

SemSynX: Flexible Similarity Analysis of XML Data via Semantic and Syntactic Heterogeneity/Homogeneity Detection

Jesús M. Almendros-Jiménez¹(✉) and Alfredo Cuzzocrea²

¹ Informatics Department, University of Almería, Almería, Spain
jalmen@ual.es

² DIA Department, University of Trieste and ICAR-CNR, Trieste, Italy
alfredo.cuzzocrea@dia.units.it

Abstract. In this paper we introduce and experimentally assess *SemSynX*, a novel technique for supporting similarity analysis of XML data via semantic and syntactic heterogeneity/homogeneity detection. Given two XML trees, *SemSynX* retrieves a list of semantic and syntactic heterogeneity/homogeneity matches of objects (i.e., elements, values, tags, attributes) occurring in certain paths of the trees. A *local score* that takes into account the path and value similarity is given for each heterogeneity/homogeneity found. A *global score* that summarizes the number of equal matches as well as the local scores globally is also provided. The proposed technique is highly *customizable*, and it permits the specification of *thresholds* for the requested *degree of similarity* for paths and values as well as for the *degree of relevance* for path and value matching. It thus makes possible to “adjust” the similarity analysis depending on the nature of the input XML trees. *SemSynX* has been implemented in terms of a *XQuery library*, as to enhance interoperability with other XML processing tools. To complete our analytical contributions, a comprehensive experimental assessment and evaluation of *SemSynX* over several classes of XML documents is provided.

1 Introduction

Data fusion [4, 8] is a research topic of increasing interest, mainly motivated by the need of accessing multiple data sources in an integrated way. This requirement is becoming harder and harder in novel emerging contexts, such as challenging *Cloud Computing environments* (e.g. [19]), where Cloud entities not only need to *exchange data* but also to *combine data*, for a wide spectrum on advanced applications. Without loss of generality, the main data-fusion’s goal is to provide users with a *complete and concise view* of all the existent data (including *data objects* and *data entities*). “Concise” means that no object is represented twice and with so-called contradictions (e.g., [9]). “Complete” means that no object is forgotten in the final (integrated/fused) result.

Semantic and syntactic heterogeneity can be found in data sources and detecting them is the major goal of the *data-fusion research initiative* [4, 8].

In this respect, *schema matching* [23] and *duplicate detection* [27] are two well-studied mechanisms. In addition to this, *data-conflict detection* and *data repairing/cleaning* strategies have been proposed in the data-fusion research area (see [4, 12, 13, 15, 29] as some authoritative examples) in order to deal with semantic heterogeneity, while syntactic heterogeneity has been typically addressed by *schema matching approaches* (e.g., [9]). Data heterogeneity has also been investigated via using *Ontologies* [21, 22], whose main functionality is that of supporting semantics-based description of data sources. In this research area, *approximate data instance matching* [14] aims at providing a measure of *data similarity*, which is later exploited to identify multiple instances of real-word (data) objects based on suitable *distance functions*. *XML data clustering* [1, 2, 25] is also tightly connected to the data-instance matching problem, and the general goal here consists in classifying target XML documents in groups of similar content. Applications of XML similarity also include *data warehousing version control* (e.g., [11]) and *change management* (e.g., [24]).

In order to be able of integrating data from multiple sources, a typical strategy consists in, first, identifying (data) objects to be integrated (usually, by means of unique keys), and, secondly, mixing attributes in the integrated data repository. Nevertheless, attributes of a certain object can have different names, even representing the same concept, or they can have the same names, even representing so-called *conflicting values* (e.g., [9]). In the first case, an *attribute name mapping* is required, while in the second one, a *conflict resolution strategy* is adopted. Strategies for conflict resolution range from the *most-frequent value* and the *most-trusted source* selection to more intuitive *date-based* selection. Also, values represented as strings can be compared via applying well-known *string comparison algorithms* (e.g., [18, 28]).

In our research, we focus on *XML data* specially, for which several data fusion strategies have been proposed (e.g., [8, 12, 16, 21, 26]). XML data require specific mechanisms for data fusion since they are semi-structured data represented by *tagged trees* where leaf nodes have textual content. *Graph-based data*, e.g. *RDF data*, are also subject of recently-proposed targeted data fusion mechanisms (e.g., [5, 20, 30]), and they expose several “touching points” with the related XML data fusion research area. It is also significant to highlight that, in the semi-structured context (both XML and RDF data), conflicts of attribute values are more sophisticated since the same real-world (data) objects can be represented using different tree/graph structures. This makes the investigated problem harder.

