On Computing the Importance of Associations in Large Conceptual Schemas

Antonio Villegas, Antoni Olivé, and Maria-Ribera Sancho

Department of Service and Information System Engineering
Universitat Politècnica de Catalunya
Barcelona, Spain
{avillegas,olive,ribera}@essi.upc.edu

Abstract. The visualization and the understanding of large conceptual schemas require the use of specific methods. These methods generate clustered, summarized or focused schemas that are easier to visualize and to understand. All of these methods require computing the importance of the elements in the schema but, up to now, only the importance of entity types has been taken into account. In this paper, we present three methods for computing the importance of associations by taking into account the knowledge defined in the structural and behavioral parts of the schema. We experimentally evaluate these methods with large real-world schemas and present the main conclusions we have drawn from the experiments.

Keywords: Conceptual Modeling, Importance, Associations.

1 Introduction

A conceptual schema defines the general knowledge about the domain that an information system of an organization needs to know to perform its functions [16,20,18]. The conceptual schema of many real-world information systems are too large to be easily managed or understood. One of the most challenging and long-standing goals in conceptual modeling is to ease the comprehension of large conceptual schemas [11,15]. The visualization and understanding of these schemas requires the use of specific methods, which are not needed in small schemas. These methods generate indexed [24,5,23,22], clustered [10,12,19], summarized [8,28,27] or focused [26,25] schemas that are easier to visualize and to understand.

Many of the above methods require computing the importance (also called relevance or score) of each element in the schema. The computed importance induces an ordering of the elements, which plays a key role in the steps and result (output) of the method. Up to now, the existing metrics of importance for schema elements were mainly centered in computing the importance of entity types, but not in the importance of associations.

The main objective of this paper is to analyze existing metrics for measuring the importance of entity types, and then to adapt them to be able to work

A. Düsterhöft et al. (Eds.): Thalheim Festschrift, LNCS 7260, pp. 216–230, 2012.

with associations. We present two different methods to compute the importance of associations inspired by the entity-type importance methods of occurrence counting and link analysis [24]. In addition, our contribution also includes a method to obtain the importance of associations by adapting a betweenness-centrality measure [2,3] from the fields of graph theory and complex networks [1].

Our approach takes into account the knowledge defined in the schema about associations, including their participant entity types, the cardinality constraints of those participations, the general constraints of the schema, and the specification of behavioral events. All of them contribute to measure the importance of associations.

We have experimentally evaluated each method using the conceptual schemas of the osCommerce [21] and EU-Rent [9], the UML2 metaschema [13], a fragment of the HL7 schemas [26], and the OpenCyc ontology [6]. All of them contain a large amount of entity types and associations, which make difficult their understanding. We analyze the differences between methods and schemas, and make conclusions on the importance of associations and its value in the comprehension of large conceptual schemas.

The rest of the paper is organized as follows. Section 2 introduces the basic concepts and notations. Section 3 reviews the concept of reification of associations. Section 4 presents the methods to compute the importance of associations in detail. Section 5 describes the experimentation with the methods, the results obtained, and the conclusions we have drawn. Finally, Section 6 summarizes the paper and points out future work.

2 Basic Concepts and Notations

In this section we review the main concepts and the notations we have used to define the knowledge of conceptual schemas. In this paper, we deal with schemas written in the UML/OCL[13,14], which consists of the elements summarized in Def. 1.

Definition 1. *(Conceptual Schema) A conceptual schema $\mathcal{CS}$ is defined as a triple $\mathcal{CS} = \langle \mathcal{E}, \mathcal{R}, \mathcal{C} \rangle$, where:*

- *$\mathcal{E}$ is a set of* entity types. *Some $e \in \mathcal{E}$ represents event types [17].*
- *$\mathcal{R}$ is a set of* associations *between entity types of $\mathcal{E}$. The degree of an association is the number of entity types that participate on it. An association r has a degree $n \geq 2$.*
- *$\mathcal{C}$ is a set of* schema rules. *$\mathcal{C}$ contains textual OCL constraints and the transformation of all graphical UML constraints, including cardinality constraints, into OCL expressions.*

Table 1 summarizes the basic metrics used in the rest of the paper. If $r \in \mathcal{R}$ then $members(r)$ denotes the set of entity types that participate in the association r, and $assoc(e)$ the set of associations in which e participates. Note that an

entity type e may participate more than once in the same association, and therefore $members(r)$ and $assoc(e)$ are multisets (may contain duplicate elements). Moreover, $conn(e)$ denotes the multiset of entity types connected to e through associations.

Table 1. Definition of basic metrics

Notation	Definition
$members(r)$	$= \{e \in \mathcal{E} \mid e \text{ is a participant of } r\}$
$assoc(e)$	$= \{r \in \mathcal{R} \mid e \in members(r)\}$
$conn(e)$	$= \uplus_{r \in assoc(e)} \{members(r) \backslash \{e\}\}$[1]

We denote by $\mathcal{C}$ the set of schema rules of a conceptual schema, including constraints, derivation rules and pre- and postconditions. Each schema rule c is defined in the context of an entity type, denoted by $context(c)$. In OCL, each rule c consists of a set of OCL expressions (see OCL [14]) that may refer to several entity types which are denoted by $ref(c)$. We also include in $\mathcal{C}$ the schema rules corresponding to the equivalent OCL invariants of the cardinality constraints.

