Personalizing Your Web Services with Constructive DL Reasoning Join

Freddy Lécué
The University of Manchester, Manchester, UK
Booth Street East, Manchester, UK
{(firstname.lastname)@manchester.ac.uk}

Abstract

Nowadays web users have clearly expressed their wishes to receive and interact with personalized services directly. However, existing approaches, largely syntactic content-based, fail to provide robust, accurate and useful personalized services to its users. Towards such an issue, the semantic web provides technologies to annotate and match services' descriptions with users' features, interests and preferences, thus allowing for more efficient access to services and more generally information. The aim of our work, part of service personalization, is on automated instantiation of services which is crucial for advanced usability i.e., how to prepare and present services ready to be executed while limiting useless interactions with users? We introduce the constructive Description Logics reasoning *join* and couple it with concept abduction to i) identify useful parts of users profiles that satisfy services requirements and ii) compute the description required by a service to be executed but not provided by users profiles.

Introduction

Personalization in web-based applications (Mobasher, Cooley, and Srivastava 2000), as a global tendency nowadays, aims at alleviating the burden of information overload by tailoring the information presented based on an individual and immediate user's needs. Among numerous examples that can be found all across the web, we can highlight the proliferation of personalized home web sites such as iGoogle (http://www.google.com/ig), but also the fact that many other web applications of different kinds treat user configuration as one of their most prominent characteristics. From collaborative- (Goldberg et al. 1992), content- (Lieberman 1995) to hybrid-based, various personalization techniques have been introduced, depending on the data they manipulate and the level of personalization. The user profile, as a collection of data modelling the user extended with its interests and preferences are prominent elements to ensure accurate and efficient personalized access to information.

In recent years, web service, as an emergent technology to consume information on the web, has benefited from research progress in web personalization. The possibility to customize their results (Baumgartner, Henze, and Herzog 2005) give the users the chance to experience those services in a personalized fashion, which is prominent to permit the users to fulfil their desires more suitably. However most approaches ensure personalization by collecting and analyzing syntactic content of user profile and services description e.g., (Blake and Nowlan 2007), which limits both accuracy and automation of personalization. The *semantic web*, through its *Web Ontology Language* OWL (Smith, Welty, and McGuinness 2004) and its Description Logics (DLs) (Baader and Nutt 2003) formalism, provides many advantages over the current "formatting only" version of the web, its services and users. By tagging the semantic content of the information using machine-processable languages, services with their functionalities and user profiles with their interests, preferences can be both enhanced.

To reach the goal of more accurate and automated personalization we benefit from DL reasoning on these descriptions. We address service instantiation, part of personalization which is crucial for advanced usability, aims at preparing and presenting pre-selected services ready to be executed while limiting useless interactions with users. To this end, execution-time constraints attached to services descriptions are required to be satisfied before their execution. Most approaches (Baumgartner, Henze, and Herzog 2005) undervalue this issue and rarely consider suitable and efficient methods to address this level of personalization. Indeed information which is required by (input parameters of) services to be executed is manually collected from the users. On the contrary, we consider its automation and the personalized presentation of services to users through their semantic instantiation. Thus i) improving and easing the user interaction with services by anticipating her needs will be beneficial for both parties. We define a framework, where standard DL reasoning (Li and Horrocks 2003) and concept abduction (Noia et al. 2003), extended with constructive reasoning *join* are applied to i) adapt services to users by identifying useful parts of their profile that satisfy service requirements (i.e., inputs) and ii) compute the descriptions required by a service to be executed but not provided by users profiles.

Section 2 briefly reviews i) concept abduction, ii) web services and iii) user profiles using DLs. Section 3 introduces constructive reasoning *join*. Section 4 presents our personalization approach. Section 5 reports some experiment results regarding its scalability. Section 6 briefly comments on related work. Finally Section 7 draws some conclusions.

Background

The model we consider to represent semantics of services and user profiles is provided by an ontology. We focus on DL as a formal knowledge representation language to define ontologies since the latter offers good reasoning support for most of its expressive families and compatibility to current W3C standards e.g., OWL. We illustrated our work with the DL $\mathcal{ALC}$ (Baader and Nutt 2003) (where satisfiability, subsumption are decidable), which is a basic DL from which comparisons with other varieties can be made, to perform reasoning-based personalization. Adaptations to more expressive DLs e.g., $\mathcal{SHOIQ}$ (OWL-DL) are possible.

$$
\begin{array}{l}
British \equiv Person \sqcap \exists hasSpokenLanguage.English \\
Name \equiv \exists hasFN.FirstName \sqcap \exists hasLN.LastName \\
BusinessAccount \equiv Account \sqcap \exists hasID.OpenID \sqcap \\
\qquad \exists hasSocialNet.SocialNetAccount \\
PersonalAccount \equiv Account \sqcap \exists hasID.ElectronicID \sqcap \\
\qquad \exists hasSocialNet.LinkedIn \\
SkypeAccount \sqsubseteq Account,\ BankAccount \sqsubseteq Account \\
LinkedIn \sqsubseteq SocialNetAccount \\
SocialNetAccount \sqsubseteq Account \\
OpenID \sqsubseteq ElectronicID \sqsubseteq Account \\
GMail \sqsubseteq MaillingAddress \sqsubseteq OpenID
\end{array}
$$

Figure 1: Sample of an $\mathcal{ALC}$ Domain Ontology $\mathcal{T}$.