Following the so-delineated research area, in this paper we introduce and experimentally assess **SemSynX**, a *novel technique for supporting similarity analysis of XML data via detecting semantic and syntactic heterogeneity/homogeneity matches of objects* (i.e., elements, values, tags, attributes) *occurring in certain paths of the two (XML) trees* that model the input data sources. **SemSynX** retrieves a list of similarity/dissimilarity matches found in the target objects as well as a measure of similarity of the XML trees expressed by a *score*. A *local score* that takes into account path and value similarity is provided for each heterogeneity/homogeneity found. A *global score* that summarizes the

number of equal matches as well as the local scores is also provided. Such a list of similarity/dissimilarity matches is suitable to support *automatic definition of schema mapping* and *conflicting strategies* of the target XML data. In addition to this, **SemSynX** is highly *customizable*, and it permits the specification of *thresholds* for the required *degree of similarity* for paths and values as well as for the *degree of relevance* for path and value matching, hence finally it makes possible to “adjust” the comparison of input XML trees depending on the nature of the trees themselves. Furthermore, in order to improve matching in some specialized scenarios, *selection of paths* to be matched between documents is supported, while assigning suitable *weights* to selected paths. *Semantics-based approach* has also been adopted in order to *compare label content*, thus introducing more precision during the matching of similar items characterized by a specific semantics. We complete our analytical contributions by means of a comprehensive experimental assessment and evaluation of **SemSynX** over several classes of XML documents. A preliminary version of this paper appears in [3].

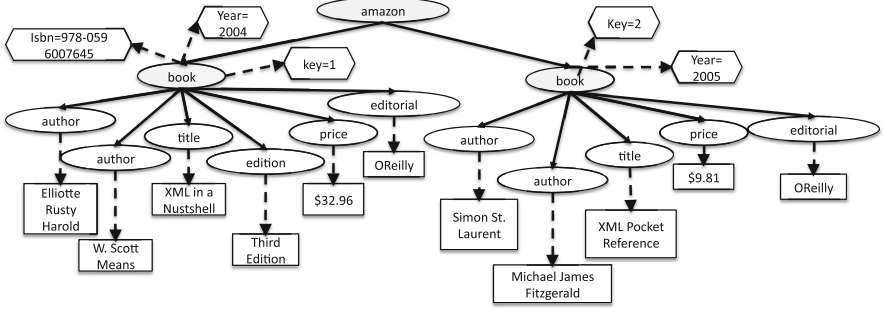
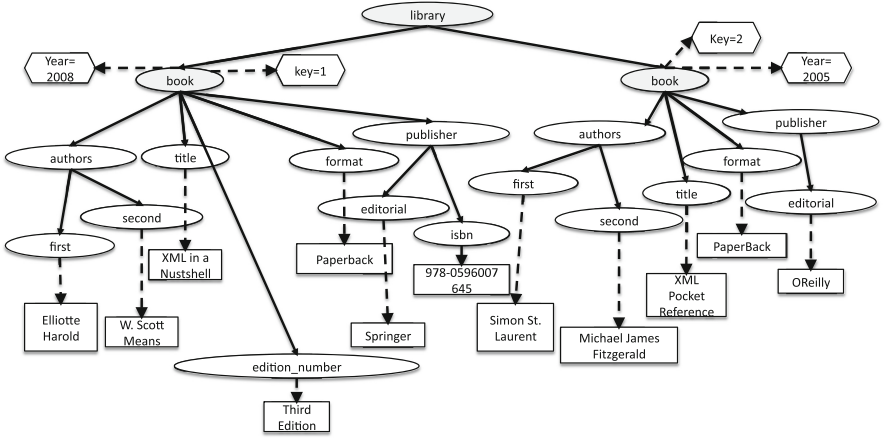
1.1 On the Innovativeness of the SemSynX Proposal

By analyzing active literature, we recognize that several studies on how to match string values exist, but there is a general lack of methods supporting the definition of *similarity functions* that work on “specific cases”. Since **SemSynX** aims at offering a high degree of parametrization, our strategy is assigning a proper similarity function to each label. As an alternative to this, a *naïve* strategy would assign similarity functions to types, indeed, but semantics of labels of the same type can vary one from another. As an example, labels of type **Integer** can represent the age of a person, the number of children as well as the population of a city, alternatively. Therefore, we would “lost” the specific semantics of the XML element. In order to avoid this, we would aim at semantically enriching meta-data of target documents (i.e., *DTD* or *XML Schemas*) with similarity functions. Unfortunately, neither DTD nor XML Schemas provide mechanisms for expressing semantics of element. As a consequence, our strategy embedded in **SemSynX** has been that of attaching a suitable *XML template* to each XML document, being this template capable of providing semantics to labels, and enabling a more fine-grained comparison of labels according to their semantics.

As far as we know, **SemSynX** is the first technique that fully-provides semantics-based similarity analysis of XML documents. All this confers highly *flexibility* to our proposed similarity analysis framework, with powerful benefits in the context of large-scale management of XML data over Clouds (e.g., [19]).

