Learning and Reasoning about Uncertainty in the Semantic Web

Pedro Oliveira and Paulo Gomes

CISUC-DEI-FCTUC
University of Coimbra
Pólo II, 3030 Coimbra, Portugal
pcoliv@student.dei.uc.pt, pgomes@dei.uc.pt

Abstract. The main idea behind the Semantic Web is the representation of knowledge in an explicit and formal way. This is done using ontology representation languages as OWL, which is based on Description Logics and other logic formalisms. One of the main objectives with this kind of knowledge representation is that it can then be used for reasoning. But the way reasoning is done in the Semantic Web technology is very strict, defining only a right and wrong view of the world. The real world is uncertain and humans have learned how to deal with this crucial aspect. In this paper, we present an approach to reasoning with uncertainty information in the Semantic Web. We have applied Markov Logic, which is able to reason with uncertainty information, to several Semantic Web ontologies, showing that it can be used in several applications. We also describe the main challenges for reasoning with uncertainty in the Semantic Web.

Keywords: Semantic Web, Probabilistic Reasoning, Markov Logic.

1 Introduction

The idea of the Semantic Web [1] envisions a world where agents share and transfer structured knowledge in an open and semi-automatic way. In most of the cases, this knowledge is characterized by uncertainty. However, Semantic Web languages like OWL[1] do not provide any means of dealing with this uncertainty. They are mainly based on crisp logic, unable of dealing with partial and incomplete knowledge. Reasoning in the Semantic Web resigns to a deterministic process of verifying if statements are true or false.

In the last years, some efforts have been made in representing and reasoning with uncertainty in the Semantic Web (see [2] for a complete overview about the subject). These works are mainly focused on how to extend the logics behind Semantic Web languages to the probabilistic/possibilistic/fuzzy logics, or on how to combine these languages with probabilistic formalisms like Bayesian Networks. In all of these

[1] http://www.w3.org/TR/owl-features/

approaches, this is achieved by annotating the ontologies with some kind of uncertainty information about its axioms, using this information to perform uncertainty reasoning. Nevertheless, several questions arise: how are these uncertainties asserted? How can reasoning be done with this uncertainty information?

One promising approach to reasoning with uncertainty is Markov Logic [3]. In this type of logic there is no right and wrong world, there are multiple worlds with different degrees of probability. Markov Logic is based in first-order logic and probabilistic graphical models to deliver the probability of a given logic formula. This type of logic has been applied to several application domains [3] and has show to be robust and able to deal with uncertain knowledge.

In our work, we are studying how we can reason about uncertainty in OWL ontologies without any kind of uncertainty associated. In this paper, we describe an approach that uses Markov Logic to accomplish this task. First, the ontology is interpreted as first-order logic, and ontology individuals are used to learn the uncertainty of the resulting formulas. Next, we use Markov Logic inference capabilities to perform approximate probabilistic reasoning in the resulting model. We present several experiments of this approach with different OWL ontologies.

All the capabilities described in this paper are implemented in *Incerto*[2], an open source probabilistic reasoner for the Semantic Web.

The next sections introduce the concepts of Semantic Web and Markov Logic. Section 4 describes our approach to the transformation of OWL into Markov Logic. Section 5 presents the experimental work done and its main results. We finalize this paper by describing future work and conclusions of our work.

2 Semantic Web

In the current web, while it is easy to a human infer the meaning of objects in a web page, to a machine this task is not so easy, being only possible to interpret the keywords and links of those objects. The Semantic Web [1] tries to fill this knowledge gap between human and machines by adding background knowledge to the web, allowing machines to infer the real meaning of objects. This background knowledge is usually expressed by ontologies [4], i.e., sets of knowledge terms for some particular topic, including the vocabulary, semantic interconnections, and rules of logic/inference of those terms.