Table 2. Definition of extended metrics

Notation	Definition
$ref(c)$	$= \{e \in \mathcal{E} \mid e \text{ is referenced in } c \in \mathcal{C}\}$
$links_{context}(c)$	$= \{\{e, e'\} \mid e, e' \in \mathcal{E} \wedge e = context(c) \wedge e' \in ref(c)\}$
$links_{nav}(c)$	$= \{\{e, e'\} \mid e, e' \in \mathcal{E} \wedge e \rightarrow e' \text{ is a navigation in } c\}$
$links(c)$	$= links_{context}(c) \cup links_{nav}(c)$
$rconn(e)$	$= \uplus_{c \in \mathcal{C}} \{e' \in \mathcal{E} \mid \{e, e'\} \subset links(c)\}$

A special kind of OCL expression is the navigation expression that define a schema navigation from an entity type to another through an association (see `NavigationCallExp` of OCL in [14]). Such expressions only contain two entity types as its participants, i.e. the source entity type and the target one ($e \rightarrow e'$). We denote by $links_{nav}(c)$ the set of pairs $\langle e, e' \rangle$ that participate in the navigation expressions of c. We also denote by $links_{context}(c)$ the sets of pairs of entity types composed by the context of the rule c and every one of the participant entity types of such rule ($e \in ref(c)$). Finally, we define $links(c)$ as the union of $links_{context}(c)$ with $links_{nav}(c)$ and, $rconn(e)$ as the multiset of entity types that compose a pair with e in $links(c)$. Note that since we use $\uplus$, $rconn(e)$ may contain duplicates because it takes into account each rule c, and an entity type e can be related to another one e' in two or more different rules.

[1] Note that "\" denotes the difference operation of multisets as in $\{a, a, b\} \backslash \{a\} = \{a, b\}$ and "$\uplus$" denotes the multiset (or bag) union that produces a multiset as in $\{a, b\} \uplus \{a\} = \{a, a, b\}$.

Intuitively, $rconn(e)$ is the multiset of entity types to which an entity type e is connected through schema rules. Table 2 shows the formal definition of the extended metrics for schema rules.

3 Reifications

Reifying an association consists in viewing it as an entity type [16]. When we view an association r as an entity e_r, we say that the entity reifies the association. The reification of an association does not add any new knowledge to a schema, but it is an interesting schema transformation that can be used in our context of importance of associations. Since the existing methods for importance computing are defined for entity types, the reification of association into entity types is a way to easily adapt the schema and such methods in order to compute the importance of associations.

Formally, the reification of an association $r \in \mathcal{R}$ is an entity type e_r connected with the participant entity types of r through *intrinsic* binary associations. For the case of association classes, since an association class is also an entity type, we make an implicit reification only changing the connections with the participants by adding the intrinsic binary associations.

In the case of a binary association r (left side of Fig. 1), the cardinality constraints after the transformation are placed in the sides of the entity type e_r that reifies the association. The cardinality constraints of a participant (e.g. α of A) go to the new intrinsic association between the other participant (B) and the new entity (C), placed on the side of the new entity type. On the side of the previous participants (A and B) the cardinality equals 1. Furthermore, to maintain all the semantics we also need a new *uniqueness* constraint expressed in OCL, as shown in Fig. 1.

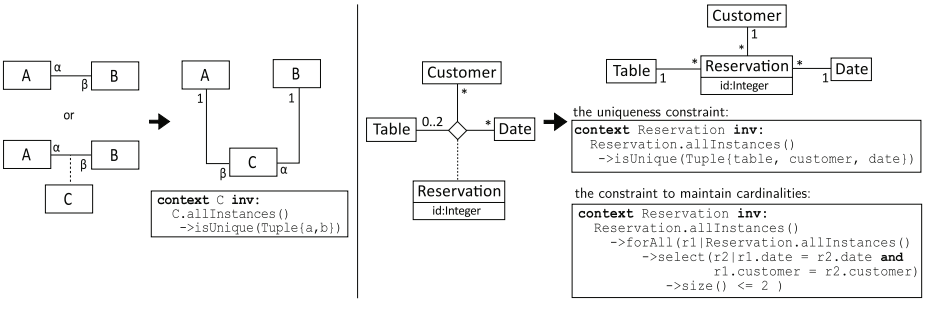

Fig. 1. Reification of binary (left side) and ternary (right side) associations

For the case of an n-ary association the transformation is similar (see right side of Fig. 1). However, the multiplicities in the intrinsic binary associations between the participants and the entity type that reifies the n-ary association are always "1" on the participant side and "*" on the entity-type side. In this case a uniqueness constraint is also needed and follows the same idea as with binary

associations. Furthermore, the cardinality constraints of the initial association must be expressed as an OCL constraint to maintain all the semantics in the reified version. For example, the multiplicity "0..2" in the *Table* participant of Fig. 1 (right side) is transformed into the OCL constraint shown at the bottom.