2 Running Example

Figures 1 and 2 show two example XML trees, namely $T_{X,1}$ and $T_{X,2}$, respectively, where two objects modeling books, i.e. “*XML in a Nutshell*” and “*XML Pocket Reference*”, are represented in *both* XML trees. The first tree, $T_{X,1}$, includes the XML elements **year**, **isbn**, **author**, **title**, **edition**, **price** and

Fig. 1. Example XML document $T_{X,1}$ Fig. 2. Example XML document $T_{X,2}$

editorial, while the second tree, $T_{X,2}$, also includes, beyond the same previous $T_{X,1}$'s elements except **price**, the element **format**. However, looking in more details, the representation of the *same* data objects (i.e., the two books) are different in the target XML trees. In fact, $T_{X,1}$ models **isbn** as an attribute, while $T_{X,2}$ makes use of an element to this end. Also, $T_{X,1}$ makes use of the element **author** to model authors, while elements **first**, **second**, and so forth, are used in $T_{X,2}$ as enclosed under the element **authors** to this end. Additionally, **editorial** is used in $T_{X,2}$ as sub-element of **publisher**, which also includes the element **isbn**, while **editorial** is under **book** in $T_{X,1}$. As regards conflicts, **edition** is used in $T_{X,1}$ and **edition_number** in $T_{X,2}$ to represent the same concept. Finally, **year** and **editorial** for the data object “*XML in a Nutshell*” have different values for both XML trees $T_{X,1}$ and $T_{X,2}$.

As highlighted in Sect. 1, the goal of SemSynX is to automatically discover similarity/dissimilarity matches in the target XML trees, by setting as input parameters the following two parameters: (i) paths of objects to be

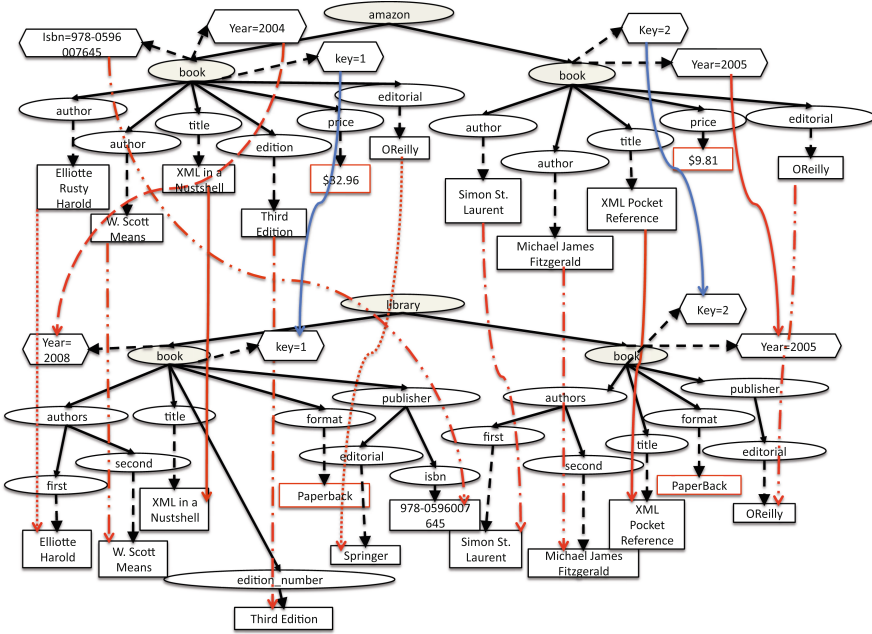


Fig. 3. Similarities/Dissimilarities detected by **SemSynX** for the XML trees $T_{X,1}$ and $T_{X,2}$ of Figs. 1 and 2, respectively

analyzed, and (ii) names of keys of such objects¹ Consider again the two example XML trees shown in Figs. 1 and 2, respectively. Here, objects paths are set as $P_{X,1}:/\text{amazon}/\text{book}$ for $T_{X,1}$ and $P_{X,2}:/\text{library}/\text{book}$ for $T_{X,2}$; key names are set as the attribute **key** which is associated to both books of $T_{X,1}$ and $T_{X,2}$. The similarities/dissimilarities of $T_{X,1}$ and $T_{X,2}$ of Figs. 1 and 2, respectively, detected by **SemSynX** are finally shown in Fig. 3. Here, full arrows model similarities, while dashed arrows model dissimilarities.

The **SemSynX** running example shown in Fig. 3, and focusing on *both* similarities and dissimilarities of XML trees $T_{X,1}$ and $T_{X,2}$ of Figs. 1 and 2, respectively, provides the following similarity/dissimilarity analysis. First, exact matches exist, for both paths and values. This is the case of **title** of both $T_{X,1}$ and $T_{X,2}$ and **year** in the $T_{X,2}$. Special cases of exact matching are those exchanging attributes by tags. This is the case of **isbn** in $T_{X,1}$. Weaker matching are those in which the value is the same while the path is different. This is the case of the second element **author** (i.e., “*W. Scott Means*”) and **edition** in $T_{X,1}$, and both **author** and **editorial** in $T_{X,2}$. Paths are different in two cases: tag changing (in **edition** of $T_{X,1}$), tag removing/adding (in the second element **author** of $T_{X,1}$, i.e. “*W. Scott Means*”, and both **author** and **editorial** of $T_{X,2}$). Other weaker

¹ The use of keys is not mandatory in our approach, but keys are used in the running example to guide similarity search.

matching are those in which the path is the same but value is different. This is the case of **year** in $T_{X,1}$. The weakest matching are those in which paths are similar and values are not equal or are similar. This is the case of the first **author** (i.e., “*Elliotte Rusty Harold*”) and **editorial** in $T_{X,1}$. Finally, two elements can be considered as “new” in $T_{X,1}$ and $T_{X,2}$: **price** and **format**, respectively.