The most prominent markup language proposed by the *W3C* to model ontologies in the Semantic Web is the *Web Ontology Language*[3] (OWL). OWL provides an expressive shared vocabulary to represent knowledge in the Semantic Web. This vocabulary allows expressing axioms about classes, properties, and individuals of the domain. In this paper, we will focus on OWL2[4] [5], the new version of OWL proposed by the *W3C*, which subsumes the decidable subsets of the original OWL

[2] http://code.google.com/p/incerto/
[3] http://www.w3.org/2004/OWL/
[4] http://www.w3.org/TR/owl2-quick-reference/

(OWL-DL and OWL-Lite).

OWL2 is based on the Description logic $\mathcal{SROIQ(D)}$ [5]. Description logics [6] are a family of logical languages specially designed to model terminological domains. Formulas in Description Logics are composed by two symbols: *concepts* (i.e., sets of individuals) and *roles* (i.e., relationships between individuals). A relevant feature of Description Logics is their separation of knowledge bases in two distinct parts: the intensional knowledge in the form of a terminology, called *Terminological Box* (TBox), and the extensional knowledge, called *Assertional Box* (ABox). The TBox provides the vocabulary, in terms of concepts and rules, of the knowledge base. This is usually done by defining concepts using the logical equivalence constructor (e.g., $Woman \equiv Person \sqcap Female$). The Abox uses the TBox vocabulary to make assertions about individuals (e.g. $Woman(ANNA)$).

3 Markov Logic

Markov Logic [3] combines first-order logic and probabilistic graphical models (Markov networks [7]) in the same representation. The main idea behind Markov Logic is that, unlike first-order logic, a world that violates a formula is not invalid, but only less probable. This is done by attaching weights to first-order logic formulas: the higher the weight, the bigger is the difference between a world that satisfies the formula and one that does not, other things been equal. These sets of weighted formulas are called Markov Logic networks (MLNs). Given a set of constants (i.e., individuals) of the domain and an interpretation, the groundings of the formulas in an MLN can generate a Markov network by adding a variable for each ground atom, an edge if two ground atoms appear in the same formula, and a feature for each grounded formula. The probability distribution of the network is defined as

$$\mathrm{P}(X = x) = \frac{1}{Z} exp\left(\sum_{i=1}^{F} w_i n_i(x)\right), \quad (\mathbf{1})$$

where F is the number of formulas in the MLN, $n_i(x)$ is the (binary) number of true groundings of F_i in the world x, w_i is the weight of F_i, and Z is a normalizing constant.

There are two relevant tasks of Markov Logic for this work: weight learning and inference.

3.1 Weight Learning

Given an MLN without weights and a set of example data composed by individuals of the domain, weights can be learned generatively by maximizing the *pseudo-log-likelihood* [8] of that data. Basically, it is an iterative process where if the model predicts that a formula is true less often than it really is in the data, the weight is increased; otherwise, it is decreased. The pseudo-log-likelihood of world x given weight w is defined as

$$\log P_w^*(X = x) = \sum_i \log P_w(X_i = xi \mid N_x(X_i)), \tag{2}$$

where xi is the truth value of variable i, and $N_x(X_i)$ is the truth values of the neighbors of i.

3.2 Inference

The most interesting inference task in Markov Logic is to find the marginal and conditional probabilities of a formula given an MLN and possibly other formulas as evidence. Since exact inference can be too difficult in large domains, approximate inference algorithms, like those based on randomized sampling (e.g., *Markov Chain Monte Carlo* [7] (MCMC)), are usually used. However, MCMC is not efficient in domains where formulas with deterministic or near-deterministic dependencies exist (e.g., formulas with infinite weight) because these areas of the search space can be very difficult to traverse by simple flipping the value of the non-evidence variables. To solve this problem, we can use *MC-SAT* [9], a combination of MCMC and the *SampleSAT* satisfiability solver [10]. MC-SAT uses *slice sampling* to help capturing the dependencies between variables, allowing jumping from these difficult areas.