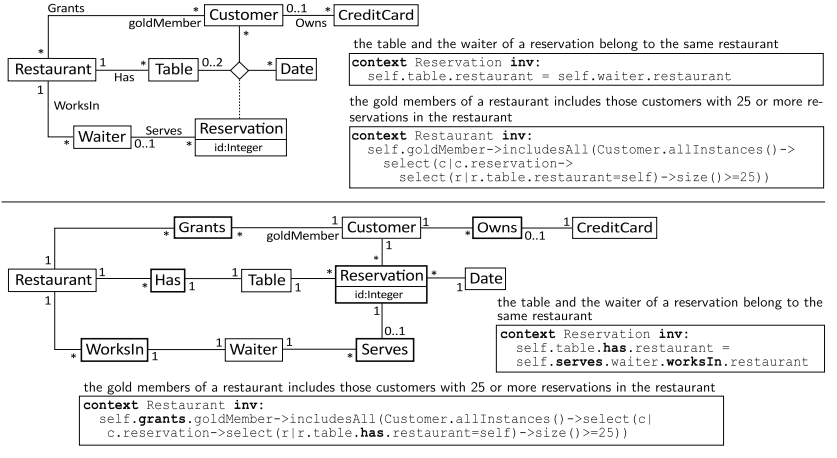

Fig. 2. A fragment of conceptual schema (up) and its version with reifications (down)

After reifying all the associations in a schema $\mathcal{CS} = \langle \mathcal{E}, \mathcal{R}, \mathcal{C} \rangle$ we obtain a new schema $\mathcal{CS}^+ = \langle \mathcal{E}^+, \mathcal{R}^+, \mathcal{C}^+ \rangle$. The set of entity types $\mathcal{E}^+$ contains the entity types $\mathcal{E}$ of $\mathcal{CS}$, and the entity types $\mathcal{E}_\mathcal{R}$ from the reifications of the initial associations ($\mathcal{E}^+ = \mathcal{E} \cup \mathcal{E}_\mathcal{R}$). The associations in $\mathcal{R}^+$ are the binary intrinsic associations that connect each $e_r \in \mathcal{E}_\mathcal{R}$ with its participants $e \in \mathcal{E}$. Finally, $\mathcal{C}^+$ contains the same constraints as in $\mathcal{C}$ including the required navigation changes to match the new schema $\mathcal{CS}^+$, and the uniqueness constraints introduced in the reifications.

Figure 2 (top) shows an example of conceptual schema with 6 associations (one of them, an association class of degree 3) and two OCL constraints that describes a fragment of an information system for reservations in restaurants. Figure 2 (bottom) shows the same schema with reifications (marked with bold rectangles). Note that the two constraints include the new navigations (bold text) to match with the new schema. Basically, a navigation $e \rightarrow e'$ has to be changed by including the navigation to the entity type that reifies the association in the middle of the expression, producing a navigation like $e \rightarrow e_r \rightarrow e'$. The reification of the schema produces 6 additional uniqueness constraints and a constraint to preserve the cardinality 0..2 in *Table* as indicated in Fig. 1. These 7 constraints are not shown in Fig. 2 for the sake of simplicity.

4 Methods for Computing the Importance

In this section we present the definition of three methods for computing the importance of associations in a schema. The methods are an adaptation of methods that compute the importance of entity types to associations. Each method

is followed by a brief description showing its details, and its application to the example of Fig. 2. We apply the methods to both entity types and associations, but the focus of the paper is on associations. Concretely, these methods require to be applied to the schema with reifications (bottom of Fig. 2) in order to compute the importance of the entity types e_r that are reifications of associations. Then, such importance is directly the importance of each association $r \in \mathcal{R}$ of the original schema because each e_r is the representation of an association r in the schema with reifications.

4.1 Occurrence Counting

There exist different kinds of methods to compute the importance of elements in a schema. The simplest family of methods is that based on *occurrence counting* [5,12,22], where the importance of a schema element is equal to the number of characteristics it has in the schema. Therefore, the more characteristics about an element, the more important it will be.

Our approach adapts the occurrence counting methods described in [24] to compute the importance of associations taking into account the characteristics an association may have in a schema. Those include the number of participants, its multiplicities, and its usage in OCL expressions to navigate the schema. Formally, the importance $\mathcal{OC}(r)$ of an association $r \in \mathcal{R}$ is defined as,

$$\mathcal{OC}(r) = |conn(e_r)| + |rconn(e_r)|$$

where the previous metrics for entity types are applied to the entity type e_r that appears from the reification of r, as explained in Sec. 3. Concretely, $|conn(e_r)|$ is the number of associations where e_r participates, which can be easily mapped to the number of connections that the original association r has in the schema. In the same way, $|rconn(e_r)|$ indicates the usage in OCL expressions of the entity type e_r, and consequently, the usage of r.

Table 3 shows the results of applying the occurrence method to the example in Fig. 2. Note that *Reservation* is the association that has a greater importance due to its number of participants and its usage in OCL navigations. As expected, the association *Owns* has a lower importance because it does not participate in any OCL expression apart from the uniqueness constraints of its reification (as shown in Fig. 1 for a binary association).

We also show in Tab. 3 the importance of entity types computed with the same method (the sum of $|conn(e)|$ and $|rconn(e)|$). *Restaurant* is the most important entity types according to the method, followed by *Customer* and *Table*.