The motivating example has clearly shown the potentialities and the flexibility of our proposed technique **SemSynX** in dealing with XML data similarity (and dissimilarity) detection that is perfectly suitable with the requirements posed by modern Cloud Computing environments (e.g., [19]), since the same detection task can also be easily deployed on top of Cloud infrastructures.

3 The SemSynX Approach

In this Section, we provide the details and the proof-of-concept of **SemSynX**, our proposed XML data similarity analysis technique.

SemSynX is based on the analysis of similarity of both paths and values, where similarity is measured by means of a score. From Sect. 1, recall that both a local and a global score is provided. In the case of paths, the local score represents the number of equal matches as well as the number of changes made, i.e. the number of tags that have been removed, added and changed. It is finally obtained as the percentage of tags that occur in the same order in both compared paths. In the case of values, the local score is provided by a similarity function, which is defined by the user (e.g., the Levenshtein’s distance [18]). **SemSynX** offers a wide repertoire of string comparison functions and semantics-based functions (see Sect. 4), but others can be still defined by user for supporting every particular matching case that is of interest for the target similarity analysis.

SemSynX enables the specification of the following thresholds and weights:

1. **Equality Threshold:** Cut value at which the content of labels is considered equal. Content of labels are compared by means of a user-defined similarity function that reports a value between 0 and 1, and, depending on the type of content, users can establish a suitable equality threshold. In the case of non-exact matching, relaxing equality threshold is required.
2. **Similarity Threshold:** Cut value at which the content of labels is considered similar. Users are allowed to establish a similarity threshold for content of labels. Relaxing similarity threshold for matching of close-values is ensured.
3. **Path Threshold:** Cut value at which paths are considered equal. The user can specify a threshold based on which paths are considered equal or different. High thresholds ensure similarity in structure of XML elements, while low thresholds permit XML elements of varying in structure. Paths are thus classified into similar paths or *new paths*.
4. **Path Weight:** Degree of relevance of paths. Higher relevance of paths rather than values’ relevance requires similar structure.
5. **Value Weight:** Degree of relevance of values. Higher relevance of values rather than paths’ relevance requires similar content.

6. **Main Path Weight:** Degree of relevance of *main paths*. **SemSynX** makes it possible to select paths of each document to be compared. This functionality turns to be useful in order to remove non-relevant paths as well as paths that user know in advance as being different. Selected paths are called “main paths”, while *secondary paths* are similar paths not selected by the user. The user can assign more relevance to main paths, but leaving some weight to secondary paths as well as new paths.
7. **Secondary Path Weight:** Degree of relevance of secondary paths. Low weight to secondary paths involves in assigning more relevance to main paths, while high weight to secondary paths makes it possible to debug the matching. Whenever increasing weight of secondary paths, the global score is considerably higher, and the selection of main paths is not successful.
8. **New Path Weight:** Degree of relevance of new paths. By assigning more relevance to new paths, distinct elements can be detected.

These model concepts characterize **SemSynX** as a flexible XML similarity analysis technique. Indeed, the main advantage of **SemSynX** over state-of-the-art approaches is represented by the ability of customizing path and value similarity via the so-called “degree of similarity” (see Sect. 1) as well as specifying thresholds and weights for path and value similarity as to support the desired degree of similarity. Also, the flexibility provided by **SemSynX** is higher than the one of other comparative approaches because it works with user-defined similarity functions ranging from string-based comparison to semantics-based comparison (see Sect. 4) for each type of XML element, and it enables the selection of main, secondary and new paths, as well assigning weights to such paths in a proportional manner.

3.1 XML Data Similarity Analysis in SemSynX

By combining the above concepts, **SemSynX** distinguishes the following cases of XML data similarity analysis, which represent the core of the proposed technique. The cases referred in the following description are still related to the running example provided in Sect. 2.

1. Equal Value and Equal Tag/Attribute. This is the best matching. Both value and tag/attribute match. This is the case, for instance, of `title` in both books “*XML in a Nutshell*” and “*XML Pocket Reference*” (of both XML trees $T_{X,1}$ and $T_{X,2}$), and `editorial` in the second book “*XML Pocket Reference*” (of both XML trees $T_{X,1}$ and $T_{X,2}$). Nevertheless, the semantic of equality can be customized by a threshold over which values are considered equal. A small change in any of the element `title` can be accepted as exact match, depending on the threshold. Equal tag/attribute does not mean equality of paths. For instance, depending on the path threshold, `editorial` can be rejected since $P_{X,3} : /amazon/book/editorial$ and $P_{X,4} : /amazon/book/publisher/editorial$ are corresponding paths in both documents. In particular, when the threshold is over 0.75, similarity of

`editorial` is no longer detected. Change of an attribute for a tag is also allowed. This is the case of `isbn` in the first book “*XML in a Nutshell*” (of both XML trees $T_{X,1}$ and $T_{X,2}$).

