An Abductive CQA Based Matchmaking System for Finding Renting Houses

Jianfeng Du[1,2], Shuai Wang[1], Guilin Qi[3], Jeff Z. Pan[4], and Che Qiu[1]

[1] Guangdong University of Foreign Studies, Guangzhou 510006, China
jfdu@mail.gdufs.edu.cn
[2] State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China
[3] Southeast University, Nanjing 211189, China
[4] The University of Aberdeen, Aberdeen AB243UE, UK

Abstract. A matchmaking system for finding renting houses is required as the housing problem becomes serious in China and many people resort to rent a house. A semantic approach based on abductive conjunctive query answering (CQA) in Description Logic ontologies is exploited to provide more matches for a request about renting houses. Moreover, a matchmaking system based on this approach is developed. This demo will guide users to find suitable renting houses using this matchmaking system and show the advantages of the system.

1 Motivation

The housing problem is an important social problem in China due to the large scale of population. This problem becomes more serious recently. According to some survey, the housing price-to-income ratio in China is very high and such a high ratio implies that about 85 percent of the families cannot afford a house in cities [7]. Under this situation, many people resort to rent a house and use matchmaking systems to find suitable renting houses.

Keyword search based matchmaking systems for finding renting houses are backing up the current rental search engines which are available at the World Wide Web. Although these matchmaking systems are rather efficient, they are hard to provide sound and complete matches. On the one hand, keyword search based matchmaking makes little use of the background knowledge and will easily miss real matches. For example, HEMC (Higher Education Mega Center) locates in Panyu (a district in Guangzhou). When a request for renting houses in Panyu is posed, a matchmaking system based on keyword search will not output any offer which describes that the renting house is in HEMC, because it does not involve reasoning by using the relationship between HEMC and Panyu. On the other hand, keyword search based matchmaking makes little use of the semantics of offers or requests and will easily output wrong matches. For example, a matchmaking system based on keyword search does not manage the meaning of the word "top floor". When a request for renting houses at the top floor is posed,

the system will output offers describing that the renting houses are at any floor, because the word "floor" appears in the offer descriptions.

To make the matchmaking results more sound and complete, ontology-based matchmaking systems have been proposed. There are two approaches to ontology-based matchmaking. One approach exploits a Description Logic (DL) [1] ontology to compute semantic distances between offers and requests, where offers and requests are expressed as DL concept descriptions. This approach is followed by many methods such as those proposed in [2, 8, 16]. They mainly focus on defining a reasonable distance function between two DL concept descriptions. The other approach exploits DL inference methods to compute different kinds of matches. Concept subsumption checking is the most popular one among such methods [12, 15]. Other known DL inference methods include concept abduction and concept contraction [14]. They are used to compute possible matches in a negotiation framework. That is, an offer is regarded as a possible match for a request if it can get subsumed by the request after *abduction*, i.e. adding some information to itself, and *contraction*, i.e. removing some information from the request.

However, the two main ontology-based matchmaking approaches are not easy to scale to real-life applications that involve a large number of offers and requests, because composing the DL concept descriptions for offers and requests is time consuming and laborious. To alleviate human efforts to formalize offers or requests, our initial solution is to treat the matchmaking problem as the conjunctive query answering (CQA) problem. In this solution, offer information is expressed as individual assertions in the back-end ontology, request information is expressed as conjunctive queries, and a match for a request is defined as an answer to a conjunctive query that expresses the request. This solution allows offer information to be automatically extracted from text sources, such as Web pages, using off-the-shelf ontology population techniques [3], thus significantly reducing human efforts. However, this solution is still hard to provide complete matches due to incompleteness of information extracted from the World Wide Web. When some information about an offer is missing in the back-end ontology, the offer will not be an answer to a conjunctive query that expresses the given request, but it will turn to be after missing information is added, so this offer can also be considered as a match for the request.

To realize the above idea, in [6] we introduced the *abductive CQA* problem which computes all *abductive answers* to a conjunctive query in a consistent ontology. An abductive answer is an answer to the given query in some consistent ontology enlarged from the given one by adding a bounded number of individual assertions, where the individual assertions that can be added are confined by user-specified concept or role names. We developed an abductive CQA based matchmaking system for finding renting houses, where a match for a request is defined as an abductive answer to a conjunctive query that expresses the request.

In the next two sections, we highlight the novelty of our matchmaking system and give preliminaries about this demo, respectively. Then, before the last section in which we explain what will be shown in this demo, we describe the architecture of the matchmaking system.