4.2 Link Analysis

Link-analysis methods [23,22] define the importance of a schema element as a combination of the importance of the schema elements connected to it. Therefore, the more important the elements connected to a schema element are, the more important such schema element will be. In these methods the importance is

Table 3. Occurrence counting method applied to schema of Fig. 2

$r \in \mathcal{R}$	$\|conn(e_r)\|$	$\|rconn(e_r)\|$	$\mathcal{OC}(r)$	$e \in \mathcal{E}$	$\|conn(e)\|$	$\|rconn(e)\|$	$\mathcal{OC}(e)$
Grants	2	8	10	CreditCard	1	4	5
Has	2	13	15	Customer	3	9	12
Owns	2	7	9	Date	1	2	3
Serves	2	9	11	Restaurant	3	16	19
WorksIn	2	10	12	Table	2	10	12
Reservation	4	21	25	Waiter	2	9	11

shared through connections, changing from an element-centered philosophy to a more interconnected view of the importance.

Our approach adapts the link-analysis methods to compute the importance of entity types that are described in [24] to be used with associations. Concretely, we follow the same approach as the extended version of the EntityRank method (see Sec. 3.4 of [24]), which is based on Google's PageRank [4]. Each entity type in the schema is viewed as a state and each association between entity types as a bidirectional transition between them.

The link-analysis method we propose requires the reification of all the associations in the schema. Thus, we can compute their importance as in the case of entity types. Concretely, from a link-analysis perspective the importance of an entity type is the probability that a random surfer exploring the schema arrives at that entity type with random jumps (q component) or by navigation through associations ($1 - q$ component). Therefore, the resulting importance of the entity types in the schema with reifications $e \in \mathcal{E}^+$ correspond to the stationary probabilities of the Markov chain, given by:

$$\mathcal{LA}(e) = \frac{q}{|\mathcal{E}^+|} + (1-q)\left(\sum_{e' \in conn(e)} \frac{\mathcal{LA}(e')}{|conn(e')|} + \sum_{e'' \in rconn(e)} \frac{\mathcal{LA}(e'')}{|rconn(e'')|}\right)$$

Once we have the importance of all entity types $e \in \mathcal{E}^+$ in the schema with reifications, the next step is to obtain the importance of associations by analyzing the cases of those entity types $e \in \mathcal{E}_\mathcal{R}$ that are the reification of associations separately. Formally,

$$\mathcal{LA}(r) = \frac{\mathcal{LA}(e_r)}{\sum_{e \in \mathcal{E}_\mathcal{R}} \mathcal{LA}(e)}$$

where $\mathcal{LA}(r)$ is the relative importance of the association $r \in \mathcal{R}$, taking into account the importance $\mathcal{LA}(e_r)$ of its reification and the importance of the rest of reifications of associations in the schema $\mathcal{CS}^+$.

Table 4 shows the results of applying the link analysis method to associations and entity types of the example in Fig. 2. We forced that $\sum_{e \in \mathcal{E}} \mathcal{LA}(e) = 1$. Note that the link-analysis method discovers that the most important association is

Reservation but do not obtain big differences between the rest of associations. This situation occurs because of the small size of our example. According to our experience [24], this method should be used with schemas of larger sizes.

Table 4. Link analysis method applied to schema of Fig. 2

$r \in \mathcal{R}$	$\|conn(e_r)\|$	$\|rconn(e_r)\|$	$\boldsymbol{\mathcal{LA}(r)}$	$e \in \mathcal{E}$	$\|conn(e)\|$	$\|rconn(e)\|$	$\boldsymbol{\mathcal{LA}(e)}$
Grants	2	8	0.13	CreditCard	1	4	0.11
Has	2	13	0.16	Customer	3	9	0.21
Owns	2	7	0.15	Date	1	2	0.08
Serves	2	9	0.14	Restaurant	3	16	0.26
WorksIn	2	10	0.14	Table	2	10	0.17
Reservation	4	21	0.28	Waiter	2	9	0.17

4.3 Betweenness Centrality

Betweenness, in graph theory and network analysis [1], is a measure of the centrality of a vertex or an edge within a graph. It indicates the relative importance of such vertex/edge within the graph [2,3]. Since a conceptual schema can be seen as a graph with nodes (entity types) and edges (associations), it is possible to adapt the measure of node betweenness from graphs to compute the importance $\mathcal{BC}(r)$ of associations. Figure 3 shows a mini-example where *E5* is the most central (appears in a higher number of shortest paths) entity type, followed by *E3* and *E7*.

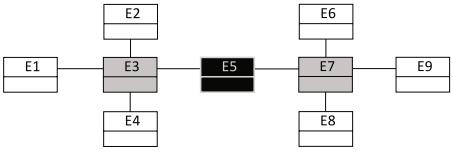

Fig. 3. Example of betweenness centrality of entity types in a conceptual schema

Basically, those associations that belong to more navigation paths in OCL and that their reifications are central in the schema are meant to be important. We firstly compute the betweenness of the entity types $e_r \in \mathcal{E}_\mathcal{R}$ that are reifications of associations $r \in \mathcal{R}$ by using the schema with reifications $\mathcal{CS}^+$. Formally,

$$\mathcal{BC}(e_r) = \sum_{e,e' \in \mathcal{E}^+} \frac{\mathcal{N}_{e,e'}(e_r)}{\mathcal{N}_{e,e'}},$$

where $\mathcal{N}_{e,e'}$ is the number of shortest paths between a pair of entity types $e, e' \in E^+$ in the schema with reifications traversing intrinsic associations $r \in \mathcal{R}^+$, and $\mathcal{N}_{e,e'}(e_r)$ is the number of those paths that go through e_r, which results from the reification of an association r.