2. Similar Value and Equal Tag/Attribute. Here, tags are equal but values are similar. This is the case of the first author name of the first book “*XML in a Nutshell*” (of both XML trees $T_{X,1}$ and $T_{X,2}$). Obviously, detecting similarity versus equality for values depends on the specified threshold. The value of `year` in the first book “*XML in a Nutshell*” of $T_{X,1}$ is 2004, while the value of `year` in the first book “*XML in a Nutshell*” of $T_{X,2}$ is 2008. Thanks to user-defined similarity functions, in our approach `Integer` values can be specifically handled. For instance, the user can define a similarity function that consider 2004 and 2008 as similar or distinct. Moreover, equality and similarity thresholds play a relevant role in this case. Change of an attribute for a tag is also allowed.

3. Distinct Value and Equal Tag/Attribute. This is one case of conflict, also exposing semantic dissimilarity. The tag is equal but the content is distinct. The value has to be neither equal nor similar according to the defined threshold. This is the case of `editorial` in the first book “*XML in a Nutshell*” (of both XML trees $T_{X,1}$ and $T_{X,2}$). Again, equality of tags does not mean equality of paths. Change of an attribute for a tag is also allowed.

4. Equal Value and Distinct Tag/Attribute. This is another case of conflict, also exposing syntactic dissimilarity. The value is the same but the tag is distinct. Again, detecting this case depends on the defined threshold. This is the case of `edition` and `edition_number` of the first book “*XML in a Nutshell*” (of both XML trees $T_{X,1}$ and $T_{X,2}$). Tags are compared as strings, and thus `edition` is `edition_number` are distinct. Change of an attribute for a tag is also allowed.

5. Similar Value and Distinct Tag/Attribute. This is, again, a syntactic disagreement. This case is not reported in the running example. Change of an attribute for a tag is also allowed.

6. Distinct Value and Tag/Attribute. This is the worst case which also means that a new element has been found in the compared XML trees. This is the case of `price` (in the second book “*XML Pocket Reference*” of $T_{X,1}$) and `format` (in the second book “*XML Pocket Reference*” of $T_{X,2}$). The value has to be neither equal nor similar according to the required threshold.

As result of the XML similarity analysis provided by SemSynX, a list of agreements/disagreements is retrieved, being the list itself represented as a XML document. This XML document makes use of tags for describing the kind of similarity (dissimilarity, respectively) for each element detected by the target analysis, as well as the path to such element and its content, by also reporting the similarity (local) score. A global score is also reported on the top of the document, and counts of agreements and disagreements are shown. In particular, the global score is computed in a *proportional manner* by taking into account weights assigned to main, secondary and new paths.

Figure 4 shows the final similarity analysis provided by SemSynX for the first book “*XML Pocket Reference*” of the running examples, having fixed: 0.5 as value


```

<result global_score="0.5336309523809524"
dissimilar="2" similar="8">
  <new>
    <path>library/book</path>
    <subpath>book/format</subpath></new>
  <new>
    <path>amazon/book</path>
    <subpath>book/price</subpath></new>
  <EqValueandDistinctLabel score="0.52380952380952381">
    <item>
      <content>Elliotte Harold</content>
      <path>book/authors/first</path></item>
    <item>
      <content>Elliotte Rusty Harold</content>
      <path>book/author</path></item>
  </EqValueandDistinctLabel>
  <EqValueandDistinctLabel score="0.66666666666666666666">
    <item>
      <content>W. Scott Means</content>
      <path>book/authors/second</path></item>
    <item>
      <content>W. Scott Means</content>
      <path>book/author</path></item>
  </EqValueandDistinctLabel>
  <EqValueandDistinctLabel score="0.75">
    <item>
      <content>Third Edition</content>
      <path>book/edition_number</path></item>
    <item>
      <content>Third Edition</content>
      <path>book/edition</path></item>
  </EqValueandDistinctLabel>
  <EqLabelandContent score="1">
    <item>
      <path>book/title</path>
      <content>XML in a Nutshell</content></item>
    </item>
    <path>book/title</path>
    <content>XML in a Nutshell</content></item>
  </EqLabelandContent>
  <EqLabelandDistinctValue score="0.39583333333333333334">
    <item>
      <path>book/publisher/editorial</path>
      <content>Springer</content></item>
    <item>
      <path>book/editorial</path>
      <content>OReilly</content></item>
  </EqLabelandDistinctValue>
  <EqAttributeandDistinctValue score="0.5">
    <item>
      <path>book</path>
      <attribute>year</attribute>
      <content>2008</content></item>
    <item>
      <path>book</path>
      <attribute>year</attribute>
      <content>2004</content></item>
  </EqAttributeandDistinctValue>
  <EqAttributeandContent score="1">
    <item>
      <path>book</path>
      <attribute>key</attribute>
      <content>i</content></item>
    <item>
      <path>book</path>
      <attribute>key</attribute>
      <content>i</content></item>
  </EqAttributeandContent>
  <EqAttributeLabelandContent score="0.5">
    <item>
      <path>book</path>
      <attribute>isbn</attribute></item>
      <path>book/publisher/isbn</path>
      <content>978-0596007645</content>
    </EqAttributeLabelandContent>
  </result>