Besides standard reasoning satisfiability or subsumption to guarantee consistency of knowledge bases, (Noia et al. 2003) suggest computing *abduction* (Definition 1) between concepts C and D, representing what is underspecified in D to completely satisfy C in an ontology $\mathcal{T}$ (e.g., Figure 1).

Definition 1. ***(Concept Abduction Problem - CAP)*** *Let $\mathcal{L}$ be a DL, C, D be two concepts in $\mathcal{L}$, $\mathcal{T}$ be a set of axioms in $\mathcal{L}$. A CAP i.e., $C \backslash D$ consists in finding a concept $B \in \mathcal{L}$ such that $\mathcal{T} \not\models D \sqcap B \equiv \bot$, and $\mathcal{T} \models D \sqcap B \sqsubseteq C$.*

Example 1. ***(CAP and Missing Description)*** *Let C and D be two $\mathcal{ALC}$ descriptions respectively defined by $BusinessAccount \sqcap \exists hasSkype.SkypeAccount$ and $PersonalAccount$. The missing description B required by D to satisfy C* (be subsumed by) *C, denoted by $C \backslash D$ is* (1)*. D needs an $OpenID$ and a $SkypeAccount$ to satisfy C.*

$$\exists hasID.OpenID \sqcap \exists hasSkype.SkypeAccount \quad (1)$$

Semantic Web Services

Semantics of web services can be expressed by their process level (Pistore et al. 2005) (i.e., internal and complex behaviours), causal level (McIlraith and Son 2002) (i.e., preconditions and effects) and functional level (i.e., simple interface) descriptions. We focus on the latter level and more generally their functional input and output parameters, which are prominent to personalize and execute any service. Enhancing with DL concepts that determine their semantics, semantic web services S can be expressed as:

$$S \doteq \exists requires.Input \sqcap \exists returns.Output \quad (2)$$

(2) confines a service to being anything that $requires$ some $Input$s to be processed and $returns$ some $Output$s. Both parameters are supposed to be satisfiable in $\mathcal{T}$. OWL-S profile (Ankolenkar et al. 2004) or SA-WSDL (Kopecký et al. 2007) can be used to describe this functional level. For the sake of clarity, we assume services without open preconditions. However (2) can be extended along (Pistore et al. 2005; McIlraith and Son 2002) without loss of generality.

Example 2. ***(Semantic Web Service)*** *Suppose service S_1, formalized in* (3)*, locating friends and colleagues of a person. Starting from a* `FirstName`*,* `LastName`*,* `BusinessAccount`*,* `SkypeAccount` *and a* `GMail` *address of this person, S_1 returns the list of her nearby* `ContactPersons` *according to her* `Location`*.*

$$S_1 \doteq \exists requires.C^1_{S_1} \sqcap \exists requires.C^2_{S_1} \sqcap \exists requires.C^3_{S_1} \sqcap \exists requires.C^4_{S_1} \sqcap \exists returns.ContactPerson \quad (3)$$

where conjuncts $C^{i,1\leq i\leq 4}_{S_1}$ (described in Figure 1) are:

$$C^1_{S_1} \doteq \exists hasFN.FirstName \sqcap \exists hasLN.LastName \quad (4)$$

$$C^2_{S_1} \doteq BusinessAccount \sqcap \exists hasSkype.SkypeAccount \quad (5)$$

$$C^3_{S_1} \doteq MaillingAddress \sqcap \exists hasMail.GMail \quad (6)$$

$$C^4_{S_1} \doteq Location \sqcap \exists hasLat.Latitude \sqcap \exists hasLong.Longitude \quad (7)$$

Semantic User Profile

User profiles can be expressed along i) identity and possessions, ii) preferences (Kleemann and Sinner 2005), iii) skills (Calì et al. 2004), iv) behaviour and v) knowledge or beliefs. Descriptions from (ii) to (v) such as the user's interests and preferences are mainly relevant information for prioritizing services in a discovery process rather than making services and their (input) parameters adapted to the user. Therefore, a personalized access to services requires profile P descriptions (i), modelled using DLs in (8).

$$P \doteq \bigsqcap_i \exists hasInfo.Info_i \bigsqcap_j \forall hasInfo.(\neg NonInfo_j) \quad (8)$$

While descriptions $Info_i$ identify users, $NonInfo_j$ refer to information users are not inclined to provide e.g., sensitive data such as medical record. Information in (8) is collected from a short questionnaire but can be also automatically generated from (Weiß et al. 2008).