Once we have the importance in the schema with reifications, the next step consists of obtaining the importance of associations of the original schema as in the case of the link-analysis method. Formally,

$$\mathcal{BC}(r) = \frac{\mathcal{BC}(e_r)}{\sum_{e \in \mathcal{E}_\mathcal{R}} \mathcal{BC}(e)}$$

To compute the shortest paths between entity types in the schema with reifications, we give a different length to each intrinsic association $r \in \mathcal{R}^+$ that appears in such schema, denoted by $\delta(r)$, according to its usage in OCL expressions. Concretely, we assign a shorter length for those intrinsic associations that are more navigated in order to favor their selection in the computation of shortest paths. To do so, we define in Table 5 a new measure $rconn(r)$ that computes the OCL navigations where an association r is traversed in the constraints $\mathcal{C}^+$.

Table 5. Definition of metrics for $r \in \mathcal{R}^+$

Notation	Definition
$rconn(r)$	$= \uplus_{c \in \mathcal{C}^+} \{\{e, e'\} \in links_{nav}(c) \mid association(\{e, e'\}) = r\}$
$\delta(r)$	$= \mathcal{M} - \lvert rconn(r)\rvert + 1$
$\mathcal{M}$	$= max_{r \in \mathcal{R}^+}(\lvert rconn(r)\rvert)$

Additionally, Table 5 includes the definition of the length $\delta(r)$ of an association of the schema with reifications. Note that $\mathcal{M}$ is the maximum number of navigations where an association $r \in \mathcal{R}^+$ is traversed. Therefore, those associations r types without occurrences in OCL expressions ($\lvert rconn(r)\rvert = 0$) will have a greater length (concretely, the greatest, $\delta_{max} = \mathcal{M} + 1$) than those with a big amount, and therefore their participation in shortest paths will be lower. On the contrary, the associations with a number of navigations closer to the maximum ($\lvert rconn(r)\rvert \approx \mathcal{M}$) will have a shorter length ($\delta \approx 1$), and will participate in shortest paths. Therefore, this approach uses the navigations through associations described in OCL expressions to compute the shortest paths in an importance-related way.

Table 6. Betweenness centrality method applied to schema of Fig. 2

$r \in \mathcal{R}$	$\mathcal{BC}(r)$	$\mathcal{BC}_\delta(r)$	$e \in \mathcal{E}$	$\mathcal{BC}(e)$	$\mathcal{BC}_\delta(e)$
Grants	0.14	0.07	CreditCard	0	0
Has	0.06	0.1	Customer	0.53	0.5
Owns	0.17	0.16	Date	0	0
Serves	0.13	0.16	Restaurant	0.25	0.19
WorksIn	0.07	0.05	Table	0.13	0.21
Reservation	0.43	0.46	Waiter	0.09	0.1

Table 6 shows the results of applying the betweenness centrality method to the example in Fig. 2. Note that $\mathcal{BC}_\delta$ takes into account the lengths $\delta(r)$ whereas $\mathcal{BC}$ does not. We forced that $\sum_{e \in \mathcal{E}} \mathcal{BE}(e) = 1$ and $\sum_{e \in \mathcal{E}} \mathcal{BE}_\delta(e) = 1$.

It is important to observe that without lengths the path from *Restaurant* to *Customer* through the association *Grants* is shorter than traversing the entity

type *Table*. By contrast, since that second path is more navigated in OCL constraints, taking into account lengths we observe a significant reduction in the importance of *Grants* (0.14 to 0.07) and an increment in the importance of *Has* and *Table* (0.06 to 0.1 and 0.13 to 0.21, respectively). Therefore taking into account the lengths in associations according to their usage in OCL expressions is a more realistic approach to compute the importance.

5 Experimental Evaluation

We have implemented the three methods described in the previous section and we have evaluated them using five distinct case studies: the osCommerce [21], the EU-Rent [9], the UML2 metaschema [13], a fragment of the HL7 schemas [26], and the OpenCyc ontology [6]. For the case of the *betweenness centrality* method we used an existing implementation of the Brandes algorithm [7,3]. Table 7 summarizes the main characteristics of the five schemas.

Table 7. Schema elements of the case studies

	Entity Types	Associations	Constraints
osCommerce schema	346	183	457
EU-Rent schema	185	152	283
UML2 metaschema	293	377	188
HL7 schemas	2695	228	9
OpenCyc	2951	1385	0

In case of two or more entity types or associations get the same importance, our implementation is non-deterministic: it might rank first any of those. Some enhancements can be done to try to avoid ranking equally-important entity types or associations in a random manner, like prioritizing those with a higher amount of attributes or a higher amount of participations in OCL expressions (or any other measure) in case of ties. However, this does not have an impact to our experimentation. In the following, we summarize the main studies we have performed with the results of the application of the three methods.

5.1 Time Analysis

The first study we have made measures the execution time of each importance-computing method when applied to each of the five schemas. It is clear that a good method does not only require to achieve relevant results, but it also needs to present them in an acceptable time according to the user requirements. To find the time spent by our method it is only necessary to record the time lapse between the start of the method, and the receipt of the rankings of associations and entity types.