4 Markov Logic for the Semantic Web

As we previously seen, MLNs are formed by a set of weighted first-order logic formulas. If we want to use Markov Logic in the Semantic Web, we have to determine where these formulas and weights come from.

4.1 Formulas

The Semantic Web language used in this work (OWL2) is based on the Description Logic $\mathcal{SROIQ}(\mathcal{D})$. One characteristic of Description Logic languages is that they follow a *model-theoretic* semantics [6], and therefore can (in most of the cases) be interpreted as formulas in first-order logic. The main idea behind this interpretation is that concepts correspond to unary predicates, roles to binary predicates, and individuals correspond to constants. In our case, $\mathcal{SROIQ}(\mathcal{D})$ can be easily interpreted as first order formulas. Some examples of these translations are provided (Table 1).

Table 1. OWL2 examples of interpretation as first-order logic formulas. A complete description of the interpretation can be found on the *Incerto* website[5].

OWL2 Expression	First-order logic formula
$SubClassOf(CE_1, CE_2)$	$\forall x : CE_1(x) \Rightarrow CE_2(x)$
$TransitiveProperty(OPE)$	$\forall x, y, z : OPE(x, y) \wedge OPE(y, z) \Rightarrow OPE(x, z)$
$ClassAssertion(CE, a)$	$CE(a)$

4.2 Weights

The most obvious way to acquire the uncertainty from an ontology is to delegate this task to the ontology creators. This is the approach used by other works [2]. However, creating and maintaining large uncertainty-annotated ontologies can be a cumbersome and difficult task, invalidating all the gains that could arise from the annotation. This fact raises the need for developing mechanisms to learn this uncertainty automatically. This can be useful not only to help users when creating uncertain ontologies, but also to gain access to the vast number of non-annotated ontologies already available.

In this paper, we explore the use of the weight learning capabilities of Markov Logic to learn uncertainty information. As previously seen, in Markov Logic, formulas' weights can be learned generatively through example data. This example data comprises individuals of the domain and their relations. In the case of OWL2, this corresponds to the ABox of the ontology. Therefore, the ABox can be interpreted as ground atoms, and weights can be learned with that information.

5 Experimental Analysis

In this section, we present our experiences on using Markov Logic to learn and reason about uncertainty in OWL2 ontologies. The main objective of these experiments is to show the feasibility of our approach in real-world domains. All the experiences were made with *Incerto*, using Alchemy[6] [11] as the Markov Logic engine. All the ontologies and results of the experiences can be also found on the *Incerto* website[7].

5.1 The Financial Experiment

Evaluation Procedure and Data Set. Uncertainty reasoning is very important in discovering hidden knowledge in risk assessment domains. In this experiment, we will use a financial ontology, GoldDLP[8], to assess the risk of certain financial operations. In this ontology, there is information about a bank that offers services like

[5] http://code.google.com/p/incerto/wiki/OWL2FOL
[6] http://alchemy.cs.washington.edu/
[7] http://code.google.com/p/incerto/wiki/EPIA2009Experimentation
[8] http://www.cs.put.poznan.pl/alawrynowicz/semintec.htm

loans and credit cards to private persons. The ontology contains 116 class/property axioms and 297 individuals, mainly distributed between accounts, clients, credit cards, and loans. One of the most interesting tasks in this domain is to determine if a given loan is a problematic loan. There is an OWL class responsible for that information, named *ProblemLoan*, and some axioms about that class (e.g., *ProblemLoan* is the complement of *OkLoan*). The main task in this experiment is to determine each loan's probability of being a *ProblemLoan*.

