Towards Definition and Composition of Uncertain RESTful Resources

Pierre De Vettor[1], Michaël Mrissa[1], and Djamal Benslimane[1]

Université de Lyon, CNRS
LIRIS, UMR5205, F-69622, France
Lyon, France
{pierre.de-vettor, michael.mrissa, djamal.benslimane}@liris.cnrs.fr

Abstract. Nowadays, huge quantities of data are produced and published on the Web, coming from individuals, connected objects, and organizations. Uncertainty happens when combining data from different sources that contain heterogeneous, contradictory, or incomplete information. Today, there is still a lack of solutions in order to represent uncertainty that appears on the Web. In this paper, we introduce the concept of uncertain RESTful resource and propose a model and an algebra to interpret such resources.

Keywords: data uncertainty, RESTful resource, Hypertext composition, data combination

1 Introduction

Nowadays, individuals, organizations, and connected objects produce and publish a huge amount of data on the Web [12], through APIs and public endpoints [15], which is then combined into mashups [4] to produce high valuable new data. In this context, data uncertainty may occur as data comes from heterogeneous, contradictory, or incomplete sources [11]. In this case, there is a chance that each data source provides different information, which may be correct under some circumstances, and incorrect under others. Instead of choosing a unique version, yet arbitrary, of information, we believe users should be given the whole spectrum of possibilities to describe an entity.

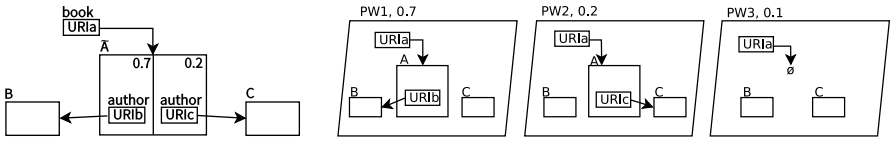
The main objective of this paper is to propose a theoretical framework for describing, manipulating, and exposing uncertain data on the Web. We present a model to define and interpret **uncertain Web resources**. We define an interpretation model and an algebra to compute uncertainty in the context of classical hypertext navigation and in the context of data query evaluation. The paper is structured as follows: Section 2 describes our uncertainty model and interpretation. Section 3 explains how we interpret query evaluation in this uncertainty-aware context. Section 4 presents our implementation details and evaluation. Section 5 presents other approaches that handle uncertainty. Finally, section 6 concludes and presents some future work.

2 Uncertain Web Resources

The semantics of uncertain Web resources can be explained based on the theory of possible worlds [17]. In our view, an uncertain resource has several possible representations which can potentially and individually be interpreted as true. These possibilities can be interpreted as a set of possible worlds (PW_1 ,..., PW_n) with a probability $prob(PW_i)$. We call them **possible Webs**, and inside these possible Webs, data is considered as certain. Based on the definition of Web resources [10], a Web resource is an entity or object, identified by an URI, accessible via HTTP methods. We define an uncertain Web resource $\widetilde{R}$ as follows:

$$\widetilde{R} =< uri_r, \{< rep_i, P_i > | i \in [1, n], \sum_{i=1}^{n} \left(P_i\right) \leq 1\} >$$

Where rep_i are the possible representations of $\widetilde{R}$. Since multiple representations of a resource cannot coexist at the same URI, these representations are mutually exclusive, and we have $P_i \in]0; 1]$. Having $\sum_{i=1}^{n} \left(P_i\right) \leq 1$ indicates that other representations may exist but their actual content is unknown (or does not exist). As an example, Fig. 1a shows that the two possible representations of our book resource generate three Webs in which representations are certain. We rely on the popular uncertain database model *Block-Independent Disjoint* (BID) [6] to define the following: *every resource is independent, and each URI identifies a unique resource, whose representation are disjoint, i.e., only one representation is true at a time*. Our model specifies that (1) possible resource representations are disjoint and (2) resource interpretations are independent from each other. Fig. 1a shows how we interpret uncertain resources as a set of probable representations with a probability (number in upper right), generating possible Webs in which this representation is true and unique. In possible Web $PW1$, resource A has one representation which contains a link to B; resource C exists but is not connected to A. In possible Web $PW3$, the uncertain unknown resource $\widetilde{A}$ has no existing representation. In this paper, an unknown resource is noted $\emptyset$. Technically, a GET request over such a resource leads to an HTTP error, such as a `404 not found` error.