```

Fig. 4. Similarity analysis provided by SemSynX for the first book “*XML pocket reference*” of the running example

Table 1. Retrieved Scores and Thresholds

ThEq	ThSim	ThP	RV	RP	Score	Diss	Sim
1	1	1	1	1	0,178	11	3
1	1	0,75	1	1	0,178	11	3
1	1	0,5	1	1	0,303	7	5
1	1	0,25	1	1	0,392	5	6
1	1	0	1	1	0,437	4	11
0,75	0,5	0	1	1	0,437	4	7
0,5	0,25	0	1	1	0,533	2	8

(a) Running Example

Documents	ThEq	ThSim	ThP	RV	RP	Score	Diss	Sim
Ex1-Ex2	1	1	1	1	1	0,090	10	1
Ex1-Ex2	1	1	0,5	1	1	0,583	2	5
Ex1-Ex2	1	1	0,25	1	1	0,784	0	6
Ex1-Ex2	1	1	0,25	5	1	0,928	0	6
Ex1-Ex3	1	1	1	1	1	0,703	0	6
Ex1-Ex3	0,5	0,25	1	1	1	0,703	0	6
Ex1-Ex3	0,5	0,25	1	1	5	0,910	0	6

(b) Documents of Figure 5

equality threshold, 0.25 as value similarity threshold, and 0.0 as path similarity threshold, respectively. All the paths have considered relevant here, and the Levenshtein’s distance function [18] has been considered for all the elements.

4 Experimental Results

In this Section, we propose the experimental assessment and analysis of SemSynX over several classes of XML documents. The final goal of this assessment is to show the evident improvements obtained by customizing the similarity analysis by means of several thresholds, relevance measures, selection of paths and user-defined similarity functions. It should be noted that this special feature

<pre> <example> <member id="1"> <familyname>Smith</familyname> <firstname>John</firstname> <age>67</age> <address>Fifth avenue</address> <city>NY</city> </member> </example> </pre>	<pre> <example> <member id="1"> <personal_info> <name> <familyname>Smith</familyname> <firstname>John</firstname> </name> <age>67</age> <postal_address> <street>Fifth avenue</street> <city>NY</city> </postal_address> </personal_info> </member> </example> </pre>	<pre> <example> <member id="1"> <familyname>Smith </familyname> <firstname>J.</firstname> <age>67 years</age> <address> 767 Fifth Avenue, 3rd Floor</address> <city>New York, NY 10153</city> </member> </example> </pre>
Example 1	Example 2	Example 3

Fig. 5. Synthetic XML documents

represents one of the main advantages of SemSynX over state-of-the-art approaches (see Sect. 3).

As synthetic XML documents, we considered instances with different characteristics (see Sect. 3), in order to stress the flexible properties of SemSynX. Hence, we considered both XML documents of the running example (see Figs. 1 and 2) and XML documents shown in Fig. 5. We then set different values of thresholds, according to our main XML similarity analysis asset (see Sect. 3). Table 1 shows the retrieved results. In more details, in the case of the running example, where we compared the XML document $T_{X,1}$ (see Fig. 1) and the XML document $T_{X,2}$ (see Fig. 2), Table 1(a) reports the retrieved results. Here, when thresholds are strong (i.e., 1.0), the similarity global score is 0.178 and the rate of similar/dissimilar elements is 3/11, which appears to be too low. By decreasing the path similarity threshold from 1.0 to 0.0, a score of 0.437 and a rate of 11/4 are obtained. Let us remark that decreasing the path similarity threshold from 0.25 to 0.0 causes that paths with similarity under 25 % are also compared. In order to improve similarity scores, equality and similarity thresholds for values must be decreased from 1.0 to 0.75 and 0.5 and from 1.0 to 0.5 and 0.25, respectively, thus obtaining rates of similar/dissimilar elements equal to 7/4 and 8/2, respectively. The last case provides a score equal to 0.533 that represents more than 50 % of similarity and 2 distinct elements over 8 similar elements in total, which is the expected result from the user point of view.

In the case of the example XML documents shown in Fig. 5, we compared have been taken in order to compare the document *Example 1*, named as $D_{X,1}$, against two documents *Example 2*, named as $D_{X,2}$, and *Example 3*, named as $D_{X,3}$, respectively. In the first case, i.e. the comparison of $D_{X,1}$ against $D_{X,2}$, it follows that $D_{X,2}$ is similar to $D_{X,1}$ in content but dissimilar in structure. Thus, we hope to find best scores when path comparisons are relaxed (see Sect. 3). As shown in Table 1(b), by decreasing path similarity threshold to 0.25 and conferring 5 times more relevance to value similarity, even when value equality and similarity remain fixed, the score is improved up to more than 92 % of similarity. In the second case, i.e. the comparison of $D_{X,1}$ against $D_{X,3}$, it follows that $D_{X,3}$ is similar to $D_{X,1}$ in structure but dissimilar in content. As shown in Table 1(b),

by decreasing equality and similarity thresholds to 0.5 and 0.25, respectively, and conferring 5 times more relevance to paths, the score is improved up to more than 91 % of similarity. Let us remark that, in all the cases, the rate of similarity/dissimilarity is equal to 6/0.