Experimental Results. Using generative learning and MC-SAT, we found that nine loans have a probability >90% of satisfying the conditions necessary for being a *ProblemLoan*. If we compare the results with a non-probabilistic reasoner, like Pellet[9] [12], these are the same nine individuals identified deterministically by it. However, our approach returns some more interesting results that were not identified by Pellet. All the other loans have a probability between 35-39% of satisfying the conditions of *ProblemLoan*. This information is valuable because, roughly speaking, it demonstrates that any loan has an associated probability of being a problematic loan. This kind of results cannot be achieved using non-probabilistic reasoning, and therefore demonstrates the necessity of probabilistic reasoning to have a more profound understanding about the domain. However, if we use an existent Semantic Web probabilistic reasoner (e.g., Pronto[10] [13]), its results are the same of a non-probabilistic one, since the ontology does not contain any information about the uncertainty of its axioms.

5.2 The Social Network Experiment

One of the most used Semantic Web vocabularies is the *Friend of a Friend*[11] (FOAF) vocabulary. This vocabulary allows describing social network data (i.e., persons and their relations) in OWL, with special incentive in linking users from different social networks. There are several web-based social networks that provide information about their users in FOAF (see Mindswap[12] for a comprehensive list), and some projects are already exploiting that information (e.g., Google Social Graph API[13]).

The objective of this experiment is to use Markov Logic to explore the relational structure of FOAF networks. As data set, we choose Advogato[14], a social network of free software developers. Advogato provides three interesting FOAF properties to our analysis: *foaf:knows(x,y)*, meaning that user *x* knows user *y*; *foaf:currentProject(x,y)*, meaning that user *x* is currently working in project *y*; and *foaf:member(x,y)*, meaning that user *x* is member of the group *y*. After gathering and processing all the available FOAF profiles, we had a total of 6688 individuals, representing 4198 users, 2487 projects, and 3 groups. Based on the Link Mining literature [14][15], we identified

---

[9] http://pellet.owldl.com/
[10] http://pellet.owldl.com/pronto
[11] http://www.foaf-project.org/
[12] http://trust.mindswap.org
[13] http://code.google.com/apis/socialgraph/
[14] http://advogato.org/

three interesting tasks to our experiment: link prediction, link-based classification, and link-based cluster analysis.

5.2.1 Link Prediction

Link prediction [14] is the problem of predicting the existence of a link between two objects based on the relations of the object with other objects. In our domain, we are particularly interested in predicting the acquaintance between users, i.e., the *foaf:knows* property. For this purpose, based on our common sense about the domain, we defined three simple first-order logic rules to perform this task:

Table 2. Link prediction rules.

Weight	Formula
0.09	$knows(x,y) \wedge knows(y,z) \Rightarrow knows(x,z)$
2.70	$knows(x,y) \Leftrightarrow knows(y,x)$
1.11	$currentProject(x,z) \wedge currentProject(y,z) \Rightarrow knows(x,y)$

The first two rules define *knows* as a transitive and symmetric property, respectively, while the last rule states that if two persons work on the same project, they probably know each other. Weights were learned generatively with all the individuals available. To better describe the results of the link prediction, we developed a simple artificial example composed by 9 users and 3 projects. Next, using MC-SAT, we queried for the conditional probabilities of the *foaf:knows* property for all those users. A graphical representation of the example, accompanied by a table with the results, is provided.

Fig. 1. Graphical representation of the artificial example. Users are represented by circles (A-I) and projects by squares (P1-P3). Black directed edges represent the *foaf:knows* relation, while gray undirected edges represent the *foaf:currentProject* relation.