(a) composition (b) generated worlds

Fig. 1: Uncertain Resource Example 1

2.1 HTTP request over uncertain resources

In this subsection, we introduce the notion of uncertainty-aware client, which is a client who is able to manipulate uncertain resources. In order to respect the Web principles, and to adapt to every client, we rely on *content negotiation*. Content negotiation is an HTTP mechanism that allows to serve different *versions* of the same resource representation (i.e., at the same URI), to fit with the client. Doing so, the client who does not know, or does not care, how to process uncertain resources, can receive a certain (but arbitrary) version of the resource representation[1]. In this paper, we make a difference between classical and uncertainty-aware GET requests. We propose the notation $\widetilde{GET}$ to describes a GET request from an uncertain-aware client. Let $\widetilde{R}$ be an uncertain resource deployed at uri_r, we defined the following expected behaviors:

$$\widetilde{GET}(uri_r) := \{< rep_1, P_1 >, \ldots, < rep_n, P_n >\}$$

In case, where the client performs a $\widetilde{GET}$ request over a certain resource, the response will provide the representation with a probability of 1. In our approach $\widetilde{GET}$ is **not** defining a new HTTP method. $\widetilde{GET}$ acts as a standard GET with a specific HTTP header which we define in Section 4 as $X - Accept - Uncertain : true$. We choose to define a specific header to avoid interference with the standardized usage of the *accept* header. Indeed, the *Accept* header is the classical header for content negotiation, as it is used to specify an expected mime-type for the resource representation. The good practice is then to specify an adhoc specific header to respect the HTTP standards (see *RFC7231* [2]).

2.2 Composing uncertain Web resources

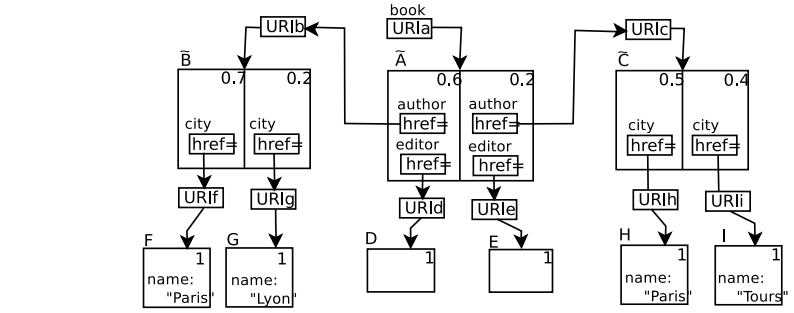
In a composition of Web resources, each combinaton of possible resource representations generates a new possible Web PW_x, whose probability is computed as follows:

$$P(PW_x) = \prod_{i \in [1,n]} \Big(prob(rep_i) \Big)$$

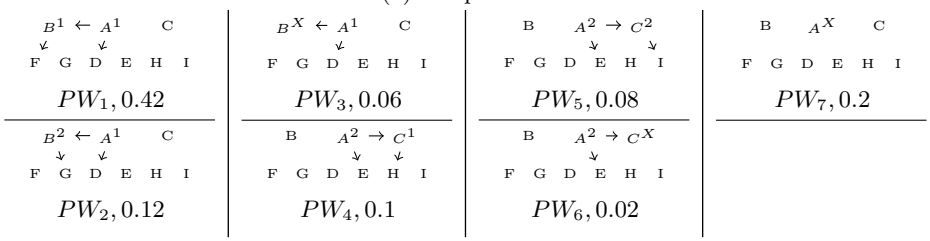
where $rep_i \in Card(PW_x)$, and $Card(PW_x)$ being the representations involved in PW_x. The probability of the unknown representations of a resource R_a is computed as follows: $prob\big(rep_a^x\big) = 1 - \sum_{i=1}^{n} prob(rep_a^i)$ where rep_a^i are the different representations of resource R_a. Fig. 2a shows a more complex example, where resources are certain and uncertain, generating the possible Webs shown in Fig. 2b. As an example, the probability of possible Webs PW_4 is $prob(PW_4) = prob(A^2) \times prob(C^1) \times prob(H) \times prob(E) = 0.2 \times 0.5 \times 1 \times 1 = 0.1$. In the next section, we describe how to interpret and compute a query in an uncertain composition.

