strictly considering one as more important than the other. *Pareto composition* allows to combine two preference relations without imposing a hierarchy on the dimensions: all dimensions are considered as being of equal importance.

A lexicographic composition \succ_L is based on the assumption that one relation can be considered more important than the other, i.e., there is a total ordering between all attributes. Thus, objects are generally ranked according to the more important attribute and only in case of ties the less important attribute is used for ranking.

Definition 3 (Lexicographic Composition). *Given the preference relations \succ_1, \dots, \succ_n over the attributes A_1, \dots, A_n and assuming a total order among A_1, \dots, A_n , the lexicographic composition \succ_L is defined as: $x \succ_L y \Leftrightarrow x \succ_1 y \vee (x_1 = y_1 \wedge x \succ_2 y) \vee (x_1 = y_1 \wedge x_2 = y_2 \wedge x \succ_3 y) \vee \dots \vee (x_1 = y_1 \wedge \dots \wedge x_{n-1} = y_{n-1} \wedge x \succ_n y)$.*

In contrast, Pareto composition yields a new preference relation following the fair principle of Pareto domination: an object X is said to *Pareto-dominate* another object Y iff X shows better or equal values than Y with respect to all attributes preferences and is strictly better with respect to at least one attribute preference.

Definition 4 (Pareto Composition). *Given the preference relations \succ_1, \dots, \succ_n over the attributes A_1, \dots, A_n , the Pareto composition \succ_P is defined as: $x \succ_P y \Leftrightarrow (\forall i : x \succ_i y \vee x =_i y) \wedge \exists j : x \succ_j y$.*

The evaluation of Pareto composition for database retrieval is often referred to as “skyline queries” and is a direct application of the maximum vector problem [13]. Like for the preference modeling recently a lot of research has been invested to find efficient skylining algorithms, as e.g., [6,20,2]. The two composition paradigms form the extreme cases of possible compositions: whereas a lexicographic order adheres to a strict ranking between the preferences, the Pareto composition assumes no order at all. However, for the application in trust negotiation both paradigms are problematic: by focusing on the highly preferred attributes the lexicographic order biases towards negotiations that will *not* disclose a very sensitive credential, even if they disclose all other credentials. Given the fact that the set of credentials disclosed should be kept rather small this is definitely not a desirable behavior. The Pareto composition on the other hand is too careful: by considering the disclosure of each credentials as equally problematic a lot of incomparability between different negotiations is introduced and the user has to choose between lots of possible negotiations. In fact, the result sets of Pareto compositions are known to grow exponentially with the number of dimensions (here: the number of possible credentials) [3]. Our solution to reduce the amount of incomparable disclosure sets is described in the next section: we will allow users to specify a preference order over the attributes themselves and thus distinguish between more or less preferred (i.e., sensitive) credentials.

4.1 Amalgamating Preference Relations

Specifying preferences for each single attribute yields the challenge to combine them on the object level: what if an object is better than another in terms of one attribute but worse in terms of another? In the application for negotiations this is problematic, too: a disclosure set can be more or less preferred depending on *which* credentials are contained in this disclosure set (and not only *whether* one credential is disclosed or not which is considered in the comparison following the Pareto composition). To overcome this problem, recently the concept of preference amalgamations (or trade-offs) has been proposed [1] forming a useful extension of the Pareto composition. It does not only consider all attributes

equally desirable, it additionally allows the user to specify a connection between two or more attributes. This is especially helpful, because in many practical applications, users are willing to perform trade-offs, i.e., relax their preferences in one (or a set of) attributes in favor of an object's better performance in another (set of) attributes.

Example 1. Alice may state that a negotiation where she has to disclose her credit card and not her bank account is less preferred than a negotiation where she does not disclose her credit card but her bank account. Mind that these two disclosure sets are incomparable from the Pareto composition point of view because in the bank account dimension the first is better (it does not include the disclosure of the bank account and the other does) but in the credit card dimension the second is better (the second set does not contain the credit card). Hence, Alice relaxed her preference for not disclosing her bank account instead of disclosing it in favor of the fact that the credit card is not disclosed.