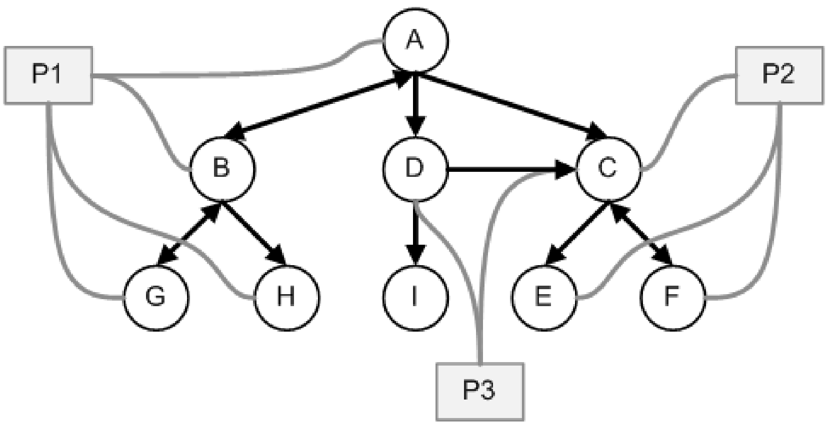

Table 3. *foaf:knows(x,y)* results of the previous example. Columns represent the *x*, lines the *y* (e.g., *P(foaf:knows(A,G))* = 0.90).

	A	B	C	D	E	F	G	H	I
A	0.83	0.97	0.95	0.93	0.48	0.50	0.91	0.82	0.47
B	1.00	0.97	0.48	0.50	0.44	0.43	1.00	0.98	0.48
C	1.00	0.49	0.93	1.00	0.97	1.00	0.44	0.46	0.47
D	1.00	0.50	0.98	0.84	0.47	0.48	0.50	0.50	0.92
E	0.46	0.44	1.00	0.48	0.86	0.86	0.45	0.43	0.41
F	0.48	0.43	1.00	0.48	0.85	0.86	0.43	0.43	0.41
G	0.90	1.00	0.46	0.49	0.45	0.43	0.84	0.89	0.42
H	0.83	1.00	0.49	0.51	0.43	0.45	0.89	0.83	0.45
I	0.49	0.45	0.46	1.00	0.40	0.42	0.42	0.44	0.59

Some interesting results can be seen in this example:

- *knows(A,G)* is greater than *knows(A,F)*, even if both users are at the same distance from *A*. The only difference between them is that *G* works in the same project than *A*, getting a bigger probability;
- *knows(D,A)*, *knows(C,A)*, and *knows(C,D)* have big probabilities, mostly because the symmetry of *knows*. However, the probability of *knows(C,D)* is the greatest, since both users also work in the same project, *P3*;
- Since *H* and *F* doesn't share any direct connection, the probability of *knows(H,F)* is low, but not null.

5.2.2 Link-based Classification

The main task in link-based classification [15] is to predict the category of an object based on the relations of that object with other objects. In our domain, there are three groups of users related to the experience of the user in the community: *Apprentice*, *Journeyer*, and *Master*. These groups are expressed through the *foaf:member* property. The objective of this experiment is to predict each user's group based on their connections to other users. For this purpose, we defined another simple rule that uses the relationship between users expressed on the three previous rules:

Table 4. Link-based classification rule.

Weight	Formula
0.19	$knows(x,y) \wedge member(x,z) \Rightarrow member(y,z)$

This rule states that the group of a user is influenced by the groups of the users that he knows. The weight of the rule was learned generatively in conjunction with the three rules of the previous experiment (their weights remained very similar). Next, we extracted a random sub-network composed by 172 users (11 Apprentices, 55 Journeyers, 93 Masters) and 54 projects and randomly removed the group information to 27% of the users (i.e., 47 users). With the rules of Table 2 and Table 4 and the sub-network individuals, we used MC-SAT to predict the membership of the missing group users. The results can be seen in the next table.

Table 5. Link-based classification results. Between brackets is the number of individuals of the group.

Group	Specificity	Precision	Recall	F-measure
Apprentice (4)	0.98	0	0	0
Journeyer (15)	0.97	0.83	0.33	0.48
Master (28)	0.37	0.7	1	0.82
Weighted Avg	0.61	0.68	0.70	0.64

Good results can be achieved on predicting user's groups taking only in account the relational structure of the network. The bad results on predicting the *Apprentice* group are probably derived from the small number of elements of that group in the test network. The results could be probably improved if other non-relational information about users was provided (e.g., nationality, age, sex).