Example 3. ***(Semantic User Profile)*** *Let P_1 be a* `British` *lady identified by her* `Name`*,* `ElectronicID`*, member of a social network* `LinkedIn`*, without unauthorized access to her* `Bank Account`*.*

$$P_1 \doteq Female \sqcap \exists hasInfo.C^1_{P_1} \sqcap \exists hasInfo.C^2_{P_1} \sqcap \forall hasInfo.C^3_{P_1} \quad (9)$$

where conjuncts $C^{i,1\leq i\leq 3}_{P_1}$ (described in Figure 1) are:

$$C^1_{P_1} \doteq Person \sqcap (\exists hasName.Name) \sqcap \exists hasNationality.British \quad (10)$$

$$C^2_{P_1} \doteq Account \sqcap (\exists hasID.ElectronicID) \sqcap \exists hasSocialNet.LinkedIn \quad (11)$$

$$C^3_{P_1} \doteq \neg BankAccount \sqcup \neg(\exists hasBank.BankID) \quad (12)$$

User profile (8) can be adapted with straightforward modifications and extensions (e.g., roles) following (Calì et al. 2004) e.g., with extra roles such as $hasInterest$, $hasPreference$ (Kleemann and Sinner 2005).

(2), (8) accept an instance (or a literal value) that is of the DL type (e.g., `first.last@gmail.com` for the `GMail` type in (6)). Personalization is then inherently about concrete values and identification of input (of services) concept instances (i.e., service instance) not DL descriptions. However formalization of services (2) and user profiles (8) in DLs is required to compare them at semantic level, and then achieve their instantiation with concrete values.

Joining User Profile to Service Descriptions

Since standard DL reasoning are interesting in checking some property (true or false) such as subsumption and satisfiability, it is straightforward to check whether a profile P (8) is compatible (Li and Horrocks 2003) under subsumption $\sqsubseteq_{\mathcal{T}}$ with a service S (2). To this end a role hierarchy between $requires$ and $hasInfo$ roles of S and P is required:

$$\mathcal{T} \models hasInfo \sqsubseteq requires \tag{13}$$

However, exhibiting the description which properly ensures P to be compatible with S cannot be inferred from the latter reasoning. Indeed, only implicit proof of its existence can be derived. Hence we describe concept *join* to explain in details which description in P could ensure a compatibility with S, hence an adaptation to S and its personalization.

Example 4. ***(Compatible Descriptions)***
According to Examples 2, 3, and standard DL reasoning $\mathcal{T} \not\models C^2_{P_1} \sqcap C^2_{S_1} \sqsubseteq \bot$. However which description ensure a compatibility between $C^2_{P_1}$ and $C^2_{S_1}$?

Concept Join

As we are interested in description parts of P which ensure P and S to be compatible, we want to extract J (for Join) from P such that $J \sqsubseteq S$ remains true in $\mathcal{T}$ (Definition 2) Clearly, service personalization is not interested in descriptions D (for Discarded) such that $P \equiv D \sqcap J$ which cannot be used to instantiate inputs of S under $\sqsubseteq_{\mathcal{T}}$. Indeed such descriptions are not information required by S to be executed.

Definition 2. ***(Concept Join)***
Let $\mathcal{L}$ be a DL, P, S be two concepts in $\mathcal{L}$, and $\mathcal{T}$ be a set of axioms in $\mathcal{L}$ such that $\mathcal{T} \not\models P \sqcap S \sqsubseteq \bot$. A Concept Join Problem, denoted as $CJP\langle \mathcal{L}, P, S, \mathcal{T}\rangle$ (shortly $P \bowtie S$) is finding a pair of concepts $\langle D, J\rangle \in \mathcal{L} \times \mathcal{L}$ such that i) $\mathcal{T} \models P \equiv D \sqcap J$ and ii) $\mathcal{T} \models J \sqsubseteq S$. Then J (or $\bowtie_J$) is a join between P and S in $\mathcal{T}$.

We use $\mathcal{P}$ as a symbol for a CJP and we denote with $SOLCJP(\mathcal{P})$ the set of all solutions of the form $\langle D, J\rangle$ to a CJP $\mathcal{P}$. In case $P \sqsubseteq S$ in $\mathcal{T}$, it is straightforward to prove that $\langle \top, P\rangle \in SOLCJP(\mathcal{P})$. In case $S \sqsubset P$, a CJP has no solution because of unsatisfied condition (ii).