Definition 5 (Amalgamated Preference). *Given the preference relations \succ_1, \dots, \succ_n over the set of attributes A_1, \dots, A_n , a set $\mu \subseteq \{1, \dots, n\}$ with cardinality k , as well as two k -dimensional tuples X_μ, Y_μ restricted to attributes with indices in μ , then (with π as the projection in the sense of relational algebra) the function $AmalPref(X_\mu, Y_\mu)$ is defined as: $AmalPref(X_\mu, Y_\mu) := \{(x, y) \mid \forall i \in \mu : (\pi_{A_i}(x) = \pi_{A_i}(X_\mu) \wedge \pi_{A_i}(y) = \pi_{A_i}(Y_\mu)) \wedge \forall j \in \{1, \dots, n\} \setminus \mu : (\pi_{A_j}(x) = \pi_{A_j}(y) \vee \pi_{A_j}(x) \succ_j \pi_{A_j}(y))\}$.*

An amalgamated preference is a relation denoted by $\succ_\mu^{X_\mu, Y_\mu}$ such that $x \succ_\mu^{X_\mu, Y_\mu} y \Leftrightarrow (x, y) \in AmalPref(X_\mu, Y_\mu)$.

In order to better understand this definition we provide a formalization of Example 1 after introducing some notation in the next section (see Example 3). In general, this definition means: given two tuples X_μ, Y_μ from the same amalgamated dimensions given in μ , the relation $\succ_\mu^{X, Y}$ is a set of pairs of the form (o_1, o_2) where the attributes of o_1 projected on the amalgamated attributes equal those of X_μ , the attributes of o_2 projected on the amalgamated dimensions equal those of Y_μ , and furthermore all other attributes (which are not within the amalgamated dimensions defined in μ) are identical in o_1 and o_2 . The last requirement again denotes the ceteris paribus condition [18], i.e., the dominated object has to show equal values with respect to all non-amalgamated attributes.

5 Preference-Based Selection of Credential Disclosure Sets

In this section we extend the preference theory from the previous section in order to handle credential disclosure sets and to find the most preferred negotiations out of the set of all possible negotiations. This process is based on qualitative preferences defined over single credentials⁴ (such as “I prefer to give my bank

⁴ “Credentials” and “signed attributes” are used interchangeably throughout the paper.

account information. My credit card number would be the second choice.”) or over sets of credentials (such as “giving my e-mail is preferred to disclosing my postal code together with my date of birth”).

5.1 Modeling Credential Disclosure Sets

Let $C = \{c_1, \dots, c_n\}$ be the set of credentials a party of a negotiation owns. The set of credentials a party has to disclose during the whole negotiation in order to succeed is a subset of C . Following the representation in Table 1, we represent a set of credentials as a bit vector with n dimensions comprising one dimension for each single credential such that setting a bit i to 1 means that during the negotiation the credential c_i is disclosed.

Definition 6 (Credential Disclosure Vector). *Let S be a credential disclosure set over the set of credentials C . The Credential Disclosure Vector representing S is the bit vector $X = (x_1, \dots, x_n)$ ($n = |C|$) such that $x_i = 1$ iff $c_i \in S$ and $x_i = 0$ otherwise.*

Example 2. In our scenario, the set of credentials Alice owns is $C = \{c_{name}, c_{bdate}, c_{phone}, c_{email}, c_{pcode}, c_{id}, c_{passp}, c_{bname}, c_{bacc}, c_{cc}, c_{pin}\}$. Mapping this set into a vector allows us to easily represent the disclosure set S_1 from Table 1 as $(1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0)$. In the following, we will assume this order as it is depicted in Table 1 for the rest of our examples.

These bit vectors represent objects (credential disclosure sets) for which each attribute dimension is the credential name and only two possible values exist: either 0 (a certain credential is not disclosed) or 1 (a credential is disclosed). In the following, we will refer to a credential disclosure set and its bit representation interchangeably.

5.2 Modeling Preferences

As we have seen so far, preference relations act on two different levels: on the object level and on the attribute level.

Object Level Preferences. Preference relations on the object level act among disclosure sets or, more precisely, on their bit vectors. These preferences are computed out of attribute level preferences given by the user. Since object level preferences cannot be easily defined by the user, preferences on the attribute level are used to build up preferences on object level.

Attribute Level Preferences. Privacy plays a main role in trust negotiations and therefore it may be assumed that a user always prefers “not to disclose” a credential in a negotiation. Therefore, for our running scenario we assume the attribute level preference $0 \succ_i 1$ for each credential c_i . However, a user may want to specify the opposite preference for some credentials in order to force the negotiation to select a negotiation path in which a specific credential is

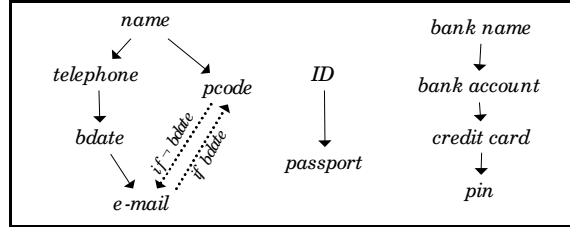


Fig. 3. Alice's preferences for the disclosure of her credentials

disclosed.⁵ Therefore, our theory allows for attribute level preferences in both directions.