In summary, our experimental assessment and analysis has shown that similarity analysis of XML documents greatly depends on the thresholds and relevance required for path and value similarity, and, depending on the structure/content of the documents (i.e., depending on the nature of the documents themselves), thresholds, relevance, selected paths and user-defined similarity functions play a key role. This completely proves the powerful flexibility of the proposed XML similarity analysis technique **SemSynX**.

5 Conclusions and Future Work

In this paper, we have introduced and experimentally assessed **SemSynX**, a XML similarity analysis technique whose main benefits can be summarized by three main concepts: customization, flexibility and interoperability. The proposed technique retrieves the list of agreement/disagreement matches of two input XML documents, also determining suitable local and global similarity scores. **SemSynX** is fully customizable, thus enabling the specification of thresholds for path and value equality/similarity. Also, it makes possible to confer more relevance to paths versus values, and vice-versa, according to a proportional approach, as well as selection of paths and similarity functions. Finally, we have provided an experimental assessment and analysis of **SemSynX** against several classes of XML documents. The tool has been implemented as an XQuery library.

As future work we plan to move the attention on the following research lines. Firstly, incorporating within **SemSynX** a more sophisticated notion of path similarity. On one hand, path order as well as path duplicates are not still considered in path similarity analysis. On the other hand, asymmetric comparison (i.e., electing a master XML document) should be addressed in future efforts. Secondly, the thresholding technique will be extended. A possible improvement can be introduced by considering *distinct thresholds* for each type of tag. Thirdly, a very interesting future extension is represented by considering RDF data as target data format. This opens an interesting research line where trees are replaced by graphs. Providing similarity scores for RDF graphs enables comparison and fusion of *Semantic Web Data*. Lastly, considering *adaptiveness features* (e.g., [6,7]), *knowledge representation techniques* (e.g., [10]), and *uncertainty in data* (e.g., [17]) are open research problems to be considered in future, and suitable characteristics of **SemSynX** should deal with those accordingly.

Acknowledgments. This work was funded by the EU ERDF and the Spanish Ministry of Economy and Competitiveness (MINECO) under Project TIN2013-44742-C4-4-R as well as by the Andalusian Regional Government under Project P10-TIC-6114.

References

1. Aïtelhadj, A., Boughanem, M., Mezghiche, M., Souam, F.: Using structural similarity for clustering XML documents. *Knowl. Inf. Syst.* **32**(1), 109–139 (2012)
2. Algergawy, A., Mesiti, M., Nayak, R., Saake, G.: XML data clustering: an overview. *ACM Comput. Surv. (CSUR)* **43**(4), 25 (2011)
3. Almendros-Jiménez, J.M., Cuzzocrea, A.: Towards flexible similarity analysis of XML data. In: *On the Move to Meaningful Internet Systems: OTM 2015 Workshops*, Rhodes, Greece, 26–30 October 2015, pp. 573–576 (2015)
4. Bleiholder, J., Naumann, F.: Data fusion. *ACM Comput. Surv. (CSUR)* **41**(1), 1 (2008)
5. Bryl, V., Bizer, C., Isele, R., Verlic, M., Hong, S.G., Jang, S., Yi, M.Y., Choi, K.-S.: Interlinking and knowledge fusion. In: Auer, S., Bryl, V., Tramp, S. (eds.) *Linked Open Data. LNCS*, vol. 8661, pp. 70–89. Springer, Heidelberg (2014)
6. Cannataro, M., Cuzzocrea, A., Mastroianni, C., Ortale, R., Pugliese, A.: Modeling adaptive hypermedia with an object-oriented approach and XML. In: *Proceedings of the Second International Workshop on Web Dynamics*, pp. 35–44 (2002)
7. Cannataro, M., Cuzzocrea, A., Pugliese, A., Bucci, V.P.: A probabilistic approach to model adaptive hypermedia systems. In: *Proceedings of the First International Workshop for Web Dynamics*, pp. 12–30 (2001)
8. Cecchin, F., de Aguiar Ciferri, C.D., Hara, C.S.: XML data fusion. In: Bach Pedersen, T., Mohania, M.K., Tjoa, A.M. (eds.) *DAWAK 2010. LNCS*, vol. 6263, pp. 297–308. Springer, Heidelberg (2010)
9. Costa, G., Cuzzocrea, A., Manco, G., Ortale, R.: Data de-duplication: a review. In: Biba, M., Xhafa, F. (eds.) *Learning Structure and Schemas from Documents. SCI*, vol. 375, pp. 385–412. Springer, Heidelberg (2011)
10. Cuzzocrea, A.: Combining multidimensional user models and knowledge representation and management techniques for making web services knowledge-aware. *Web Intell. Agent Syst.* **4**(3), 289–312 (2006)
11. Cuzzocrea, A., Puglisi, P.L.: Record linkage in data warehousing: state-of-the-art analysis and research perspectives. In: *Database and Expert Systems Applications, DEXA 2011, International Workshops*, Toulouse, France, August 29 – September 2 2011, pp. 121–125 (2011)
12. Do Nascimento, A.M., Hara, C.S.: A model for XML instance level integration. In: *Proceedings of the 23rd Brazilian Symposium on Databases*, pp. 46–60. Sociedade Brasileira de Computação (2008)
13. Dong, X.L., Naumann, F.: Data fusion: resolving data conflicts for integration. *Proc. VLDB Endowment* **2**(2), 1654–1655 (2009)
14. Dorneles, C.F., Gonçalves, R., dos Santos Mello, R.: Approximate data instance matching: a survey. *Knowl. Inf. Syst.* **27**(1), 1–21 (2011)
15. Geerts, F., Mecca, G., Papotti, P., Santoro, D.: Mapping and cleaning. In: *IEEE 30th International Conference on Data Engineering (ICDE 2014)*, pp. 232–243. IEEE (2014)
16. Hara, C.S., de Aguiar Ciferri, C.D., Ciferri, R.R.: Incremental data fusion based on provenance information. In: Tannen, V., Wong, L., Libkin, L., Fan, W., Tan, W.-C., Fourman, M. (eds.) *Buneman Festschrift 2013. LNCS*, vol. 8000, pp. 339–365. Springer, Heidelberg (2013)
17. Leung, C.K.-S., Cuzzocrea, A., Jiang, F.: Discovering frequent patterns from uncertain data streams with time-fading and landmark models. *T. Large-Scale Data-Knowl. Centered Syst.* **8**, 174–196 (2013)

18. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions and reversals. In: Soviet physics doklady, vol. 10, p. 707 (1966)
19. Lung, C.-H., Sanaullah, M., Cao, Y., Majumdar, S.: Design and performance evaluation of cloud-based XML publish/subscribe services. In: IEEE International Conference on Services Computing, SCC 2014, Anchorage, AK, USA, June 27 – July 2 2014, pp. 583–589 (2014)
20. Mendes, P.N., Mühleisen, H., Bizer, C.: Sieve: linked data quality assessment and fusion. In: Proceedings of the Joint EDBT/ICDT 2012 Workshops, pp. 116–123. ACM (2012)
21. Milano, D., Scannapieco, M., Catarci, T.: Using ontologies for XML data cleaning. In: Meersman, R., Tari, Z. (eds.) OTM-WS 2005. LNCS, vol. 3762, pp. 562–571. Springer, Heidelberg (2005)
22. Oliveira, P., de Fatima Rodrigues, M., Henriques, P.R.: An ontology-based approach for data cleaning. In: ICIQ, pp. 307–320 (2006)
23. Rahm, E., Bernstein, P.A.: A survey of approaches to automatic schema matching. VLDB J. **10**(4), 334–350 (2001)
24. Sundaram, S., Kumar, S.: Madria.: a change detection system for unordered XML data using a relational model. Data Knowl. Eng. **72**, 257–284 (2012)
25. Tekli, J., Chbeir, R., Yetongnon, K.: An overview on XML similarity: background, current trends and future directions. Comput. Sci. Rev. **3**(3), 151–173 (2009)
26. Weis, M., Manolescu, I.: Declarative XML data cleaning with XClean. In: Krogstie, J., Opdahl, A.L., Sindre, G. (eds.) CAiSE 2007 and WES 2007. LNCS, vol. 4495, pp. 96–110. Springer, Heidelberg (2007)
27. Weis, M., Naumann, F.: Detecting duplicates in complex XML data. In: Proceedings of the 22nd International Conference on Data Engineering, ICDE 2006, pp. 109–109. IEEE (2006)
28. Winkler, W.E.: The state of record linkage and current research problems. In: Statistical Research Division, US Census Bureau (1999)
29. Yaguinuma, C.A., Afonso, G.F., Ferraz, V., Borges, S., Santos, M.T.: A fuzzy ontology-based semantic data integration system. J. Inf. Knowl. Manage. **10**(03), 285–299 (2011)
30. Zhang, D., Song, T., He, J., Shi, X., Dong, Y.: A similarity-oriented RDF graph matching algorithm for ranking linked data. In: 2012 IEEE 12th International Conference on Computer and Information Technology (CIT), pp. 427–434. IEEE (2012)

Hybrid Artificial Intelligent Systems

11th International Conference, HAIS 2016, Seville,
Spain, April 18-20, 2016, Proceedings

Martínez-Álvarez, F.; Troncoso, A.; Quintián, H.;
Corchado, E. (Eds.)

2016, XIX, 765 p. 211 illus., Softcover

ISBN: 978-3-319-32033-5