Proposition 1. ***(Concept Join Complexity)***
Let $\mathcal{P} = \langle \mathcal{L}, P, S, \mathcal{T}\rangle$ be a CJP. If concept subsumption with respect to a $\mathcal{T}$ in $\mathcal{L}$ is a problem $\mathcal{C}$-hard for a complexity class $\mathcal{C}$, then deciding whether a concept belongs to $SOLCJP(\mathcal{P})$ is $\mathcal{C}$-hard.

Proof. Since $\mathcal{T} \models P \sqsubseteq S$ iff $\langle \top, P\rangle \in SOLCJP(\mathcal{P})$, such a problem is $\mathcal{C}$-hard. □

Therefore, in our $\mathcal{ALC}$ context, deciding whether a concept belongs to $SOLCJP(\mathcal{P})$ is EXPTIME-hard (Baader and Nutt 2003) for a general $\mathcal{T}$.

Maximality

While (i), (ii) in Definition 2 ensure to extract descriptions from P which are in S, no condition ensures the solution to be "*maximal*" under $\sqsubseteq_{\mathcal{T}}$ i.e., obtaining the closest description to S (i.e., the most general) with respect to P. In this regard, Definition 2 needs to be extended with the $\sqsubseteq_{\mathcal{T}}$-maximality condition iii) $\forall J' : \mathcal{T} \models P \equiv D' \sqcap J' \wedge \mathcal{T} \models J' \sqsubseteq S \Rightarrow J' \sqsubseteq J$.

There is always the trivial solution $\langle \top, P\rangle$ to a CJP when $P \equiv S$. This solution corresponds to an ideal personalization, where all descriptions in P are required to instantiated all inputs in S. On the other hand, when $P \sqsubset S$ in $\mathcal{T}$, this trivial solution satisfies (i) and (ii). Such a solution does not necessarily meet the "*maximaility*" criterion. Indeed a closer description from P to S under $\sqsubseteq_{\mathcal{T}}$ can be computed. In case the $\sqsubseteq_{\mathcal{T}}$-maximality condition (iii) is met by the trivial solution, P is required to fully instantiate S but P is more specific than descriptions required by S. In the case of $\langle D, S\rangle$ as solution, some useless descriptions D in P are discarded to properly instantiate S with J (from P).

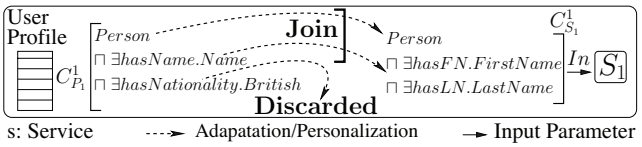

Figure 2: Concept Join.

Example 5. ***(Concept Join)***
The join J of $C^1_{P_1}$ and $C^1_{S_1}$ is defined by $Person \sqcap \exists hasName.Name$ whereas the discarded description D is $\exists hasNationality.British$ (Figure 2). The `Name` of $C^1_{P_1}$ can be used to instantiate the `FirstName` and `LastName` requirement (i.e., input parameter) of $C^1_{S_1}$. An instance of J is then required to instantiate (and execute) S_1: $Name$ whereas an instance of $British$ is not.

Semantic Web Service Personalization

We present how we jointly consider standard DL reasoning (Li and Horrocks 2003; Paolucci et al. 2002) together with constructive reasoning abduction (Noia et al. 2003) and join (Definition 2) to adapt services by means of user profiles.

Personalized Adaptation of Services

Adapting services S to be executed by users through their profiles P requires the identification of relevant part not only in P (Definition 2) but also in S. Such part of P could then fit (i.e., be used by) a part of (inputs of) S. Given S, personalization may fail due to under specification of P (description required by S, not provided by P). It is then important to identify this missing description and use it to extend P.

Example 6. ***(Limited Personalization of Services)*** *Even if $C^2_{S_1}$, $C^2_{P_1}$ in Examples 2, 3 are related to* `Account` *in $\mathcal{T}$, $C^2_{P_1}$ cannot be used to personalize S_1 and its conjunct $C^2_{S_1}$ because of missing description* (1) *in its profile P_1 i.e., neither* `OpenID`*, nor* `SkypeAccount`*.*

Given (13), Algorithm 1 achieves this personalization using abduction and join. Potential matching between conjuncts C_S and C_P of S and P are evaluated. First of all, truth values of $\mathcal{T} \models C_P \equiv C_S$ (Exact) and $\mathcal{T} \models C_P \equiv C_S$ (PlugIn) in lines 7 and 10 are checked, emphasizing on description required by S and provided by P. Then $\mathcal{T} \not\models C_P \sqcap C_S \sqsubseteq \bot$ (Intersection) is valued, focusing on a weaker personalization i.e., less part of C_P is relevant to instantiate C_S. Our approach, based on structural algorithms for satisfiability and subsumption (Borgida and Patel-Schneider 1994), successively checks Exact, PlugIn and Intersection because of their logical implication e.g., if $C_P \equiv C_S$ (Exact), then $C_P \sqsubseteq C_S$ (PlugIn) in $\mathcal{T}$. Since it is reasonable to assume that users and service providers do not enter contradicting information, S, P are considered consistent in $\mathcal{T}$.