The Pareto composition of the attribute level preferences allows us to compare on the object level. However, as it is described in the previous section, in our scenario we exploit amalgamated preferences, i.e., preferences crossing the border of one single attribute dimension.

Amalgamated Preferences. Amalgamated Preferences in our scenario connect two credentials with a preference relation. In Figure 3, Alice's amalgamated preferences are represented in a graphical manner.

Example 3. Returning to Example 1 and following Definition 5, we may now define the preference that the bank account is preferred to the credit card as an amalgamated preference. For defining the numbers of the different dimensions in μ we rely on the order in Table 1 (as it is done for the vectors representing the credential disclosure sets) starting with 1. Hence, the bank account's dimension is 9 and the credit card's dimension is 10: $\succ_{\{9,10\}}^{(1,0),(0,1)}$. This relation amalgamates the two dimensions 9 and 10 and allows to decide between two negotiations where in one the credit card is disclosed but the bank account is not and in the other the bank account is disclosed but the credit card is not; according to the ceteris paribus condition all dimensions except the amalgamated ones 9 and 10 have to show equal values in both negotiations.

Conditional Preferences. The notion of amalgamated preferences allows for conditional preferences. In Figure 3, conditional preferences are depicted as dotted arrows with a condition attached. Alice's preference concerning post code and email depends on whether the date of birth is additionally disclosed or not. For each value of the condition, we introduce a new amalgamated preference: for the case where date of birth is disclosed we introduce $\succ_{\{2,4,5\}}^{(1,1,0),(1,0,1)}$ and for the other case we introduce $\succ_{\{2,4,5\}}^{(0,0,1),(0,1,0)}$. This example solves the quasi-identifier problem presented in Section 2.2. However, conditions may be even more complex: they may be situational [10] and therefore include the external context

⁵ This may be the case for vouchers or discount credentials that may allow the user to receive a discount or even a free purchase when performing a transaction.

of a negotiation. For example whom one is negotiating with may play a role (e.g., one prefers to disclose the bank account to the credit card number if the requester is the bank and vice versa otherwise). These kinds of conditions can easily be modeled by our framework as additional dimensions in the vectors to be compared.

Possible Conflicts. As soon as one considers more than one preference in order to build up a concise knowledge base of all the user's preferences, one has to resolve possible conflicts between the given preferences. This is because a contradicting preference relation may lead to cycles in the object level preference and therefore does not allow for concise comparison anymore: finding the optimal object in a given set of objects becomes non-deterministic.

Example 4. One example could be that two amalgamations given by the user directly contradict, such as $\succ_{\{2,3\}}^{(1,0),(0,1)}$ and $\succ_{\{2,3\}}^{(0,1),(1,0)}$.

But although amalgamated preferences do not directly contradict, they may also conflict as soon as one considers a transitive chain as in the following example:

Example 5. Assume there already exists one amalgamated preference $\succ_{\{2,3\}}^{(1,0),(0,1)}$.

Adding the amalgamation $\succ_{\{1,2,3\}}^{(1,0,1),(0,1,0)}$ would lead to an indirect contradiction:

$(0, 1, 0) \succ_{\{2,3\}}^{(1,0),(0,1)} (0, 0, 1) \succ_1 (1, 0, 1)$ holds but this directly contradicts the amalgamation to be added which states the opposite:

$(1, 0, 1) \succ_{\{1,2,3\}}^{(1,0,1),(0,1,0)} (0, 1, 0)$.

In order to avoid possible conflicts in a set of preferences, a preference to be added to this set has to meet certain conditions:

Definition 7 (Consistent Preferences). Let O be a set of objects and $P \subseteq O^2$ a preference relation on these objects. Let further P^{conv} be the converse relation wrt. to P such that $(x, y) \in P \leftrightarrow (y, x) \in P^{conv}$. We call a preference relation $S \subseteq O^2$ consistent wrt. P iff

1. $\forall x, y \in O : (x, y) \in S \rightarrow (y, x) \notin S$ and
2. $S \cap (P \cup P^{conv}) = \emptyset$.

(We will see later (in Remark 2) that the first condition in this definition takes care of cases like the one in Example 4 and the second condition corresponds to Example 5).