Figure 4 shows the execution time (in seconds) of all three methods in an Intel Core 2 Duo 3GHz processor with 4GB of DDR2 RAM. According to the results,

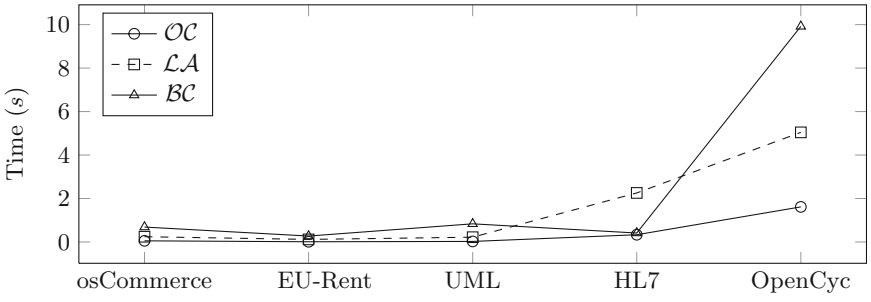

Fig. 4. Execution time between methods for each schema

the *occurrence-counting* method ($\mathcal{OC}$) is the fastest method for all the schemas, because of its simplicity with respect to the other two methods. On the contrary, the *betweenness-centrality* method ($\mathcal{BC}$) is the slowest one due to its bottleneck on computing the shortest paths between schema elements. Despite that, the $\mathcal{BC}$ method performs better than the *link-analysis* method ($\mathcal{LA}$) in the case of the HL7 due to the reduced number of associations it contains (in comparison to the number of entity types), which dramatically reduces the previous bottleneck of the shortest-paths computing process.

Consequently, if the method is intended to be used for determining the importance of associations and entity types in a context where the target schema is rapidly evolving through changes, it is better to select a faster method like the $\mathcal{OC}$, or even the $\mathcal{LA}$ (taking into account the size of the schema). Otherwise, if the context is static and the schema does not change, the method to select may be any of the three (the importance could be pre-calculated without runtime consumption).

5.2 Correlation between Methods

We apply the three methods to each of the five schemas. Each method can compute the importance of associations and entity types, and therefore produces two different rankings of schema elements: the one of entity types and the one of associations. Our research aims to know which methods give similar results, in order to select the simpler method in any case. Thus, we study the correlation between methods by analyzing the correlation of the rankings they produce.

Figure 5 shows, for each pair of methods, the results obtained in the correlation analysis for the rankings of importance of associations. Our aim is to know whether it is possible to compare the results of the importance methods and to search for a common behavior. We can observe that the most correlated methods to compute the importance of associations are the *occurrence-counting* and *link-analysis* methods, although there exists a certain variability in the results, denoted by the fact that their correlation in larger schemas tend to decrease.

Figure 6 shows, for each pair of methods, the results obtained in the correlation analysis for the rankings of importance of entity types. In this case, although

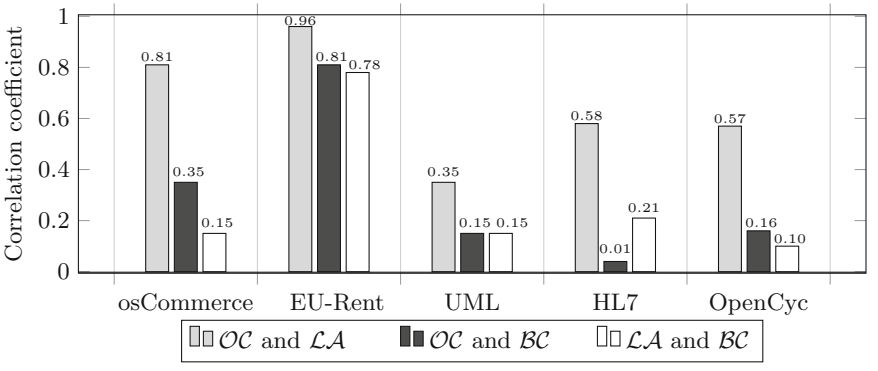

Fig. 5. Correlation between methods for each schema in the case of the importance of associations

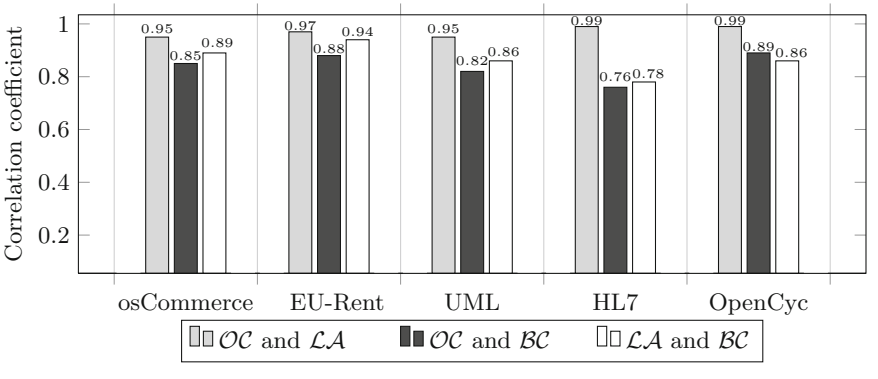

Fig. 6. Correlation between methods for each schema in the case of the importance of entity types

the pair of methods with a higher correlation are the same than in the case of associations ($\mathcal{OC}$ and $\mathcal{LA}$), all the methods produce similar rankings. It means that the three methods of importance are more consistent when applied to entity ranks than when applied to associations.