Algorithm 1: Personalized Adaptation: $adapt(S, P, \mathcal{T})$.

```
1  Input: A user Profile P, a Service S, a Terminology T.
2  Result: m: a set of matching pairs (C_P, C_S) if C_S could be
           adapted by C_P, Incompatibility otherwise.
3  begin
4      m ← ∅;
5      foreach ∃requires.C_S ∈ Input(S) do
6          // Exact Matching between C_P and C_S.
7          if ∃C_P ∈ Info(P) | T ⊨ C_P ≡ C_S then
8              m ← m ∪_set (C_P, C_S);
9          // PlugIn Matching: C_P is more specific than C_S.
10         else if ∃C_P ∈ Info(P) | T ⊨ C_P ⊑ C_S then
11             J ← C_P ⋈_J C_S; // Part J of C_P joined to C_S.
12             m ← m ∪_set (J, C_S));
13         // Intersection Matching: C_P partially covers C_S.
14         else if ∃C_P ∈ Info(P) | T ⊭ C_P ⊓ C_S ⊑ ⊥ then
15             X ← C_S\C_P; // Missing Descriptions in C_P.
16             Y ← C_S\X; // Covered Descriptions in C_S.
17             J ← C_P ⋈_J Y; // Part J of C_P joined to Y.
18             m ← m ∪_set (J, Y);
19         // Incompatibility: C_P and C_S are incompatible.
20         else if ∃C_P ∈ NonInfo(P) | T ⊨ C_P ⊑ C_S then
21             return Incompatibility;
22     return m;
```

The result is a set of matching pairs (C_P, C_S) wherein description C_P in P could be used to adapt service parameter C_S in S. While all description of C_P is required to instantiate C_S in the trivial case (line 7), only a part $C_P \bowtie_J C_S$ of C_P is identified for the PlugIn case (line 10).

Example 7. ***(Personalized Adaptation with PlugIn)*** *Given Example 5, the pair $(C^1_{P_1}, C^1_{S_1})$ is returned. Indeed, it seems possible to personalize S_1 by adapting the* `FirstName` *and* `LastName` *input parameters of S_1 with information of $C^1_{P_1}$ in P_1, and more specially with its* `Name`*.*

In case C_S does not subsume C_P, we need to exhibit description Y (line 16) and J (line 17) from C_S and C_P. According to Definitions 1 and 2, it is straightforward to identify such descriptions and ensure that Y can be adapted by J (in the sense of Algorithm 1 i.e., $\mathcal{T} \models J \sqsubseteq Y$). In such a case we identify relevant parts in C_P which could instantiate (i.e., be subsumed by) part of C_S. In addition, by applying abduction we constructed the description which is required by C_S but not provided by C_P (line 15). The latter description is important to explain the reasons of failure in personalization. Finally if S violates some descriptions in P, S and P are returned as incompatible.

Example 8. ***(Personalized Adaptation with Intersection)*** *Given* (5) *and* (11)*, $\mathcal{T} \not\models C^2_{P_1} \sqcap C^2_{S_1} \sqsubseteq \bot$. Therefore, only some parts J of $C^2_{P_1}$ can be used to adapt some parts Y of $C^2_{S_1}$ (Figure 3). Applying Algorithm 1 (lines from 15 to 17) and using results from Example 1, J and Y are defined as:*

$$X \equiv \exists hasID.OpenID \sqcap \exists hasSkype.SkypeAccount \quad (14)$$

$$Y \equiv Account \sqcap \exists hasSocialNet.SocialNetAccount \quad (15)$$

$$J \equiv Account \sqcap (\exists hasSocialNet.LinkedIn) \quad (16)$$

`LinkedIn` *(J) in $C^2_{P_1}$ can be used to instantiate the* `SocialNetAccount` *(Y) requirement (i.e., input parameter) of $C^2_{S_1}$ while X refers to input description of S_1 which could be instantiated with the current version of P.*

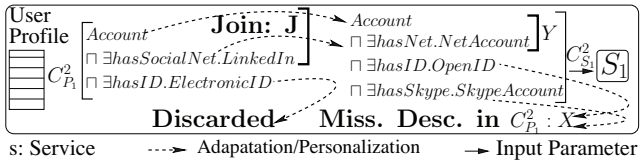

Figure 3: Service Personalization of $C^2_{S_1}$ with $C^2_{P_1}$.

