Modeling and Composing Uncertain Web Resources

Pierre De Vettor(✉), Michaël Mrissa, and Djamal Benslimane

Université de Lyon, CNRS LIRIS, UMR5205, 69622 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: Sect. 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, Sect. 6 concludes and presents some future work.

H. Sack et al. (Eds.): ESWC 2016 Satellite Events, LNCS 9989, pp. 327–341, 2016.
DOI: 10.1007/978-3-319-47602-5_47

2 Uncertain Web Resources

In order to understand the notion of uncertain resource, it is first required to remind some background knowledge on underlying definitions. Uncertain resources are Web resources, and according to the principles of REST [10], a Web resource is an entity or object, identified by an URI. A resource can provide a single object, a set of sub-resources or an abstract notion, e.g. a concept from an ontology. In order to clarify Web resources, we propose the following notation, and define a resource R as follows:

$$R =< uri_r, rep_r >$$

where uri_r is the URI that identifies this and only this resource and rep_r is the representation of the resource R in the server. A resource can be accessed with the HTTP methods (*RFC 2616*[1]), allowing to read (GET), update or create ($POST$), delete ($DELETE$) resources on a domain. In this section, we overcome the definition of Web resources in order to handle data uncertainty.

2.1 Definition

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.

In order to define the notion of uncertain resource, we rely on several assumptions: **(1)** Due to the REST principles, several representation of one URI (i.e. one resource) cannot coexist, so the possible representations of a resource must be mutually exclusive. **(2)** In order to deal with uncertain resources, rather than with uncertain data, assume that in the statement inside a given possible representation, every piece of data is considered certain. By extension, if a resource property has several possible values, it should appear into separate representations. **(3)** For the sake of simplicity, in this approach we consider that each possible representation of a given resource is represented according to the same model.

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:

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

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 $P_i \in]0; 1]$. Probabilities are not part of representations, they are meta-data provided by the server.

[1] http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html.

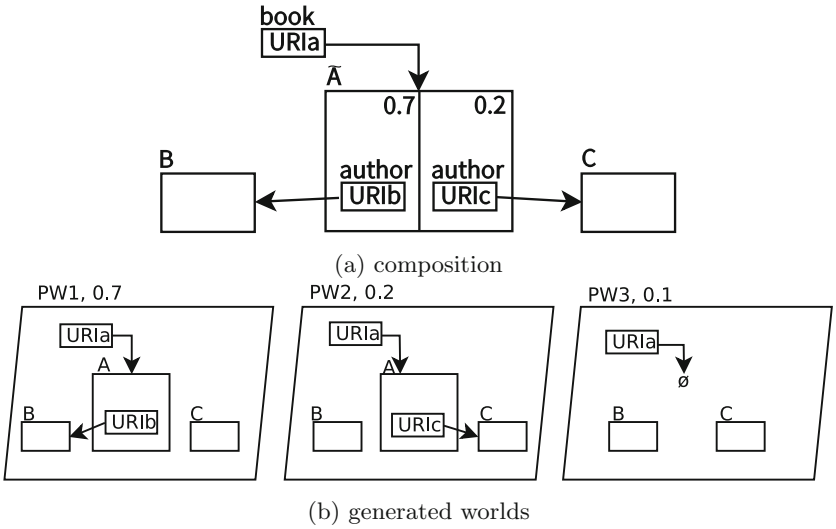


Fig. 1. Uncertain resource example 1

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. Figure 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.

2.2 Precision on Unknown Representations

Having $\sum_{i=1}^{n} (P_i) \leq 1$ indicates that a part of the resource is considered as unknown. In this paper, unknown resource representations are noted $\emptyset$. The resource can specify that other representations may exist but their actual content is left unknown. This kind of behavior can happen according to different factors, as an example, the provider could have left a part private or protected, for privacy reasons. Another possibility is that the resource no longer exist in the server, but the URI still points at it. Finally, a provider could have planned to create the resource, but had not already made it. This way, the URI points to something that does not yet exist.

In our example, shown in Fig. 1b, the possible Webs $PW1$, $PW2$, and $PW3$ are generated from the probable possible representations of the uncertain resource A. 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. Technically, a GET request over such a resource leads to an HTTP error, such as a `404 not found` or a `500 internal server` error.