The fact that the pair of methods $\mathcal{OC}$ and $\mathcal{LA}$ produce more similar results between them than with the *betweenness-centrality* method ($\mathcal{BC}$) is due to the different approach each method follows. On one hand, $\mathcal{OC}$ and $\mathcal{LA}$ compute the importance of schema elements by counting the characteristics they have (and the characteristics of the elements connected to them, in $\mathcal{LA}$) in the schema. On the opposite, $\mathcal{BC}$ computes the importance by analyzing the topology and structure of the schema measuring the shortest paths between elements and their participation in those shortest paths.

Therefore, to select a method or another depends on the approach the user wants to follow to compute the importance of associations. An approach closer to counting the number of characteristics the associations have, will choose $\mathcal{OC}$

or $\mathcal{LA}$, while an approach closer to select those associations that are more central in the schema will choose $\mathcal{BC}$. For the case of entity types, as shown in Fig. 6, the selection of a method or another has a lower impact in the obtained results, because of the greater correlation between them.

5.3 Impact of the Reifications

We have analyzed the results of the three methods when applied to compute the importance of the entity types of each of the five test schemas in their original form ($\mathcal{CS}$) and after their reification ($\mathcal{CS}^+$). The resulting correlation between $\mathcal{CS}$ and $\mathcal{CS}^+$ is an indicator of the impact that reifications have in the importance of entity types.

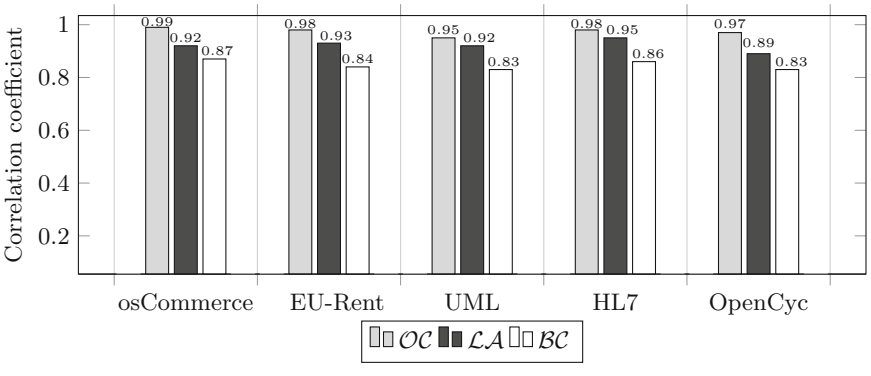

Fig. 7. Correlation between results before and after reifications for the case of the importance of entity types

Figure 7 shows that the transformation of the associations into entity types through the reification process has an impact in the importance of entity types by introducing changes in the resulting rankings. Those changes are mainly produced by the addition of implicit associations after reifications to maintain the connections. However, the correlation between the rankings of importance of entity types in $\mathcal{CS}$ and $\mathcal{CS}^+$ is close to 1, which means that the impact of reifications is minimal. Therefore, although $\mathcal{CS}^+$ changes with respect to $\mathcal{CS}$, reifying associations produces a low impact on the computed importance of entity types.

6 Conclusions and Further Work

The visualization and the understanding of large conceptual schemas require the use of specific methods. These methods generate indexed, clustered, summarized or focused schemas that are easier to visualize and understand. Almost all of these methods require computing the importance of each element in the schema but, up to now, only the importance of entity types has been studied in the literature.

The computed importance induces an ordering of the elements, which plays a key role in the steps and result of the methods that deals with large schemas. We

have proposed three methods to compute the importance of associations, and also entity types. The methods we describe are based on *occurrence-counting* ($\mathcal{OC}$), *link-analysis* ($\mathcal{LA}$), and *betweenness-centrality* ($\mathcal{BC}$). Our approach transforms the schema by reifying the associations into entity types. As a result, we use existing importance-computing methods from the literature with minor modifications to be able to work with associations.

We have implemented the three methods in a prototype tool and we have experimented them with five large real-world conceptual schemas. The results we obtained indicate that the quickest method is the $\mathcal{OC}$, which must be selected if the interaction context is dynamic and the user wants real-time feedback. Furthermore, we observe that the computed rankings of importance for entity types have a greater similarity independently of the selected method which indicates that all three methods are indistinguishable in that aspect. Conversely, for the case of the rankings of associations, the selection of a method or another has an impact in the obtained results.

The combination of the importance of entity and associations our methods compute can be applied to several techniques to deal with large conceptual schemas in order to reduce the effort a non-expert user must do to understand the knowledge within the schema. An example is the construction of reduced schema summaries with the top of both importance rankings to give a simple view of the schema contents. We plan to continue our work in that direction.

Acknowledgements. This work has been partly supported by the Ministerio de Ciencia y Tecnologia and FEDER under project TIN2008-00444/TIN, Grupo Consolidado, and by Universitat Politècnica de Catalunya under FPI-UPC program.