Combining Preferences Transitively. Based on this condition, we are now able to consistently add preferences to our knowledge base and incrementally build up one single preference relation which we call Incremented Preference Relation. This relation includes the transitive closure of the preferences incrementally added:

Definition 8 (Incremented Preference Relation). Let O be a set of objects, $P \subseteq O^2$ a relation on these objects, and $S \subseteq O^2$ the set of object pairs representing the preference to be added to P . Let further be S consistent wrt. P . We define T as the transitive closure $T := (P \cup S)^+$. Then we define the Incremented Preference Relation of P incremented by S as $P^* := \{(x, y) \in T \mid (y, x) \notin T\}$.

5.3 Filtering out Non-preferred Disclosure Sets

In this section, we present how the theoretical basis such as Pareto composition, amalgamated preferences, and incremented preferences is used to select the non-dominated objects, i.e., the most preferred disclosure sets according to the given preferences. At the end of this section we will develop the preference relation \succ which allows us to compare any two disclosure sets according to a set of preferences given by a user. Based on this relation, we will provide a formal definition of the set of optimal disclosure sets.

The following example shows how Pareto dominance helps to rule out non-preferred disclosure sets in our running scenario:

Example 6. Given the two disclosure sets S_6 and S_{10} from Table 1, $S_6 = (0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1)$ and $S_{10} = (1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1)$, and the set of attribute level preference relations over $c_i \{0 \succ_{c_{name}} 1, \dots, 0 \succ_{c_{pin}} 1\}$ it is obvious to infer that $S_6 \succ S_{10}$ holds since $S_6 \succ_1 S_{10}$ and $S_6 =_i S_{10}$ for ($2 \leq i \leq 11$). In any case, it is possible to automatically infer that a user always prefers to disclose only a subset of credentials, i.e., only ID card number and bank account.

Remark 1. We want to point out that this holds since we are considering the preferences $\neg c_i \succ_i c_i$ on attribute level. Given these preferences over the binary value space for attributes, computation of Pareto domination is reduced to simple set containment checking.

The following example shows how taking the amalgamated preferences into account prunes out additional dominated objects:

Example 7. Given the sets (from Table 1) $S_5 = (0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0)$ and $S_7 = (0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0)$, and the amalgamated preference $\succ_{\{6,7\}}^{(1,0),(0,1)}$ (that is, it is preferred to disclose the ID card number to the passport number) it is possible to infer that $S_5 \succ S_7$. Therefore, it is possible to automatically infer that the user would prefer to disclose her ID number *and* bank information instead of disclosing her passport number *and* bank information.

Pareto composition (in Example 6) and preference amalgamation (in Example 7) applied in isolation to filter out dominated disclosure sets is not sufficient in some cases—as it is shown in the following example:

Example 8. S_5 and S_6 cannot be compared with the mechanisms given so far: there is no way to combine Alice’s amalgamated preference concerning bank name and credit card and her amalgamated preference concerning bank account and pin.

This example motivates the exploitation of incremented preferences (see Definition 8) in order to further reduce the number of disclosure sets. According to Definition 5, only one amalgamated preference is applied at a time; for the remaining dimensions the ceteris paribus condition fires. However, it may be possible that several preferences—be it simple attribute level preferences or amalgamated ones—need to be applied over the same two disclosure sets in order to

compare them. Therefore, we exploit the transitive combination of preferences as it is delivered by the concept of the incremented preference relation. The following example shows how an incremented preference relation enables us to compare the two disclosure sets from Example 8.

Example 9. In order to compare S_5 and S_6 we need to combine two of Alice's preferences; for example her preference saying that bank name is preferred to credit card ($p_1 = \succ_{\{8,10\}}^{(1,0),(0,1)}$) and her preference saying that bank account is preferred to pin ($p_2 = \succ_{\{9,11\}}^{(1,0),(0,1)}$). To achieve this combination we apply the definition of an incremental preference relation (see Definition 8) in the following way. To an initially empty incremented preference P_0^* we first add p_1 . From the definition of amalgamated preferences (Definition 5) and from the fact that P_0^* is empty, it is obvious that p_1 is consistent wrt. P_0^* . Adding p_1 yields the incremented preference $P_1^* = p_1$. In the second step, we add p_2 to P_1^* in order to construct the incremental preference P_2^* . Similarly to the first step, p_2 is consistent wrt. P_1^* . This is due to the ceteris paribus condition in the definition of amalgamated preference: for any pair in p_1 the dimensions 9 and 10 are equal where for any pair in p_2 they are different. Therefore, no pair in p_1 contradicts a pair in p_2 . By applying the transitive closure in order to construct P_2^* the pair (S_5, S_6) is introduced as an element of P_2^* . This is due to the transitive chain built from the following two pairs: in p_1 we have $(S_5, (0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0))$ and p_2 states $((0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0), S_6)$. By transitively combining p_1 and p_2 to P_2^* we get $(S_5, S_6) \in P_2^*$.