2.3 Programmatic Representation of Uncertain Resources

In order to provide a way to handle uncertain resources, we proposed a formalism to physically represent them. Our mecanisms for programmatically representing an uncertain resource include all the possible representations and their associated probabilities.

Listings 1.1 and 1.2 shows the JSON representations of some uncertain resources.

```
[ {p:0.6, r:{ title :"Les miserables",
              date:"3 Avr 1862",
              author:"http://dbpedia.org/VHugo"}},
  {p:0.4, r:{ title :"Les miserables",
              date:"30 Mar 1862",
              author:"http://dbpedia.org/VictorHugo"}} ]
```

Listing 1.1. JSON representation of an uncertain book resource

```
[ {p:0.7, r:{ name:"Hugo V.",
              birth: 1802,
              city :"http://city/besancon"}},
  {p:0.2, r:{ name:"Victor HUGO",
              birth: 1802,
              author:"http://city/paris"}} ]
```

Listing 1.2. JSON representation of an uncertain author resource

2.4 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. 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 Sect. 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]).

2.5 A Certain Representation of an Uncertain Resource

In this approach, using content negociation provides us with a solution that allows to manipulate uncertain resources as classical certain resources. Doing so, a client who does not know how to process uncertain resources, or does not care about uncertainty representation, will still be able to receive a certain (but arbitrary) version of the resource representation. In this case, we choose to provide the most certain representation of the resource.

On top of that, it is also important to inform the client about the uncertain nature of the resource it is accessing. We provide two additionnal headers to enhance this certain representation with uncertain capabilities. On the one hand, we inform the client that the resource has an uncertain capacity, and is able to accept uncertain requests, by providing a $x - accept - uncertain$ in the response header. Additionnaly, we also provide the probability of the arbitrary selected representation, through the $x - uncertainty - value$. Listing 1.3 shows an example of certain (i.e., classical) GET response (only headers) over an uncertain resource.

```
curl -I http://liris.cnrs.fr/~pdevetto/uncert/index.php/df/paper/89

HTTP/1.1 200 OK
Date: Fri, 12 Jun 2016 13:35:00 GMT
Server: Apache/2.4.7 (Ubuntu)
X-Accept-Uncertain: 1
X-Uncertainty-Value: 0.225
Content-Type: application/json+ld
```

Listing 1.3. Enrichment of certain GET response over uncertain resources

NB: providers could also choose another method to define the given certain representation of an uncertain resource. This is only an approach we advocate in this paper. We only provide one possibility to do it.

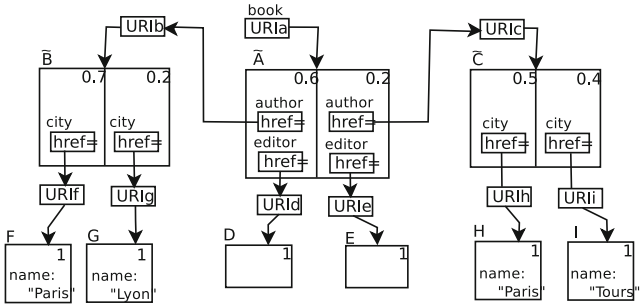
2.6 Composing Uncertain Web Resources

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:

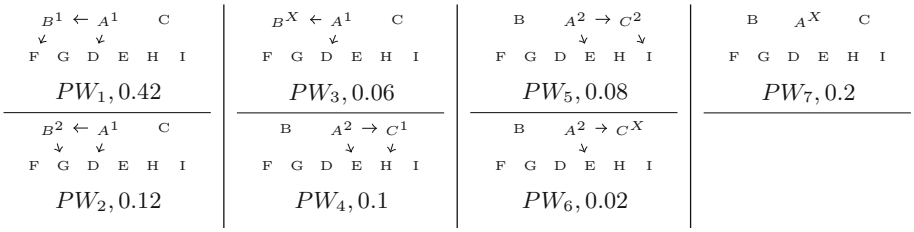
[2] https://tools.ietf.org/html/rfc7231#section-5.3.2.

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

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. Figure 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.