Missing Description in a User Profile

Missing descriptions in P could be computed according to Algorithm 1. Their instances are either discovered through other services or requested to users (Lécué and Delteil 2007) in order to extend and update their profiles. Some of their descriptions may appear comparable under $\sqsubseteq_{\mathcal{T}}$, which do not ensure neither unicity nor the minimality (under $\sqsubseteq_{\mathcal{T}}$) at the end of the personalization process. In this regard, we introduce its minimal solution in Definition 3, which can be used to improve Algorithm 1 (before line 22).

Definition 3. ***(Most Specific Set of Missing Description)*** *The set of missing description $\mathcal{M}$ of a personalization problem $(S, P, \mathcal{T})$ is defined by $\mathcal{M}(S, P, \mathcal{T})$ in* (17) *with C_S and C_P conjuncts of $\exists hasInfo.C_P$ and $\exists requires.C_S$.*

$$\inf_{\sqsubseteq}\{C_S \backslash C_P | \mathcal{T} \not\models C_P \sqcap C_S \sqsubseteq \bot\} \backslash_{set} \{\top\} \quad (17)$$

Since $\mathcal{M}$ gathers the most specific descriptions of $\{C^i_S \backslash C^j_P\}$, a same description (i.e., the most specific) is used to satisfy a finite set of different abduction problems $C^i_S \backslash C^j_P$ and then could be exposed as a description not provided by P but required by some conjuncts (related by subsumption) of S. $\mathcal{M}$ aims to i) explain why services have not been adapted and personalized (regarding a user profile) and ii) suggest a minimal solution to extend personalization.

Property 1. ***(Empty Set of Missing Description)*** *The set of missing description $\mathcal{M}$ of a web service personalization problem $(S, P, \mathcal{T})$ with either (i) $\mathcal{T} \models C_P \equiv C_S$; or (ii) $\mathcal{T} \models C_P \sqsubseteq C_S$ is the empty set.*

Proof. By Definition 1, $C_S \backslash C_P$ is defined by $\mathcal{T} \models C_P \sqcap (C_S \backslash C_P) \sqsubseteq C_S$. Therefore, we obtain in both cases that $C_S \backslash C_P \equiv \top$ is a solution i.e., $\mathcal{M}$ is defined by the empty set according to Definition 3. □

Property 1 justifies our choice of not computing missing descriptions in lines 7 and 10 of Algorithm 1.

Example 9. ***(Set of Missing Description)*** *Since the description in P_1 is not enough to totally adapt and personalize S_1 (Example 6), Algorithm 1 and Definition 3 are required to discover $\mathcal{M}$ in P_1. According to* (17) *$\mathcal{M}$ is constituted by the results of the abduction problems:*

$$C^2_{S_1} \backslash C^2_{P_1} \equiv Account \sqcap \exists hasID.OpenID \sqcap \exists hasSkype.SkypeAccount \quad (18)$$

$$C^4_{S_1} \backslash \top \equiv Location \sqcap \exists hasLat.Latitude \sqcap \exists hasLong.Longitude \quad (19)$$

$$C^3_{S_1} \backslash C^2_{P_1} \equiv MaillingAddress \sqcap \exists hasMail.GMail \quad (20)$$

Since $GMail \sqsubseteq OpenID$ and $hasMail \sqsubseteq hasID$, $\mathcal{M}$ is defined by $\{A, C^4_{S_1}\}$ where A is (21)*.*

$$A \equiv Account \sqcap \exists hasMail.GMail \sqcap \exists hasSkype.SkypeAccount \quad (21)$$

According to Property 1, $C^{i,1\leq i\leq 4}_{S_1}$ and $C^{j,1\leq j\leq 2}_{P_1}$ ensuring $\mathcal{T} \models C^j_{P_1} \sqsubseteq C^i_{S_1}$ are not considered by Algorithm 1.

Experimental Results

From a comparison perspective, it is obvious that explaining how user profile and service could match and why they could not, which is innovative, underperforms existing approaches in term of computation time (mainly because of computation of join, abduction and the number of conjuncts), but reaches a better quality of personalization (in terms of accuracy of predictions). In the following we report a large-scale-based evaluation, focusing on scalability, by evaluating performance of Algorithm 1 after decoupling standard DL reasoning (i.e., satisfiability and subsumption in lines 7, 10, 14, 20), and constructive DL reasoning (i.e., abduction and join in lines 11, 15, 16 and 17). Based on an extension of MAMAS-tng (http://dee227.poliba.it:8080/MAMAS-tng/DIG) reasoner to construct join, we study the impact of the complexity of services and user profile description on the personalization process. The experiments have been conducted on Intel(R) Core (TM)2 CPU, 2.4GHz and 2GB RAM.