2 Novelty

The matchmaking system provides a DL based framework for integrating information from different Web sources and handling user requests for renting houses. It has some advantages. One is that the system can accept a complex request considering nearby traffic lines and public facilities. These requests for renting houses cannot be handled by current rental search engines. Another advantage is that all output information for responding to a request is interpretable by existing DL reasoning facilities.

3 Preliminaries

3.1 OWL 2 and Conjunctive Query Answering

The World Wide Web Consortium (W3C) has proposed the Web Ontology Language (OWL), for which the newest version is OWL 2 [9], to model ontologies. OWL is based on DLs [1]. In particular, the most expressive and decidable species of OWL 2, OWL 2 DL, corresponds to the DL $\mathcal{SROIQ}$ [11]. An OWL 2 DL ontology consists of an RBox, a TBox and an ABox. The RBox consists of a finite set of complex role inclusion axioms and role assertions declaring that a role is symmetric, transitive, reflexive, irreflexive, or disjoint with another role. The TBox consists of a finite set of concept inclusion axioms. The ABox consists of a finite set of individual assertions that declare memberships of concepts or roles, or equivalence relations between individuals. Since $\mathcal{SROIQ}$ is a fragment of First-order Logic, its semantics can be defined by translating to First-order Logic. For example, the following two axioms expressed in $\mathcal{SROIQ}$, namely $\mathsf{hasFacility} \circ \mathsf{isA} \sqsubseteq \mathsf{hasFacility}$ and $\mathsf{House} \sqsubseteq \mathsf{Building}$, can be translated to two First-order rules given below and inherit the standard First-order semantics, where the former rule tells that if x has a facility y and y is more specific than z, then x also has a facility z, while the latter one tells that if x is a renting house, then it is also a building.

$\forall x, y, z : \mathsf{hasFacility}(x, y) \wedge \mathsf{isA}(y, z) \rightarrow \mathsf{hasFacility}(x, z)$
$\forall x : \mathsf{House}(x) \rightarrow \mathsf{Building}(x)$

A *model* of an OWL 2 DL ontology is an interpretation on all entities in the ontology that satisfies all First-order rules translated from the ontology under the standard First-order semantics. An OWL 2 DL ontology is said to be *consistent* if it admits at least one model.

A *conjunctive query* is an expression of the form $\exists \overrightarrow{y} : \mathsf{conj}(\overrightarrow{x}, \overrightarrow{y}, \overrightarrow{c})$, where $\overrightarrow{x}$ and $\overrightarrow{y}$ are both vectors of variables, and $\overrightarrow{c}$ is a vector of individuals or constants. $\mathsf{conj}(\overrightarrow{x}, \overrightarrow{y}, \overrightarrow{c})$ denotes a conjunction of *atoms* of the form $A(v)$ or $r(v_1, v_2)$, where A is an *atomic concept* (i.e. a concept name), r is an *atomic role* (i.e. a role name) or a built-in predicate, and v, v_1 and v_2 are variables in $\overrightarrow{x}$ and $\overrightarrow{y}$, or individuals or constants in $\overrightarrow{c}$. A *Boolean conjunctive query* is a conjunctive query without distinguished variables.

Given an OWL 2 DL ontology $\mathcal{O}$ and a Boolean conjunctive query $Q = \exists \overrightarrow{y} : \mathsf{conj}(\overrightarrow{y}, \overrightarrow{c})$, a model $\mathcal{I}$ of $\mathcal{O}$ is said to *satisfy* Q if there exists a tuple of (possibly

anonymous) individuals or constants whose substitution for the variables in $\overrightarrow{y}$ makes every atom in $\mathsf{conj}(\overrightarrow{y}, \overrightarrow{c})$ satisfied by $\mathcal{I}$. Q is said to be *entailed* by $\mathcal{O}$, denoted by $\mathcal{O} \models Q$, if every model of $\mathcal{O}$ satisfies Q. A tuple $\overrightarrow{t}$ of individuals is called an *answer* to a conjunctive query $Q(\overrightarrow{x}) = \exists \overrightarrow{y} : \mathsf{conj}(\overrightarrow{x}, \overrightarrow{y}, \overrightarrow{c})$ in $\mathcal{O}$ if $\mathcal{O} \models Q(\overrightarrow{x})[\overrightarrow{x} \mapsto \overrightarrow{t}]$, where $Q(\overrightarrow{x})[\overrightarrow{x} \mapsto \overrightarrow{t}]$ denotes a Boolean conjunctive query obtained from $Q(\overrightarrow{x})$ by replacing every variable in $\overrightarrow{x}$ with its corresponding individual in $\overrightarrow{t}$. The *conjunctive query answering* (*CQA*) problem is to compute all answers to a conjunctive query in an ontology.