This example forms a base case and provides evidences that the extension of an isolated application of the Pareto domination and the amalgamated preferences given by the user is needed. In order to provide *all possible combinations* of all given preferences, we need to build up an incremented preference relation which we call Complete Preference Relation \succ :

Definition 9 (Complete Preference Relation). *Let \succ_P be the Pareto composition of attribute level preferences \succ_1, \dots, \succ_n . Let further be $P = \{p_1, \dots, p_m\}$ a set of amalgamated preferences. The Complete Preference Relation \succ is defined as the incremented preference relation of \succ_P incremented by $(\bigcup P)$.*

Remark 2. Because of the conditions in Definitions 7 and 8, this includes two implicit requirements:

- 1) each amalgamated preferences must not contradict another, i.e., $p_n \cap p_m^{conv} = \emptyset$ holds for each n, m . This case matches Example 4 and condition 1 in Definition 7; it implies that our framework requires user-given amalgamated preferences not to directly contradict each other.
- 2) the union of the amalgamated preferences has to be consistent wrt. the Pareto composition. This restriction ensures that the incremented preference relation does not contain any cycles [1]. Instantiated for our scenario, this restriction requires that no amalgamated preference stated by the user contradicts the Pareto domination (i.e., each amalgamated preference has to be consistent wrt. the

Pareto composition of the attribute level preferences). This requirement corresponds to Example 5 and to condition 2 in Definition 7.

Further, these two requirements imply that adding new preferences to the incremented preference relation is a monotonic process [1]: adding new preferences will never make former comparisons invalid, it will always add comparable pairs and never remove some. This is in particular helpful for the preference elicitation process as is allowed for in our implementation (see Section 6): after adding new elicited preferences it is not needed to recompute the whole set of optimal disclosure sets but to simply remove the new dominated objects from the set.

Remark 3. In the theory developed in this paper we do not consider equivalence preferences such as “disclosing my ID card number is as equally preferred as disclosing my passport number”. They can easily be introduced according to [1]; due to space conditions we left them out in this paper.

The incremented preference relation forms the basis for our object level preference relation. It is clear from its definition, that it comprises the Pareto dominance relation as well all single amalgamated preference relations given by the user. It additionally contains all combinations of these base preferences. Therefore, it enables us to filter out all the disclosure sets which are not preferred:

Definition 10 (Optimal Disclosure Sets). *Let O be a set of disclosure sets and \succ a complete preference relation. We define the set of optimal disclosure sets O_{\succ} as follows: $O_{\succ} := \{o \in O \mid \nexists o' \in O : o' \succ o\}$.*

5.4 Revisiting the Scenario

In this section we show on our running scenario how the techniques and concepts defined in previous sections may be used to find out the optimal negotiations for Alice given the disclosure sets S_1, \dots, S_{12} yielding a successful negotiation. As we have shown above, the incremented preference relation contains the Pareto composed attribute level preferences as well as the amalgamated preference. However, in order to see the improvement of each preference concept introduced in the paper, we will divide the process of ruling out dominated disclosure sets into three steps: we show (A) how objects are ruled out by simple Pareto composition, (B) how objects are ruled out by single amalgamated preferences, and (C) how the transitive combination in the incremented preference relation rules out the remaining dominated objects.

Pareto Composition. Using the attribute level preferences $0 \succ_i 1$ (as described in Section 5.2), we can apply Pareto composition to remove dominated sets. From Pareto domination we conclude that disclosure set S_{10} can be removed since it is dominated by the set S_6 (all dimensions are equally good in S_6 and in S_{10} except dimension 1 (c_{name}) in which S_6 is preferred to S_{10}). This may be considered straightforward since S_{10} additionally requires Alice to disclose c_{name} which is an unnecessary disclosure. In addition, also S_9 can be removed since it is dominated by S_5 . Furthermore, S_6 Pareto dominates S_{12} and S_5 Pareto

dominates S_{11} . Hence, simple Pareto composition is able to filter out already four dominated disclosure sets, namely, S_9 , S_{10} , S_{11} , and S_{12} .