5.2.3 Link-based cluster analysis

In the last experiment, we had seen how to classify users in a set of predefined groups. However, in some cases, the information about groups is not available and we still need to segment the users. The goal of link-based cluster analysis [15] is to cluster objects into groups that show similar relational characteristics. In our domain, it is interesting to cluster users given their acquaintances with other users. For this task, we can use the three rules presented in the link prediction task, since they can gave us a relational matrix of the *foaf:knows* property for all the users (i.e., the probability of all the users know each other). Using the same sub-network of the last task (172 users and 54 projects), we used MC-SAT with the previously referred rules to predict the *foaf:knows* property for all the 172 users. With those results, we applied two distinct clustering techniques: the general purpose *k-means* clustering algorithm [16], and the *Markov Cluster Algorithm*[15] (MCA) [17], an unsupervised graph clustering algorithm.

After some initial experimentation, we defined the number of desired clusters in the k-means algorithm to 3, and the *inflation* property of the MCA to 1.6 (which also produces 3 clusters). Since the initialization of cluster centroids in k-means is random, the algorithm was run 100 times and the best solution is the one presented. Table 6 provides the cluster sizes and the number of shared members between solutions.

Even if the underlying techniques are conceptually distinct, both solutions provide similar clusters, both in size and composition. The biggest clusters from both solutions (*C1* and *K1*) are very similar, as well the second biggest clusters (*C2* and *K2*).

[15] http://micans.org/mcl/

Table 6. Link-based clustering analysis results. The table represents the number of shared members between the clusters of the two algorithms (e.g., cluster C2 and K2 share 25 individuals). Between brackets is the size of each cluster.

		K-means		
		K1 (114)	K2 (47)	K3 (11)
MCA	C1 (135)	**102**	22	11
	C2 (30)	5	**25**	0
	C3 (7)	7	0	**0**

6 Future Work

During our experimentations, we had some problems in finding interesting ontologies with a sufficient number of individuals that allowed learning the weights with some confidence in the results. This is mainly due to the fact that a large number of Semantic Web ontologies currently available were made to model pure terminological domains, with the main objective of answer questions about concepts and not individuals. In these ontologies, we have to find other ways of gathering information to learn the uncertainty of the axioms. We identified four main approaches to tackle this problem:

- *Learn individuals.* This is the task studied in the field of ontology population [18]. By using previously trained classifiers or general syntactic rules, we can extract information about ontology individuals and their relations from textual corpus. Other way of populating ontologies is through the analysis of structured data, like relational databases or other ontologies. In this case, mappings [19] must be made between the structured data objects and the entities of the ontology.
- *Learn the uncertainties directly from textual corpus.* This is done by analyzing textual corpus for patterns like “70% of A is B” or “Most of the A’s are B’s”. This can be done again by using previously trained classifiers or general syntactic rules.
- *Use the structure of the ontology.* The structure of the ontology can provide interesting information about the uncertainty of its axioms. Some other works [20] [21] already explored similar approaches in ontologies, however with distinct objectives than ours. The field of network analysis [22] can provide us with some interesting concepts that can be potentially transferred to our specific case.
- *Collective learning of weights.* The idea is to learn the weights collectively from multiple ontologies about the same domain. This task can be achieved by exploring techniques from collective learning fields, like relational reinforcement learning [23].

7 Conclusions

In this paper, we have described an approach to the use of uncertainty reasoning in the Semantic Web using Markov Logic. We have shown how it can be used in practice to perform reasoning with OWL2 ontologies. Our approach enables the reasoning with uncertainty in the Semantic Web with a scalability factor that current tools do not provide. The presented work also addresses an important research question: how to derive uncertainty information from an ontology. The approach for producing this information is based on Markov Logic abilities to represent the world uncertainty. We think that our work constitutes a step forward in the creation of robust reasoning mechanisms for the Semantic Web.