• **Context:** Services (2) and user profiles (8) with up to 1000 conjuncts have been respectively extracted from (Oh and Kil 2006) and generated from (Weiß et al. 2008), and then enriched using a commercial $\mathcal{ALC}$ ontology (1100 concepts, 390 properties; 384 concepts subsume the 716 remaining ones with a maximal depth of 8). Their conjuncts are constructed to be pairwise consistent (to ensure relevant instantiation), to be not comparable under $\sqsubseteq_{\mathcal{T}}$ (to discard redundant instantiation) and to ensure a same rate of Exact, Plugin and Intersection tests in Algorithm 1 be true.

• **Results:** Figure 4 illustrates the computation costs of the different DL reasoning used to personalize services with user profile along 10, 100 and 1000 conjuncts (73% of concepts in the ontology). Despite the exponential complexity of DL reasoning in $\mathcal{ALC}$, our approach guarantees to obtain personalization in suitable range of computation time: from 1.8 to 8.1 seconds for services and user profiles with 10 and 100 conjuncts. Abduction computation varies from 32.7%(case 10) to 45.4%(case 1000) due to the increasing number of their computation and its exponential complexity. Lower bounded by the complexity of subsumption (Proposition 1), large scale experiments show that join is the most time consuming reasoning even if pure subsumption is checked twice join. While we considered up to 1000 conjuncts to evaluate scalability, such services are not available yet and would be difficult to maintain and execute.

• **Validation and Open Issues:** Deciding subsumption, computing abduction and join in $\mathcal{ALC}$ are NP-complete. However the personalization process is scalable in our context (Figure 4). The performance of our approach is mainly impacted by the expressivity of the DL used, the number of conjuncts (and their complexity) used to describe services and profiles. The size and the structure of the ontology have a limited impact compared to the former. Scalability of our approach can be improved by considering only subsumption-based comparisons of profiles and services, removing computation of abduction (case in line 14). In such a case automated extension of profiles will not be supported.

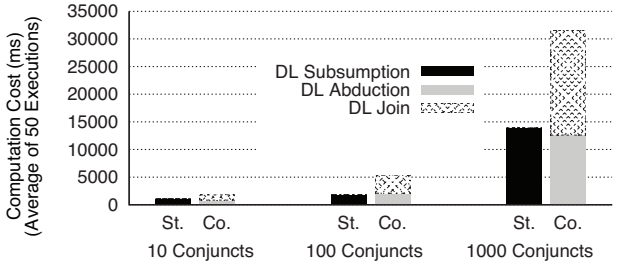

Figure 4: Standard (St.) vs. Constructive (Co.) Reasoning.

Related Work

We emphasize the key deficiencies in the related work i.e., (i) discovery-based personalization so no real adaptation of services with respect to user profile, (ii) no automated instantiation of service with explanation to the user, (iii) no automated extension of user profile.

Web Personalization

Our work could sound similar to the basic "*Auto Fill Form*" feature and HTTP cookies (Kristol 2001) supported by web browsers. Even if the idea of prepopulated web forms or user profile is indeed not new, its prepopulation in the context of web service and DL-based semantic web is. In particular, the proposals of maximizing the impact of personalization by automatically explaining how user profile could be used and extended to instantiate services have been tackled here.

(Baumgartner, Henze, and Herzog 2005) present an approach for RDF-based data extraction, combination and personalization. They generate personalized view of data by applying standard subsumption-based reasoning between data description, user profile and contextual information. Even if their approach is augmented with contextual information, automated extension of user profile is not considered.

Personalization vs. Matching-based Approaches

DL-based semantic matching approaches have also been studied in context of service discovery (Klusch and Kapahnke 2010). Contrary to discovery which aim at finding matching between service requests and advertisements (deciding true, false), service personalization goes further by providing a way to adapt services according to a user profile. In our work, we do not limit to a matching problem but also identify and explain how services can be adapted (wrt join) and express how profiles can be extended, which both are specific to service personalization.

Contrary to services instantiation, (Kleemann and Sinner 2005) perform selection-oriented personalization. Selection is based on the compatibility of users' interests, disinterests and service descriptions through standard reasoning.

DL Reasoning for Service Personalization

While abduction reasoning derives description which is missing in P to be subsumed by S, (Stuckenschmidt 2007) provides the possibility to approximate subsumption with respect to a background terminology. From another perspective concept join constructs more general concepts (under $\sqsubseteq_{\mathcal{T}}$) from P which are subsumed by S. In particular, its maximal form refers to the most general concept. Therefore, abduction and approximate subsumption extend P while join extracts a part of P in order to achieve the same objective i.e., be subsumed by S. Finally, in case P is subsumed by S, abduction and approximate subsumption does not construct any description while concept join does.