Amalgamated Preferences. In addition to Pareto composition, we may exploit the amalgamated preferences specified by Alice to further reduce the number of disclosure sets. From Alice’s preference for disclosing her ID card number instead of her passport number (amalgamated preference $\succ_{\{6,7\}}^{(1,0),(0,1)}$), we can conclude that S_7 is dominated by S_5 and that S_8 is dominated by S_6 . Furthermore, Alice has an amalgamated preference over three dimensions because of her fear of being quasi-identified: $\succ_{\{2,4,5\}}^{(1,1,0),(1,0,1)}$. From this preference we may infer that S_2 dominates S_4 and S_1 dominates S_3 . This way it is possible to remove yet another four dominated disclosure sets, namely S_3 , S_4 , S_7 , and S_8 .

Transitive Combination of Preferences. After having performed the previous steps, the remaining disclosure sets are S_1 , S_2 , S_5 , and S_6 . Among the inferred preferences, there is the preference transitively combined out of the preferences ‘bank name is preferred to credit card’ ($\succ_{\{8,10\}}^{(1,0),(0,1)}$) and ‘bank account is preferred to pin’ ($\succ_{\{9,11\}}^{(1,0),(0,1)}$). Applying both preferences transitively enables us to rule out S_2 because it is dominated by S_1 and S_6 because it is dominated by S_5 . (as it is shown in Example 9).

The procedure described in this section is able to filter out ten non-preferred credential disclosure sets based on qualitative preferences, therefore facilitating Alice’s interaction with her negotiation agent. However, two disclosure sets are still remaining: S_1 and S_5 . Both are not comparable according to the specified preferences. In the next section we give an idea of how a system could elicit which disclosure set she prefers.

6 Implementation and Experiments

We implemented a prototype computing preferred credential disclosure sets given a set of successful negotiation paths and a set of preferences specifying which credential’s disclosure is preferred to another’s. The user interface is web-based and the logical core of our approach is implemented in Prolog. Given a set of credentials a connected policy engine provides all disclosure sets yielding a successful negotiation. From this set the Prolog engine computes the non-dominated and therefore preferred disclosure sets. In case there exists one single non-dominated disclosure set, this one will be used automatically. Otherwise, the user can either choose one disclosure set or go for an iterative preference elicitation process. If the elicitation process is selected, the preference information the user provides after selecting one of the shown options is exploited to further reduce the set of non-dominated disclosure sets (see Remark 2 about monotonicity). Depending on the new result, the user has again the possibility to either choose one set of credentials to be disclosed or to perform another elicitation process. This process continues iteratively until the user either directly selects the disclosure set to use or until any set but one is filtered out.

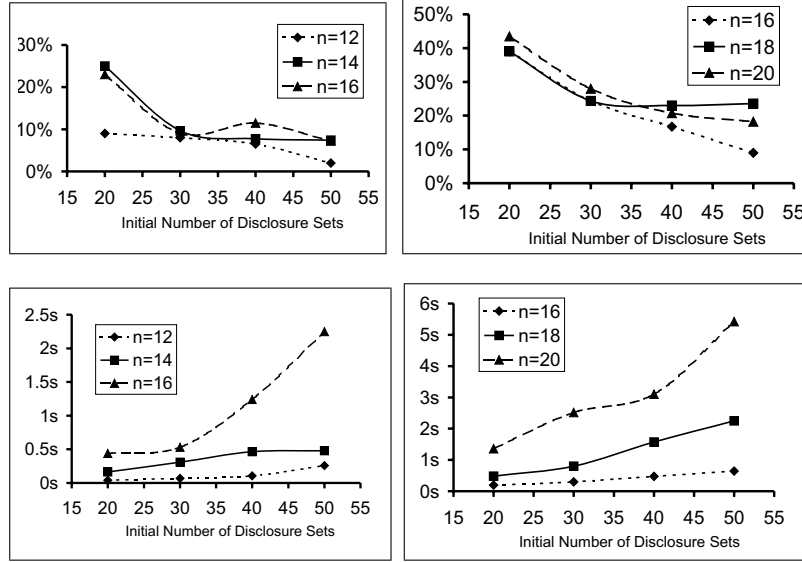


Fig. 4. Optimal Disclosure Sets (top) and Computation Time (bottom) for $k = 5$ (left) and $k = 7$ (right). n is the number of credentials and k is the maximal number of credentials that can be disclosed during a negotiation.