[1] NB: how providers define the certain representation of an uncertain resource is not a problem we address in the scope of this paper. We only provide the possibility to do it

[2] https://tools.ietf.org/html/rfc7231#section-5.3.2



(a) composition



(b) generated Webs

Fig. 2: Uncertain Resource Example 2

3 Query as Resource Paths: Definition and Assessement

In this section, we present our approach to aggregate data from uncertain resources thanks to hypertext navigation. Formally, we define a data query as an ordered set of resource requests, following the same path through the different generated possible Webs. Each Web will provide a unique result, which are then aggregated. Generating each of these possible Webs, i.e., combining and storing each combination in memory to compute the query in each one, is a time and memory-consuming task.

When dealing with uncertain resources, we follow our query path through the possible resource representations. This navigation creates a possibility tree pattern, where branches are possible Webs associated with their probability. Fig. 3 shows the tree pattern created from our book scenario.

We propose an algorithm, cf. *Algorithm 1*, to compute resulting probabilities without possible Web generation. This algorithm implements an operator, which we call GET_p, who follows a stage-by-stage routing inside the possibility tree.

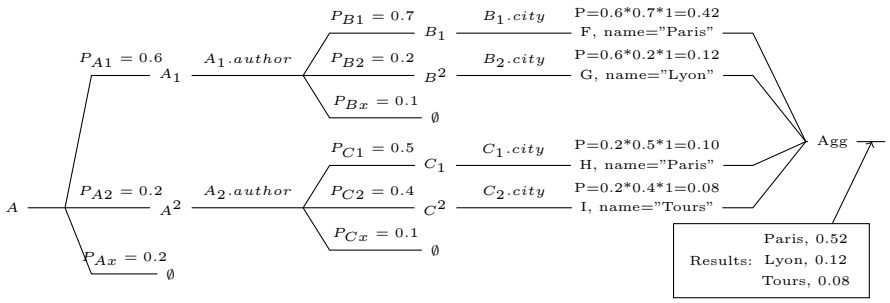


Fig. 3: Generating tree pattern while navigating resources

Algorithm 1 GETp Algorithm

```
1: procedure GETp( input_uris : list of (URI,proba) couple )
2:     results ← List()
3:     for all (URI_i,prob_i) ∈ input_uris do
4:         R̃ ← GET̃(URI_i)
5:         for all (representation, prob_r) ∈ R̃ do
6:             //Compute current probability
7:             prob_c ← prob_i * prob_r
8:             if representation ∉ results then
9:                 results.add( < representation, prob_c > )
10:            else
11:                results.update( representation, prob_c )
       return results
```

GET_p takes as input a list of URIs from an n^{th} stage of the tree, and returns the possible resource representations from the $(n+1)^{th}$ stage. The GET_p operator executes the necessary sequence of HTTP requests over the given URIs, applies the probability formula and returns the set of representation-probability couples.

As an example, we have a list of author URIs, extracted from possible book representations, each with a probability. GET_p gives us the possibility to retrieve the representation of each authors (with their probabilities) and to apply book probabilities to them. This will produce a set of author representations with *global* probabilities. The mutually exclusive status of representations guarantees a safe composition, which means resulting probabilities are coherent and their sum does not exceed 1. Finally, our computation algorithm, see *Algorithm 2*, uses GET_p to recursively process through the different stages of the probability tree. According to a query, and the URI of the first resource, our algorithm processes its way through the resource path, using object properties to find its way. In the end, the resulting data set contains all the values with their probabilities.