3.2 Abductive Conjunctive Query Answering

As mentioned before, the matchmaking problem can be treated as the CQA problem, where offer information and background knowledge are stored in a back-end ontology. For example, when we want to find all southward renting houses in Guangzhou, we can pose the following conjunctive query upon the back-end ontology:

$$\mathsf{House}(x) \wedge \mathsf{locatesIn}(x, \mathsf{Guangzhou}) \wedge \mathsf{towards}(x, \mathsf{South}).$$

Then the answers to this query, namely the individuals substituting for the variable x, are renting houses to be found. However, these answers may not provide all choices to a requester. For example, when the orientation of a renting house is missing, possibly due to incomplete extraction from Web pages, this renting house will not be an answer to the aforementioned query, although its orientation is south in reality. To compensate these answers, a certain enlarged ontology should be considered. This ontology can be seen as the result of adding missing information about offers to the back-end ontology.

Hence, in [6] we introduced a new kind of answers, called *abductive answers*, to a conjunctive query. Abductive answers are formally defined as follows. Given a consistent ontology $\mathcal{O}$, a conjunctive query Q, a non-negative integer k, two disjoint sets of concept or role names S_A and S_C, an *abductive answer* $\overrightarrow{t}$ to Q in $\mathcal{O}$ w.r.t. k, S_A and S_C is an answer to Q in $\mathcal{O} \cup \mathcal{A}$ for some set $\mathcal{A}$ of individual assertions such that the cardinality of $\mathcal{A}$ is not greater than k, all individual assertions in $\mathcal{A}$ are on abducible predicates, and any individual assertion on closed predicates that is entailed by $\mathcal{O} \cup \mathcal{A}$ is also entailed by $\mathcal{O}$, where $\mathcal{A}$ is said to be *attached with* $\overrightarrow{t}$ and the concept or role names in S_A (resp. S_C) are called *abducible predicates* (resp. *closed predicates*). In this definition, $\mathcal{A}$ can be seen as the missing information about a certain offer. The definition says that $\mathcal{A}$ should consist of at most k individual assertions, where k is a user-specified parameter which reflects the incompleteness of the given ontology $\mathcal{O}$. It also says that all concepts/roles appearing in $\mathcal{A}$ should be abducible predicates, and appending $\mathcal{A}$ to $\mathcal{O}$ should not make $\mathcal{O}$ entail any individual assertion α on closed predicates unless α is already entailed by the original $\mathcal{O}$. We call the problem of computing all abductive answers to a conjunctive query in a consistent ontology the *abductive CQA* problem.

The abductive CQA problem is similar to the ABox abduction problem proposed in [4] which, for a consistent ontology $\mathcal{O}$ and a set G of individual asser-