We tested the efficiency of our approach in terms of how many disclosure sets are ruled out. This amount is crucial since our approach is only effective if a considerable set of suboptimal negotiations is actually ruled out. Since there is no trust negotiation benchmark available which may serve for our experiments, nor any available real data about disclosure sets (due to high sensitivity even if anonymized), we used generated data as follows:

In a trust negotiation, typically only one client credential of a certain type is needed. E.g., it is either required to disclose the passport number or the ID card number, rather than both. Therefore we partition the set of client credentials C into k disjoint subsets T called *credential types*. Further, we assume that for the success of a negotiation exactly one credential for each credential type has to be disclosed. Following this, it is obvious that each type should at least contain two credentials—otherwise this single credential will be contained in each disclosure set and will be disregarded in the comparison process. Further, for each credential type we assume a totally ordered preference on the client side. Therefore, for each run of the experiment we performed the following steps: **1)** randomly create k credential types T_1, \dots, T_k for the n credentials. **2)** create a set of amalgamations such that for each type T_i we add $\succ_{j,j+1}^{(1,0),(0,1)}$ for all $c_j, c_{j+1} \in T_i$. **3)** create a set O of n disclosure sets such that each disclosure set contains one randomly chosen credential for each credential type. **4)** rule out all dominated disclosure sets following the definition of \succ . In this setting the following parameters varied:

Number of Credentials n . The higher the number of credentials the higher the number of dimensions of the vectors to be compared. Increasing this value yields an increasing of incomparable pairs in the set of disclosure sets. For our experiments we assume a number of credentials between 12 and 20.

Number of Credential Types k . This number actually determines how many credentials are definitely disclosed in each negotiation. Increasing this value also decreases the amount of suboptimal objects because the probability an object is ‘bad’ in one credential type but ‘good’ in another becomes higher which leads to more incomparable pairs. We selected 5 and 7 credential types to cover negotiations where exactly 5 or 7 credentials *have* to be exchanged in each negotiation which we consider realistic.

Number of Disclosure Sets. Represents how many different negotiation paths exist and would have been shown to the user for selection, if no preference filtering was applied.

We made several experiments with varying values. For each setting we did 20 runs; see the upper graphs in Figure 4 for the results of some representative settings. It turned out that the average percentage of optimal disclosure sets is 18.3%. Hence, on average 81,7% of the alternatives a user has to chose from are ruled out. This is a huge improvement: instead of, e.g., 30 disclosure sets with obviously suboptimal alternatives only 6 sets are shown to the user. Moreover, removing the 24 dominated possibilities is needed from a privacy point of view: if the user selected one of the suboptimal negotiations, she would definitely disclose more sensitive information which she would not have to disclose to make the negotiation succeed. These results show that for a relatively big number of possible disclosure sets (a higher number will rarely occur in reality), our approach filters out a considerable subset of suboptimal alternatives and therefore reduces the work load for the user. Furthermore, it turned out that for our experimental setting the average computation time was about 2.3 seconds (see Figure 4). Our implementation does not consider any of the numerous optimization strategies available for skyline computation (e.g., [20]) which would make the computation even faster.

The modeling of trust negotiation in our experiments has several simplifications compared to a real scenario. We do not consider redundant policies, hence, Pareto domination was never applied in our scenario which would even lead to a higher number of suboptimal disclosure sets. Furthermore, in our experiment no preferences among different credential types are given which one can assume in a realistic setting and therefore more objects would be ruled out according to these preferences.

7 Conclusions and Further Work

Selection of credential disclosure sets in trust negotiations is a difficult task. Only allowing for total orders over the set of owned credentials or linear aggregation is undesirable. In this paper we have presented an approach based on quali-

tative partial order preferences including preference composition, amalgamated preferences, and conditional preferences and demonstrated with the help of an example how this approach can be used to solve the problem of finding optimal disclosure sets. Furthermore, an iterative process in order to allow the user to specify new preferences has been sketched and a web-based implementation of the whole system has been developed in order to test the results. Our solution eases the task of selecting among possibly many alternatives in a trust negotiation by exploiting the user's preferences. Although we applied our preference framework in the realm of trust negotiation, this preference scheme can be used in any other domain where objects need to be compared or selected based on preferences specified on the objects' attributes. The principle of preference based selection (or skylining) is one of the most promising current research fields in database retrieval.

The paper focuses on negotiations in which one party's policies are public and therefore the whole set of credentials to be disclosed may be precomputed. It is required to further study the implications of using qualitative preferences on negotiations with *private* policies, therefore assuming that possibly suboptimal decisions can be taken in different steps. This may even be exploited by malicious entities forcing the other party to disclose more credentials than needed for a successful negotiation (known as "need-to-know attacks" [19]). We are currently investigating the extensions and restrictions required to extend the contribution of this paper to such a scenario. We are also planning to further evaluate our approach. This will also include a a user study using our current implementation.