(a) composition

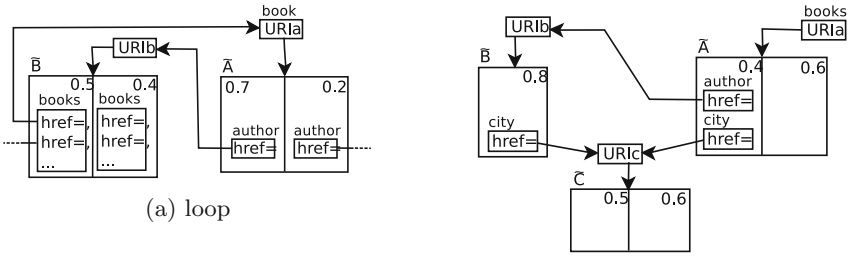


(b) generated Webs

Fig. 2. Uncertain resource example 2

2.7 Particular Composition Cases

Our previous examples show scenario based compositions which we could use to answer our query. The distributed state of resource-based applications, associated with our lightweight model, allow more complex compositions. Figure 3 shows some examples of complex resource orchestration where uncertainty appears, and to which our model could easily adapt. These examples include loop, redundancies and differences in models while navigating through hypertext.



(a) loop

(b) dependencies

Fig. 3. Uncertain resource examples

In these examples, heterogeneities can appear but are handled by our algorithm presented in the next section.

Figure 3a shows a situation where it may exist a loop in the request path. Our algorithm only dereferences the resource once, protecting us from looping infinitely through hypertext path. In Fig. 3b, the resource composition presents a duplicate resource. The important specificity here is related to this duplicate resource, and is handled by our model, which specifies that a resource only have one representation in a possible Web.

2.8 Uncertain Data vs. Uncertain Resources

In this section, we introduced the concept of uncertain Web resources, presented a model and an algebra to compute the probability of uncertain resource composition. One common question about this work, is: "How do you handle resource where representation also contains uncertain data?".

It is sometimes difficicult to make a difference between uncertain data and uncertain resources. Let a data set, containing a list of scientific articles, each of them having a list of authors URIs and a publication date, as presented in Table 1.

Table 1. Example of uncertain data set

Paper	Authors		Date	
#1	AuthorA & authorB	0.6	*April2016	1
	AuthorA & authorC	0.4		
#2	AuthorD & authorE	0.7	March 2015	0.6
	AuthorD &authorE & authorF	0.2	April 2015	0.4
	AuthorC & authorE	0.1		
#3	AuthorF & authorG	1	September 2015	0.7
			October 2015	0.2

In this example, each tabular cell can have several values each of them applied with a probability. Representing this uncertainty in resource description, would break the semantic of atomic resources, since our possible representations need to be able to be manipulated as certain representations in the possible Web they exist in.

In order to represent this uncertainty, what we propose, to fit with our model, is to generate each possible representation for each resource, and include all these representations in the resource. This way, we respect our model, and only manipulate finite resource representations. We then obtain a result set of resource representations, where we could compute probabilities by applying a product, of the data pieces involved. Our uncertain data set, generates a set of possible representations for each article as shown in Table 2.

From this example, we show that our results stays relevant and safe. As an example, paper #2 data have generated 6 possible representation whose probability sum is $0.42 + 0.28 + 0.12 + 0.08 + 0.06 + 0.04 = 1$. Each generated representation is a possibility of truth. They are all mutually exclusive.

Table 2. Generate possible representation from uncertain data set

Paper	Authors	Date	Computed
#1	AuthorA & authorB	April 2016	0.6 * 1 = 0.6
	AuthorA & authorC	April 2016	0.4 * 1 = 0.4
#2	authorD & authorE	March 2015	0.7 * 0.6 = 0.42
	AuthorD & authorE	April 2015	0.7 * 0.4 = 0.28
	AuthorD & authorE & authorF	March 2015	0.2 * 0.6 = 0.12
	AuthorD & authorE & authorF	April 2015	0.2 * 0.4 = 0.08
	AuthorC & authorE	March 2015	0.1 * 0.6 = 0.06
	AuthorC & authorE	April 2015	0.1 * 0.4 = 0.04
#3	AuthorF & authorG	September 2015	1 * 0.7 = 0.7
	AuthorF & authorG	October 2015	1 * 0.2 = 0.2

NB: This transformation method to generate uncertain resource from uncertain data set only work when data fields are independent, and data values are mutually exclusive, as our uncertain Web resource model specifies it. In case where there is a dependency between the different data parts, or where values can coexist, a different algebra applies. This is not part of the scope of this paper, and can be solved with help from concepts from related existing approaches such as [8,16].