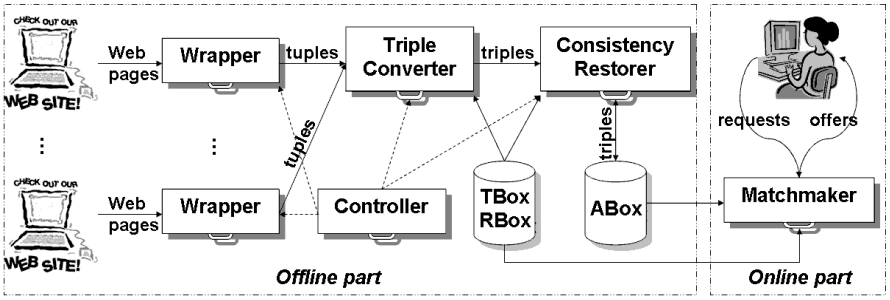

Fig. 1. The interactions among components of the matchmaking system

tions, computes all minimal sets $\mathcal{A}$ of individual assertions on a set S of predicates such that $\mathcal{O} \cup \mathcal{A}$ is consistent, and $\mathcal{O} \cup \mathcal{A}$ entails all individual assertions in G but $\mathcal{A}$ does not. Compared to the ABox abduction problem, the abductive CQA problem also restricts $\mathcal{A}$ to a set of individual assertions on abducible predicates such that appending it to $\mathcal{O}$ does not introduce inconsistency, where abducible predicates are used to define which kind of information is incomplete. However, instead of computing certain $\mathcal{A}$, the abductive CQA problem computes abductive answers to a conjunctive query by considering all possible $\mathcal{A}$. Moreover, it introduces the use of the parameter k and a set of closed predicates. The parameter k is used to control the extensiveness of abducible answers, while closed predicates are used to simulate disjoint concept or role axioms (see [6] for the explanations on this usage) and enable some optimizations in computing abductive answers. These optimizations are exploited in a method for computing abductive answers [6]. Basically, the method encodes the abductive CQA problem into a Prolog program and solves it with Prolog engines. It computes exactly all abductive answers in a consistent *DLP* ontology, which is an ontology expressed in the DLP fragment [10] of OWL 2 DL and adopts the Unique Name Assumption [1] that any two different individual names must correspond to different elements in the interpretation domain.

4 The Architecture of the Matchmaking System

We developed an abductive CQA based matchmaking system for finding renting houses, where a match for a request for renting houses is defined as an abductive answer to a conjunctive query that expresses the request. The system consists of two parts, the offline part and the online part, as shown in Fig. 1. The offline part targets integrating rental related information from different Websites into a consistent DLP ontology, and consists of several components, including a controller, a triple converter, a consistency restorer and multiple wrappers. The online part targets responding to user requests about renting houses, and is built on a matchmaker which implements the method for abductive CQA [6]. More details for all these components are given below.

4.1 Wrapper

A wrapper is a component for extracting tuples from a Website. It first downloads Web pages from a Website, then extracts tuples from these Web pages. We developed different wrappers for different Websites, including those that provide rental search services and those publishing information on traffic lines and administrative regions. The extraction process relies on a set of extraction rules, some of which were manually written and some were automatically learned by machine learning methods [13].

4.2 Triple Converter

The triple converter is a component for converting the extracted tuples to individual assertions, which are stored as RDF triples in the ABox of the back-end ontology. In this converter, the TBox and the RBox of the back-end ontology, which were manually constructed using ontology editors, confine the concept or role names that can appear in a generated individual assertion.

Moreover, some methods for relation extraction were developed to add links between two individuals from different Websites. For example, the tuples extracted from a rental Website have a free-text field mentioning traffic lines and stops that a renting house locates near. We developed a pattern based method for extracting role assertions of the form $\mathsf{locatesNear}(a, b)$ from this free-text field, where a is a renting house and b is either a traffic line or a stop. The method first generates a pattern set based on the words about traffic lines or stops that are extracted from a traffic Website, then uses this pattern set to identify entities about traffic lines or stops in the free-text field and compose role assertions of the form $\mathsf{locatesNear}(a, b)$. The individual b in a generated role assertion $\mathsf{locatesNear}(a, b)$ may not have the same name as the truly same individual extracted from other Websites. We used the same pattern set to determine whether two individuals with different names are the same in reality. That is, we defined that two individuals are the truly same if the core parts of their names are the same, where the core part of a name is computed by some manually written rules based on matched patterns of the name. On the other hand, two individuals with the same name but extracted from different Websites may not be the same in reality. We defined that such two individuals are the truly same if the concepts they belong to are not disjoint in the back-end ontology. To avoid introducing equality assertions, the triple converter only mentions a representative for all truly same individuals in the back-end ontology.

4.3 Consistency Restorer

The consistency restorer is a component for rendering the back-end ontology consistent when it is inconsistent. The method proposed in [5] is applied to compute a cost-minimal set of individual assertions that should be removed to restore consistency, where the cost of an individual assertion is determined by the importance of the concept or role name that the assertion is on. This kind of importance information is set by the administrator of the system.

4.4 Controller

The controller is a component for scheduling the execution of every wrapper, the triple converter and the consistency restorer. It periodically invokes all wrappers to extract tuples. After the tuple extraction process finishes, it invokes the triple converter to generate individual assertions and add them to the ABox. After the triple generation process finishes, it invokes the consistency restorer to render the evolved back-end ontology consistent.