4 Implementation and Evaluation

We proposed an implementation for the GET_p algorithm and the computation algorithm. Here is an example of an HTTP request, using content negotiation, to an uncertain resource:

Algorithm 2 Computation Algorithm

```
1: procedure COMPUTE( query, URI_0 )
2:     transform query in lists of properties // the path
3:     // Make the first call / first URI is certain
4:     result ← PROCESS_PATH( properties, < URI_0, 1 > )
5: procedure PROCESS_PATH( properties, input_uris )
6:     // Retrieve the next set of resources descending the path
7:     rep ← GETp( input_uris )
8:     // Stop condition, no more properties = end of the path
9:     if properties[0] = ∅ then
10:        return rep
11:    else
12:        new_uri_list ← []
13:        for all (representation, prob_r) ∈ rep do
14:            if representation[properties[0]].type == URI then
15:                // Get the property and add it to the new list
16:                new_uri_list[]←[representation.getprop(properties[0]), prob_r]
17:        properties.remove(0)
18:        return PROCESS_PATH( properties, new_uri_list )
```

```
curl --header "X-Accept-Uncertain: true" "http://uri/resource"
```

In order to keep our approach reusable, and to allow integration with other RESTful approaches, we implemented the GET_p and $COMPUTE$ algorithms as RESTful services. Service calls are made through POST, and GET retrieves a user-friendly description of the service. We propose a Web interface to execute simple SPARQL queries. Our prototype, resources and scenarios are publicly available for testing at the following URL: `http://liris.cnrs.fr/~pdevetto/uncert/index.php`.

In order to evaluate our approach, we focus on processing time of our algorithms. For this purpose, we hosted RESTful services serving uncertain Web resources in JSON-LD [14] over linked data dumps from the SWDF corpus (`http://data.semanticweb.org`), representing ESWC2015, ISWC2013, and WWW2012 conference semantic data (author, proceedings, etc.). We created three different scenario (use case workflows) involving a different amount of resources and with different graph complexities. Starting from an *inproceeding article*, the first workflow retrieves all the *articles* that share the same *keywords*. The second workflow retrieves all the *articles* written by at least one same *author*. Finally, third workflow retrieves the *authors* that have written at least one *article* with one similar *keyword*. We executed all the workflows with 30 different inproceedings articles as input data. In our evaluation, we evaluate the ratio of network latency in the total execution cost of a workflow. We show that the processing cost of our solution is negligible compared to the network cost. Under a global execution time of 2 seconds, processing time is less than 5%. After 3 seconds, it never exceeds 1%.

5 Related Work

In this section, we present several approaches that handle data uncertainty, formerly in databases, and more recently in data services. In the context of databases, existing approaches can rely on the notion of containment [1], or overlapping [5] in order to create a mediated uncertain schema to overlap the source schemas. They can also rely on a generalization of by-table semantics [8], to improve data exchange in presence of uncertainty by proposing a probabilistic matching [9]. Even if these approaches apply very well to databases, they do not fit when working in the context of Web resources. In another context, several approaches have been proposed to work with uncertainty when combining data from Web sources. These approaches rely on mediated schemas [7], or probabilistic XML [13,3] to confront and merge pieces of information from heterogeneous sources. *Pivert and Prade* [16] propose a solution to integrate multiple heterogeneous sources, resolving factual inconsistencies by analyzing the existence of suspects answers in both data sets. Finally, *Amdouni et Al.* [2] rely on the possible world theory [17] to propose an approach to handle the uncertainty of the data returned by data services, which they call uncertain data services. These works propose several methods and models to process uncertainty in the context of the Web (XML, services, or semantics), but none of them address the uncertainty that can appear while referencing or browsing information through the Web. This is a very common problem, which is usually skipped or decided arbitrarily by providers. Our approach proposes a relevant and adaptable approach to enhance Web-based applications with uncertainty awareness.

6 Conclusion

In this paper, we address the need for a solution to handle data uncertainty while referencing and navigating resources on the Web. We propose a model for uncertain Web resources, as resources which may have several mutually exclusive representations with probabilities. On top of that, we propose an algebra to interpret and evaluate data query in uncertain resource compositions.