Complexity of Our Method. Although our method generates a lot of possible representations, it does not specifically increase the complexity of the following approach to interpret hypermedia queries. The important notion we present in

the following, to prevent our algorithms for exponential growth in term of execution time, is the reduction step. At each stage of the process, we aggregate duplicates URIs and addition the probabilities. Doing so, it is easier to create several possible representations, since duplicate values will be grouped after probability computation.

3 Query as Resource Paths: Definition and Assessment

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. There is a need for an approach that allows to aggregate these results directly without having to generate the possible Webs.

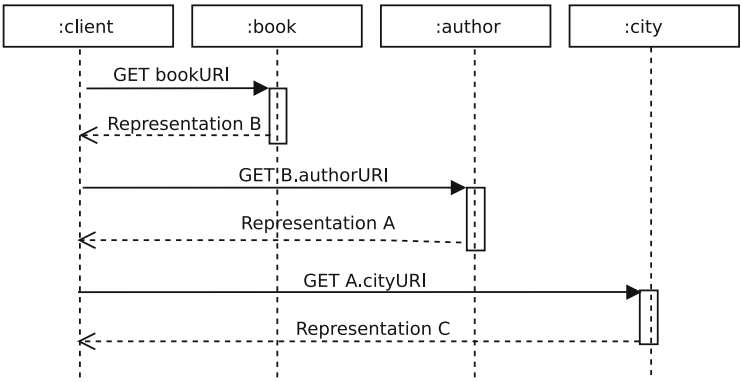


Fig. 4. Query answering in RESTful compositions

Following our example scenario, the query *What is the date and city of birth of the writer of the book "Les Miserables"?* The execution of this path in a classical RESTful composition is detailed in Fig. 4.

In order to follow this path, we must assume that resource representations specify the necessary semantics about their contents. As an example, when searching for the author of a book resource, the `author` functional property is required in order to complete. In our scenario, the semantically enhanced representations we manipulate are represented in the JSON-LD format [14]. 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. Figure 5 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.

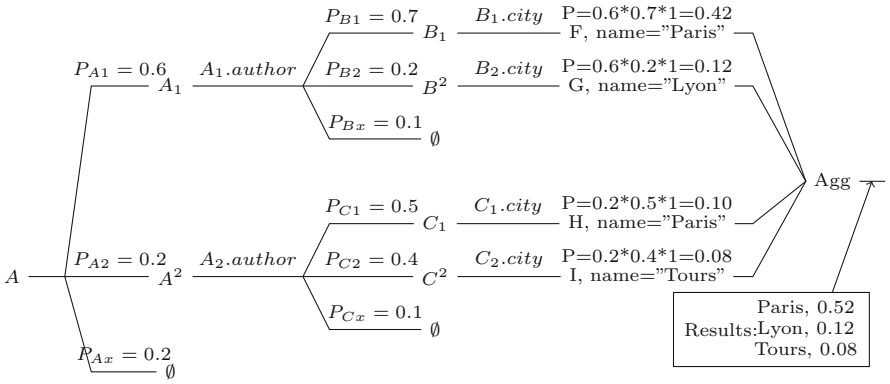


Fig. 5. Generating tree pattern while navigating resources

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 iteratively 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.

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:        next_uris ← []
13:        // For each couple (representation, probability)
14:        for all (rep_r, prob_r) ∈ rep do
15:            if rep_r.getprop(properties[0]).type == URI then
16:                // Get the property and add it to the new list
17:                next_uris[]←[rep_r.getprop(properties[0]), prob_r]
18:        properties.remove(0)
19:        return PROCESS_PATH( properties, next_uris )
```

4 Implementation and Evaluation

As introduced before, the principles of our approach rely on REST principles, so we are able to use any HTTP client to access our uncertain resources. What we propose here, is an implementation of the GET_p algorithm as well as an implementation of our evaluation algorithm, which will perform the necessary HTTP requests. As introduced before, we rely on content negotiation to specify that an HTTP client is able to understand uncertainty. Listing 1.4 shows an example of HTTP request, using content negociation to retrieve an uncertain resource.

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

Listing 1.4. Enrichment of certain GET response over uncertain resources

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. Listing 1.5 shows an example of HTTP request to call the GET_p service.