References

1. Balke, W.-T., Güntzer, U., Lofi, C.: Incremental trade-off management for preference based queries. *International Journal of Computer Science and Applications (IJCSA)* 4(1) (2007)
2. Balke, W.-T., Zheng, J., Güntzer, U.: Efficient distributed skylining for web information systems. In: Bertino, E., Christodoulakis, S., Plexousakis, D., Christophides, V., Koubarakis, M., Böhm, K., Ferrari, E. (eds.) *EDBT 2004*. LNCS, vol. 2992. Springer, Heidelberg (2004)
3. Bentley, J., Kung, H., Schkolnick, M., Thompson, C.: On the average number of maxima in a set of vectors and applications. *Journal of the ACM (JACM)* 25(4) (1978)
4. Bertino, E., Ferrari, E., Squicciarini, A.C.: Trust-x: A peer-to-peer framework for trust establishment. *IEEE Trans. Knowl. Data Eng.* 16(7) (2004)
5. Bertino, E., Mileo, A., Proveti, A.: PDL with preferences. In: *POLICY 2005*. IEEE Computer Society, Los Alamitos (2005)
6. Borzsonyi, S., Kossmann, D., Stocker, K.: The skyline operator. In: *International Conference on Data Engineering, Heidelberg, Germany* (2001)
7. Chen, W., Clarke, L., Kurose, J., Towsley, D.: Optimizing costsensitive trust-negotiation protocols. In: *Annual Joint Conference of the IEEE Computer and Communications Societies* (2005)
8. Chomicki, J.: Preference formulas in relational queries. *ACM Trans. Database Syst.* 28(4), 427–466 (2003)

9. Fishburn, P.: Preference structures and their numerical representations. *Theoretical Computer Science* 217, 359–383 (1999)
10. Holland, S., Kießling, W.: Situated preferences and preference repositories for personalized database applications. In: ER, pp. 511–523 (2004)
11. Keeney, R., Raiffa, H.: *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*. Wiley, Chichester (1976)
12. Kießling, W.: Foundations of preferences in database systems. In: International Conference on Very Large Data Bases, Hong Kong, China (2002)
13. Kung, H., Luccio, F., Preparata, F.: On finding the maxima of a set of vectors. *Journal of the ACM (JACM)* 22(4) (1975)
14. Lee, A.J., Winslett, M., Basney, J., Welch, V.: Traust: a trust negotiation-based authorization service for open systems. In: SACMAT, ACM Press, New York (2006)
15. Li, J., Li, N.: Oacerts: Oblivious attribute certificates. *IEEE Trans. Dependable Sec. Comput.* 3(4) (2006)
16. Li, N., Mitchell, J.C.: Rt: A role-based trust-management framework. In: DISCEX (April 2003)
17. Luo, X., Jennings, N.R., Shadbolt, N.: Knowledge-based acquisition of tradeoff preferences for negotiating agents. In: International Conference on Electronic Commerce. ACM Press, New York (2003)
18. McGeachie, M., Doyle, J.: Efficient utility functions for ceteris paribus preferences. In: Conference on Artificial Intelligence and Conference on Innovative Applications of Artificial Intelligence, Edmonton, Canada (2002)
19. Olson, L.E., Rosulek, M.J., Winslett, M.: Harvesting credentials in trust negotiation as an honest-but-curious adversary. In: Workshop on Privacy in electronic society. ACM Press, New York (2007)
20. Papadias, D., Tao, Y., Fu, G., Seeger, B.: An optimal and progressive algorithm for skyline queries. In: ACM SIGMOD, San Diego, CA, USA (2003)
21. Sweeney, L.: Guaranteeing anonymity when sharing medical data, the datafly system. *Journal of the American Medical Informatics Association* (1997)
22. Winsborough, W.H., Seamons, K.E., Jones, V.E.: Automated trust negotiation. In: DARPA Information Survivability Conference and Exposition. IEEE Press, Los Alamitos (2000)
23. Yao, D., Frikken, K., Atallah, M., Tamassia, R.: Point-based trust: Define how much privacy is worth. In: Ning, P., Qing, S., Li, N. (eds.) ICICS 2006. LNCS, vol. 4307, Springer, Heidelberg (2006)
24. Yu, T., Winslett, M., Seamons, K.E.: Interoperable strategies in automated trust negotiation. In: CCS (2001)