References

[1] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," *Scientific American*, vol. 284, 2001, pp. 28-37.
[2] T. Lukasiewicz and U. Straccia, "Managing Uncertainty and Vagueness in Description Logics for the Semantic Web," *Web Semantics Sci Serv Agents World Wide Web*, 2008.
[3] P. Domingos, S. Kok, D. Lowd, H. Poon, M. Richardson, and P. Singla, "Markov Logic," *Probabilistic Inductive Logic Programming*, 2008, pp. 92-117.
[4] J. Hendler, "Agents and the Semantic Web," *IEEE Intelligent Systems*, vol. 16, 2001, pp. 30-37.
[5] B.C. Grau, I. Horrocks, B. Motik, B. Parsia, P. Patel-Schneider, and U. Sattler, "OWL 2: The next step for OWL," *Web Semantics: Science, Services and Agents on the World Wide Web*, 2008.
[6] F. Baader, D. Calvanese, D.L. McGuinness, D. Nardi, and P.F. Patel-Schneider, *The Description Logic Handbook: Theory, Implementation, and Applications*, Cambridge University Press, 2007.
[7] D. Roller, N. Friedman, L. Getoor, and B. Taskar, "Graphical Models in a Nutshell," *Introduction to Statistical Relational Learning*, The MIT Press, 2007, pp. 13-55.
[8] J. Besag, "Statistical analysis of non-lattice data," *The Statistician*, vol. 24, 1975, pp. 179-195.
[9] H. Poon and P. Domingos, "Sound and Efficient Inference with Probabilistic and Deterministic Dependencies," *Proceedings of the National Conference on Artificial Intelligence*, Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2006, pp. 458-463.
[10] W. Wei, J. Erenrich, and B. Selman, "Towards Efficient Sampling: Exploiting Random Walk Strategies," *Proceedings of the National Conference on Artificial Intelligence*, San Jose, CA: AAAI Press, 2004, pp. 670-676.
[11] S. Kok, P. Singla, M. Richardson, and P. Domingos, *The Alchemy system for statistical relational AI*, Technical report, Department of Computer Science and Engineering, University of Washington, Seattle, WA, 2007.
[12] E. Sirin, B. Parsia, B.C. Grau, A. Kalyanpur, and Y. Katz, "Pellet: A practical OWL-DL reasoner," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 5, 2007, pp. 51-53.
[13] P. Klinov, "Pronto: A Non-monotonic Probabilistic Description Logic Reasoner," *The Semantic Web: Research and Applications*, 2008, pp. 822-826.
[14] L. Getoor and C.P. Diehl, "Link mining: a survey," *SIGKDD Explor. Newsl.*, vol. 7, 2005, pp. 3-12.

[15] L. Getoor, “Link mining: a new data mining challenge,” *SIGKDD Explor. Newsl.*, vol. 5, 2003, pp. 84-89.
[16] J. Marques de Sá, *Pattern Recognition: Concepts, Methods and Applications*, Springer, 2001.
[17] S. Van Dongen, “Graph Clustering by Flow Simulation,” PhD Thesis, University of Utrecht, 2000.
[18] P. Cimiano, *Ontology Learning and Population from Text: Algorithms, Evaluation and Applications*, Springer, 2006.
[19] J. Euzenat and P. Shvaiko, *Ontology Matching*, Springer, 2007.
[20] P. Gomes, “Software Design Retrieval Using Bayesian Networks and WordNet,” *Advances in Case-Based Reasoning*, 2004, pp. 184-197.
[21] H. Stuckenschmidt and M. Klein, “Structure-Based Partitioning of Large Concept Hierarchies,” *The Semantic Web – ISWC 2004*, 2004, pp. 289-303.
[22] U. Brandes and T. Erlebach, *Network Analysis: Methodological Foundations*, Springer, 2005.
[23] P. Tadepalli, R. Givan, and K. Driessens, “Relational Reinforcement Learning: An Overview,” *Proceedings of the ICML'04 Workshop on Relational Reinforcement Learning*, 2004, pp. 1–9.