```
curl −X POST "{ domain }/op/getp"
    --data "uri0={ uri0 }&prob0={ prob0 }
            &uri1={ uri1 }&prob1={ prob1 }"
```

Listing 1.5. Enrichment of certain GET response over uncertain resources

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 scenarios (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*, shown in Listing 1.6.
- The second workflow retrieves all the *articles* written by at least one same *author*.
- Finally, the 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. Comparing uncertain workflow executions with the same workflow in a certain context has no meaning, because the number of HTTP request will grow exponentially.

```
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX al: <http://liris.cnrs.fr/~pdevetto/uncert/>
PREFIX swrc: <http://swrc.ontoware.org/ontology#>
SELECT ?similararticle WHERE {
    ? article  a swrc:InProceedings ;
          dc:subject  ?subject  .
    ?subject  dc: inarticle  ? similararticle   .
}
```

Listing 1.6. Query 1: Articles that share the same subject than another article (by ID)

Our *compute* algorithm implementation will transform this query in a list of concepts to extract from resource to resource, creating our path descending through the possibility tree. Our implementation uses the ARC2 SPARQL Parser to extract query concepts. 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. The obtained results show the following: while workflows become more complex, the number of HTTP request grows, and the processing time is more and more negligible, compared to HTTP latency. Under a global execution time of 2 s, processing time is less than 5 %. After 3 s, it never exceeds 1 %. On top of that, as long as input resources defines coherent representations (and correct probabilities), no matter the query, it always generates a safe result set, with relevant values and probabilities.

5 Related Work

Uncertainties have been processed in different contexts, we envision different approaches, handling uncertainty, historically in database world, and more recently in services oriented application. Unfortunately, none of these approach handles the uncertainty that can appear when manipulating resources or when dealing with Restful applications.

5.1 Uncertainty in Databases

Fagin et al. [9] envision data exchanges in presence of uncertain data coming from probabilistic databases. Their approach is a generalization of Dong et al. by-table semantics [8] in which probabilistic matching are generated between tables in order to align fields. The generated results are associated with a probability value. The *Fagin et al.* probabilistic approach relies on creating an arbitrary binary relationship between two countable (finite or countably infinite) probability spaces. *Agrawal et al.* [1] propose a local-as-view data integration approach dealing with sources containing uncertain data. Their approach rely on the concept of containment, which means creating a mediated uncertain schema that must contain both databases. *Cheng et al.* [5] propose an approach assuming that concept values from both schemas can overlap to deal with uncertain matchings.

These works are strong although complex approaches to handle with uncertain/probabilistic data, these approaches has inspired our definition of uncertain Web resource. However, if it applies very well to database, these approach does not fit well when working with Web resources. One solution could be to layer data sources with a database endpoint, but it could not provide a sufficient solution for considering our composition semantics. These approach has lead our definition of uncertain Web resources.

5.2 Uncertainties on the Web

Several approach have been proposed to deal with uncertainty in other contexts than databases, most of the time in order to propose heterogeneous data integration approach.

Lamine Ba et al. [3] propose an approach for data integration, combining data from web sources containing uncertainty and dependencies. Their approach confront and merge diverse information about a same subject from diverse web sources. They model the following data as probabilistic XML [13] to process decisions. *Sarma et. al.* [7] envision what they call pay-as-you-go integration systems, which is related to our smart data architecture. Their system rely on a single point interface to a set of data sources, integration of data being made by creating a mediated schema for the domain. *Pivert and Prade*[16] propose a solution to integrate multiple heterogeneous and autonomous information sources, resolving factual inconsistencies by analyzing the existence of suspects answers in both data sets. Their approach verify the data provided by two data source they want to integrate, if a data piece in second source invalidates a data piece