(Calì et al. 2004), performing matching on a demand profile P_d and a supply profile P_S for job recruitment or dating system, aim at evaluating their semantic similarity. Abduction is used, but only for weighting, ranking purposes and not for extracting and reusing relevant parts of P_d and P_S.

Conclusion and Future Work

We studied service personalization and addressed automated instantiation of services which is crucial for advanced usability i.e., how to prepare and present services ready to be executed while limiting useless interactions with users? We considered a semantic augmentation of i) services and ii) user profiles to infer and explain (through constructive reasoning join) how the former could be instantiated by the latter. By adapting abduction to personalization, we aimed to extend user profiles by computing and collecting descriptions required by services to be consumed but not provided.

In future work, we will adapt our approach to the Linked Open Data-based semantic web (Bizer, Heath, and Berners-Lee 2009), that will benefit from a scalability point of view. Data privacy is another important direction to consider.

References

Ankolenkar, A.; Paolucci, M.; Srinivasan, N.; and Sycara, K. 2004. The owl-s coalition, owl-s 1.1. Technical report.

Baader, F., and Nutt, W. 2003. In *The Description Logic Handbook: Theory, Implementation, and Applications*.

Baumgartner, R.; Henze, N.; and Herzog, M. 2005. The personal publication reader: Illustrating web data extraction, personalization and reasoning for the semantic web. In *ESWC*, 515–530.

Bizer, C.; Heath, T.; and Berners-Lee, T. 2009. Linked data - the story so far. *Int. J. Semantic Web Inf. Syst.* 5(3):1–22.

Blake, M. B., and Nowlan, M. F. 2007. A web service recommender system using enhanced syntactical matching. In *ICWS*, 575–582.

Borgida, A., and Patel-Schneider, P. F. 1994. A semantics and complete algorithm for subsumption in the classic description logic. *JAIR* 1:277–308.

Calì, A.; Calvanese, D.; Colucci, S.; Noia, T. D.; and Donini, F. M. 2004. A description logic based approach for matching user profiles. In *Description Logics*.

Goldberg, D.; Nichols, D. A.; Oki, B. M.; and Terry, D. B. 1992. Using collaborative filtering to weave an information tapestry. *Commun. ACM* 35(12):61–70.

Kleemann, T., and Sinner, A. 2005. User profiles and matchmaking on mobile phones. In *INAP*, 135–147.

Klusch, M., and Kapahnke, P. 2010. isem: Approximated reasoning for adaptive hybrid selection of semantic services. In *ESWC*, 30–44.

Kopecký, J.; Vitvar, T.; Bournez, C.; and Farrell, J. 2007. Sawsdl: Semantic annotations for wsdl and xml schema. *IEEE Internet Computing* 11(6):60–67.

Kristol, D. M. 2001. Http cookies: Standards, privacy, and politics. *ACM Trans. Internet Techn.* 1(2):151–198.

Lécué, F., and Delteil, A. 2007. Making the difference in semantic web service composition. In *AAAI*, 1383–1388.

Li, L., and Horrocks, I. 2003. A software framework for matchmaking based on semantic web technology. In *WWW*, 331–339.

Lieberman, H. 1995. Letizia: An agent that assists web browsing. In *IJCAI (1)*, 924–929.

McIlraith, S. A., and Son, T. C. 2002. Adapting golog for composition of semantic web services. In *KR*, 482–496.

Mobasher, B.; Cooley, R.; and Srivastava, J. 2000. Automatic personalization based on web usage mining. *Commun. ACM* 43(8):142–151.

Noia, T. D.; Sciascio, E. D.; Donini, F. M.; and Mongiello, M. 2003. Abductive matchmaking using DLs. In *IJCAI*, 337–342.

Oh, S.-C., and Kil, H. 2006. Wsben: A web services discovery and composition benchmark. In *ICWS*, 239–248.

Paolucci, M.; Kawamura, T.; Payne, T.; and Sycara, K. 2002. Semantic matching of web services capabilities. In *ISWC*, 333–347.

Pistore, M.; Marconi, A.; Bertoli, P.; and Traverso, P. 2005. Automated composition of web services by planning at the knowledge level. In *IJCAI*, 1252–1259.

Smith, M. K.; Welty, C.; and McGuinness, D. L. 2004. Owl web ontology language guide. W3c recommendation.

Stuckenschmidt, H. 2007. Partial matchmaking using approximate subsumption. In *AAAI*, 1459–1464.

Weiß, D.; Scheuerer, J.; Wenleder, M.; Erk, A.; Gülbahar, M.; and Linnhoff-Popien, C. 2008. A user profile-based personalization system. In *DIMEA*, 281–288.