References

1. Boccaletti, S., Latora, V., Moreno, Y., Chavez, M., Hwang, D.: Complex networks: Structure and dynamics. Physics Reports 424(4-5), 175–308 (2006)
2. Brandes, U.: A faster algorithm for betweenness centrality. Journal of Mathematical Sociology 25, 163–177 (2001)
3. Brandes, U.: On variants of shortest-path betweenness centrality and their generic computation. Social Networks 30(2), 136–145 (2008)
4. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. Computer Networks and ISDN Systems 30(1-7), 107–117 (1998), Proceedings of the 7th International World Wide Web Conference
5. Castano, S., De Antonellis, V., Fugini, M.G., Pernici, B.: Conceptual schema analysis: techniques and applications. ACM Transactions on Database Systems 23(3), 286–333 (1998)
6. Conesa, J., Storey, V.C., Sugumaran, V.: Usability of upper level ontologies: The case of ResearchCyc. Data & Knowledge Engineering 69(4), 343–356 (2010)
7. Dutot, A., Guinand, F., Olivier, D., Pigné, Y.: Graphstream: A tool for bridging the gap between complex systems and dynamic graphs. EPNACS: Emergent Properties in Natural and Artificial Complex Systems (2007), `http://graphstream-project.org`
8. Egyed, A.: Automated abstraction of class diagrams. ACM Transactions on Software Engineering and Methodology (TOSEM) 11(4), 449–491 (2002)

9. Frias, L., Queralt, A., Olivé, A.: EU-Rent Car Rentals Specification. LSI Research Report. Tech. rep., LSI-03-59-R (2003), http://www.lsi.upc.edu/dept/techreps/techreps.html
10. Jaeschke, P., Oberweis, A., Stucky, W.: Extending ER Model Clustering by Relationship Clustering. In: Elmasri, R.A., Kouramajian, V., Thalheim, B. (eds.) ER 1993. LNCS, vol. 823, pp. 451–462. Springer, Heidelberg (1994)
11. Lindland, O.I., Sindre, G., Sølvberg, A.: Understanding quality in conceptual modeling. IEEE Software 11(2), 42–49 (1994)
12. Moody, D.L., Flitman, A.: A Methodology for Clustering Entity Relationship Models - A Human Information Processing Approach. In: Akoka, J., Bouzeghoub, M., Comyn-Wattiau, I., Métais, E. (eds.) ER 1999. LNCS, vol. 1728, pp. 114–130. Springer, Heidelberg (1999)
13. Object Management Group: Unified Modeling Language (UML) Superstructure Specification, version 2.2 (February 2009), http://www.omg.org/spec/UML/2.2/
14. Object Management Group: Object Constraint Language Specification (OCL), version 2.0 (February 2010), http://www.omg.org/spec/OCL/2.2/
15. Olivé, A., Cabot, J.: A research agenda for conceptual schema-centric development. In: Conceptual Modelling in Information Systems Engineering, pp. 319–334 (2007)
16. Olivé, A.: Conceptual Modeling of Information Systems. Springer, Heidelberg (2007)
17. Olivé, A., Raventós, R.: Modeling events as entities in object-oriented conceptual modeling languages. Data & Knowledge Engineering 58(3), 243–262 (2006)
18. Schewe, K.-D., Thalheim, B.: Conceptual modelling of web information systems. Data Knowl. Eng. 54(2), 147–188 (2005)
19. Schmidt, P., Thalheim, B.: Management of UML Clusters. In: Abrial, J.-R., Glässer, U. (eds.) Borger Festschrift. LNCS, vol. 5115, pp. 111–129. Springer, Heidelberg (2009)
20. Thalheim, B.: Entity-relationship modeling: foundations of database technology. Springer, Heidelberg (2000)
21. Tort, A., Olivé, A.: The osCommerce Conceptual Schema. Universitat Politècnica de Catalunya (2007), http://guifre.lsi.upc.edu/OSCommerce.pdf
22. Tzitzikas, Y., Kotzinos, D., Theoharis, Y.: On ranking RDF schema elements (and its application in visualization). Journal of Universal Computer Science 13(12), 1854–1880 (2007)
23. Tzitzikas, Y., Hainaut, J.-L.: How to Tame a Very Large ER Diagram (Using Link Analysis and Force-Directed Drawing Algorithms). In: Delcambre, L.M.L., Kop, C., Mayr, H.C., Mylopoulos, J., Pastor, Ó. (eds.) ER 2005. LNCS, vol. 3716, pp. 144–159. Springer, Heidelberg (2005)
24. Villegas, A., Olivé, A.: Extending the methods for computing the importance of entity types in large conceptual schemas. Journal of Universal Computer Science (J.UCS) 16(20), 3138–3162 (2010)
25. Villegas, A., Olivé, A.: A Method for Filtering Large Conceptual Schemas. In: Parsons, J., Saeki, M., Shoval, P., Woo, C., Wand, Y. (eds.) ER 2010. LNCS, vol. 6412, pp. 247–260. Springer, Heidelberg (2010)
26. Villegas, A., Olive, A., Vilalta, J.: Improving the usability of HL7 information models by automatic filtering. In: IEEE 6th World Congress on Services, pp. 16–23 (2010)
27. Yang, X., Procopiuc, C.M., Srivastava, D.: Summarizing relational databases. In: 35th Intl. Conf. on Very Large Data Bases, VLDB 2009, pp. 634–645 (2009)
28. Yu, C., Jagadish, H.V.: Schema summarization. In: 32nd Intl. Conf. on Very Large Data Bases, VLDB 2006, pp. 319–330 (2006)