in the first data source, it is considered as a suspect answer. Their approach finally return all the candidate answers to a query, rank-ordered according to their level of reliability.

Finally, *Amdouni et al.* [2] propose an approach to handle the uncertainty of the data returned by data services, which they call uncertain data service. They define uncertainty at three levels, in the context of DaaS services, modelisation, invocation and composition. First of all, they extend the Web services standards to model uncertainty of a service in its own description. This model introduces the notion of uncertain data service, whose can be explained by possible worlds theory [17]. These uncertain data services are defined by their inputs and sets of their probable outputs. It is this set of possible outputs returned by an invocation which can be considered as possible worlds, each of these world being Dependant and having a probability value. They defined two different kinds of invocation of uncertain data services, conventional with certain input and probabilistic where inputs are presented containing uncertain data instances.

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. Agrawal, P., Sarma, A.D., Ullman, J., Widom, J.: Foundations of uncertain-data integration. Proc. VLDB Endow. **3**(1–2), 1080–1090 (2010)
2. Amdouni, S., Barhamgi, M., Benslimane, D., Faiz, R.: Handling uncertainty in data services composition. In: 2014 IEEE International Conference on Services Computing (SCC), pp. 653–660, June 2014
3. Ba, M., Montenez, S., Tang, R., Abdessalem, T.: Integration of web sources under uncertainty and dependencies using probabilistic XML. In: Han, W.-S., Lee, M.L., Muliantara, A., Sanjaya, N.A., Thalheim, B., Zhou, S. (eds.) Database Systems for Advanced Applications. LNCS, vol. 8505, pp. 360–375. Springer, Heidelberg (2014)

4. Benslimane, D., Dustdar, S., Sheth, A.P.: Services mashups: the new generation of web applications. IEEE Internet Comput. **12**(5), 13–15 (2008)
5. Cheng, R., Gong, J., Cheung, D., Cheng, J.: Evaluating probabilistic queries over uncertain matching. In: 2012 IEEE 28th International Conference on Data Engineering (ICDE), pp. 1096–1107, April 2012
6. Dalvi, N., Re, C., Suciu, D.: Queries and materialized views on probabilistic databases. J. Comput. Syst. Sci. **77**(3), 473–490 (2011)
7. Das Sarma, A., Dong, X., Halevy, A.: Bootstrapping pay-as-you-go data integration systems. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, pp. 861–874, New York, NY, USA. ACM (2008)
8. Dong, X., Halevy, A.Y., Yu, C.: Data integration with uncertainty. In: Proceedings of the 33rd International Conference on Very Large Data Bases, pp. 687–698. VLDB Endowment (2007)
9. Fagin, R., Kimelfeld, B., Kolaitis, P.G.: Probabilistic data exchange. J. ACM (JACM) **58**(4), 15 (2011)
10. Fielding, R.T., Taylor, R.N.: Principled design of the modern web architecture. In: Proceedings of the 22nd International Conference on Software Engineering, ICSE 2000, pp. 407–416, New York, NY, USA. ACM (2000)
11. Halevy, A., Rajaraman, A., Ordille, J., Data integration: the teenage years. In: Proceedings of the 32Nd International Conference on Very Large Data Bases, VLDB 2006, pp. 9–16. VLDB Endowment (2006)
12. Heath, T., Bizer, C., Data, L.: Evolving the Web into a Global Data Space. Synthesis Lectures on the Semantic Web. Morgan & Claypool Publishers, San Rafael (2011)
13. Kimelfeld, B., Senellart, P.: Probabilistic XML: models and complexity. In: Ma, Z., Yan, L. (eds.) Advances in Probabilistic Databases. STUDFUZZ, vol. 304, pp. 39–66. Springer, Heidelberg (2013)
14. Lanthaler, M., Gutl, C.: On using JSON-LD to create evolvable restful services. In: Proceedings of the Third International Workshop on RESTful Design, WS-REST 2012, pp. 25–32, New York, NY, USA. ACM (2012)
15. Maleshkova, M., Pedrinaci, C., Domingue, J.: Investigating web APIs on the world wide web. In: 2010 IEEE 8th European Conference on Web Services (ECOWS), pp. 107–114, December 2010
16. Pivert, O., Prade, H.: Querying uncertain multiple sources. In: Straccia, U., Calì, A. (eds.) SUM 2014. LNCS, vol. 8720, pp. 286–291. Springer, Heidelberg (2014)
17. Sadri, F.: Modeling uncertainty in databases. In: Proceedings of the Seventh International Conference on Data Engineering, pp. 122–131, April 1991