4.5 Matchmaker

The matchmaker is the kernel component in the online part. To start this component, the administrator of the system should set the parameters k, S_A and S_C for computing abductive answers, and then trigger the component to perform preprocessing, i.e. to encode the back-end ontology and the parameters k, S_A and S_C to a Prolog program and load it into a Prolog engine. Afterwards, this component waits for user requests for renting houses. Once it receives a user request, it first translates it to a conjunctive query, then combines the conjunctive query with the loaded Prolog program and computes all abductive answers from the combination. These abductive answers correspond to renting houses that are offered. The abductive answers that can attach with (cardinality) smaller sets of individual assertions are output earlier. To show why a renting house is an abductive answer, a cardinality-minimal set of individual assertions attached with it is also displayed.

5 What Will be Demonstrated?

This demo will guide users to find suitable renting houses in China using the proposed matchmaking system. In more details, we will show how to compose a request for renting houses in this system and how to analyze the output information to pick up a suitable renting house. The demo will also show some advantages of the system, including that it can handle complex requests considering nearby traffic lines and public facilities and that all output information is interpretable by existing DL reasoning facilities.

Acknowledgement

Jianfeng Du and Shuai Wang are partly supported by the NSFC under grant 61005043 and the Undergraduate Innovative Experiment Project in Guangdong University of Foreign Studies. Guilin Qi is partly supported by Excellent Youth Scholars Program of Southeast University under grant 4009001011, the NSFC under grant 61003157, Jiangsu Science Foundation under grant BK2010412, and the Key Laboratory of Computer Network and Information Integration (Southeast University). Jeff Z. Pan is partly supported by the EU K-Drive project and the RCUK dot.rural project.

References

1. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press (2003)
2. Bianchini, D., Antonellis, V.D., Melchiori, M.: Flexible semantic-based service matchmaking and discovery. World Wide Web 11(2), 227–251 (2008)
3. Cimiano, P., Völker, J.: Text2onto - a framework for ontology learning and data-driven change discovery. In: Proc. of the 10th International Conference on Applications of Natural Language to Information Systems (NLDB). pp. 227–238 (2005)
4. Du, J., Qi, G., Shen, Y., Pan, J.Z.: Towards practical abox abduction in large OWL DL ontologies. In: Proc. of the 25th AAAI Conference on Artificial Intelligence (AAAI). pp. 1160–1165 (2011)
5. Du, J., Shen, Y.: Computing minimum cost diagnoses to repair populated DL-based ontologies. In: Proc. of the 17th International World Wide Web Conference (WWW). pp. 265–274 (2008)
6. Du, J., Wang, S., Qi, G., Pan, J.Z., Hu, Y.: A new matchmaking approach based on abductive conjunctive query answering. In: Proc. of the 2011 Joint International Semantic Technology Conference (JIST) (2011)
7. Feng, W., Wu, N.: On the capital production of space and china's housing problem. Journal of Wu ling 35(6), 55–59 (2010)
8. Fenza, G., Loia, V., Senatore, S.: A hybrid approach to semantic web services matchmaking. International Journal of Approximate Reasoning 48(3), 808–828 (2008)
9. Grau, B.C., Horrocks, I., Motik, B., Parsia, B., Patel-Schneider, P.F., Sattler, U.: OWL 2: The next step for OWL. Journal of Web Semantics 6(4), 309–322 (2008)
10. Grosof, B.N., Horrocks, I., Volz, R., Decker, S.: Description logic programs: combining logic programs with description logic. In: Proc. of the 12th International World Wide Web Conference (WWW). pp. 48–57 (2003)
11. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible $\mathcal{SROIQ}$. In: Proc. of the 10th International Conference on Principles of Knowledge Representation and Reasoning (KR). pp. 57–67 (2006)
12. Li, L., Horrocks, I.: A software framework for matchmaking based on semantic web technology. In: Proc. of the 12th International World Wide Web Conference (WWW). pp. 331–339 (2003)
13. Liu, B.: Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data. Data-Centric Systems and Applications, Springer (2007)
14. Noia, T.D., Sciascio, E.D., Donini, F.M.: Semantic matchmaking as non-monotonic reasoning: A description logic approach. Journal of Artificial Intelligence Research 29, 269–307 (2007)
15. Noia, T.D., Sciascio, E.D., Donini, F.M., Mongiello, M.: A system for principled matchmaking in an electronic marketplace. In: Proc. of the 12th International World Wide Web Conference (WWW). pp. 321–330 (2003)
16. Shu, G., Rana, O.F., Avis, N.J., Chen, D.: Ontology-based semantic matchmaking approach. Advances in Engineering Software 38(1), 59–67 (2007)