Future work includes opening our approach in order to deal with more complex scenarios, where possible representations could be actual Web resources with URIs. This way, we could construct a model based on hypertext navigation to define a resource according to a set of others, giving a possibility to represent the probable equivalence of resources.

References

1. P. Agrawal, A. D. Sarma, J. Ullman, and J. Widom. Foundations of uncertain-data integration. *Proc. VLDB Endow.*, 3(1-2):1080–1090, Sept. 2010.
2. S. Amdouni, M. Barhamgi, D. Benslimane, and R. Faiz. Handling uncertainty in data services composition. In *Services Computing (SCC), 2014 IEEE International Conference on*, pages 653–660, June 2014.

3. M. Ba, S. Montenez, R. Tang, and T. Abdessalem. Integration of web sources under uncertainty and dependencies using probabilistic xml. In W.-S. Han, M. L. Lee, A. Muliantara, N. A. Sanjaya, B. Thalheim, and S. Zhou, editors, *Database Systems for Advanced Applications*, Lecture Notes in Computer Science, pages 360–375. Springer Berlin Heidelberg, 2014.
4. D. Benslimane, S. Dustdar, and A. P. Sheth. Services mashups: The new generation of web applications. *IEEE Internet Computing*, 12(5):13–15, 2008.
5. R. Cheng, J. Gong, D. Cheung, and J. Cheng. Evaluating probabilistic queries over uncertain matching. In *Data Engineering (ICDE), 2012 IEEE 28th International Conference on*, pages 1096–1107, April 2012.
6. N. Dalvi, C. Re, and D. Suciu. Queries and materialized views on probabilistic databases. *Journal of Computer and System Sciences*, 77(3):473 – 490, 2011. Database Theory.
7. A. Das Sarma, X. Dong, and A. Halevy. Bootstrapping pay-as-you-go data integration systems. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 861–874, New York, NY, USA, 2008. ACM.
8. X. Dong, A. Y. Halevy, and C. Yu. Data integration with uncertainty. In *Proceedings of the 33rd international conference on Very large data bases*, pages 687–698. VLDB Endowment, 2007.
9. R. Fagin, B. Kimelfeld, and P. G. Kolaitis. Probabilistic data exchange. *Journal of the ACM (JACM)*, 58(4):15, 2011.
10. R. T. Fielding and R. N. Taylor. Principled design of the modern web architecture. In *Proceedings of the 22Nd International Conference on Software Engineering*, ICSE '00, pages 407–416, New York, NY, USA, 2000. ACM.
11. A. Halevy, A. Rajaraman, and J. Ordille. Data integration: The teenage years. In *Proceedings of the 32Nd International Conference on Very Large Data Bases*, VLDB '06, pages 9–16. VLDB Endowment, 2006.
12. T. Heath and C. Bizer. *Linked Data: Evolving the Web into a Global Data Space*. Synthesis Lectures on the Semantic Web. Morgan & Claypool Publishers, 2011.
13. B. Kimelfeld and P. Senellart. Probabilistic xml: Models and complexity. In Z. Ma and L. Yan, editors, *Advances in Probabilistic Databases for Uncertain Information Management*, volume 304 of *Studies in Fuzziness and Soft Computing*, pages 39–66. Springer Berlin Heidelberg, 2013.
14. M. Lanthaler and C. Gutl. On using json-ld to create evolvable restful services. In *Proceedings of the Third International Workshop on RESTful Design*, WS-REST '12, pages 25–32, New York, NY, USA, 2012. ACM.
15. M. Maleshkova, C. Pedrinaci, and J. Domingue. Investigating web apis on the world wide web. In *Web Services (ECOWS), 2010 IEEE 8th European Conference on*, pages 107–114, Dec 2010.
16. O. Pivert and H. Prade. Querying uncertain multiple sources. In U. Straccia and A. Cali, editors, *Scalable Uncertainty Management*, volume 8720 of *Lecture Notes in Computer Science*, pages 286–291. Springer International Publishing, 2014.
17. F. Sadri. Modeling uncertainty in databases. In *Data Engineering, 1991. Proceedings. Seventh International Conference on*, pages 122–131, Apr 1991.