A Representational Analysis on Relationships in an ER Model

Lin Liu[1], Junkang Feng[2], Kaibo Xu[3]
[1,2]School of Computing, University of the West of Scotland, United Kingdom
[1]Lin.liu@uws.ac.uk, [2]junkang.feng@uws.ac.uk
[3]Business College, Beijing Union University, China
Kaibo.xu@buu.edu.cn

Abstract: - We observe that a database is a type of representation system in that data in it represents something in the real world. Thus the notion of *representation* should be highly relevant to databases especially data modeling. In this paper, we focus on identifying the representation aspect of relationships of an ER model by means of notions of representation including *information carrying* and *nomic constraint*. In particular we review a known problem, i.e., connection traps, in ER modelling. Our main findings are: A transitive nomic constraint arises when the multiplicities of participating non-transitive relationships are *:1 or 1:1; Connection traps can be accounted for in terms of the nomic constraint or the lack of it in a path, namely, a fan trap is formed when a transitive structure contains no transitive nomic constraint, and a chasm trap is the result of the partial participation of an entity in the nomic constraint of a relationship. Furthermore, even when a path is free of connection traps, the transitive nomic constraint in the path has to project to (i.e., match) a constraint in the target domain (normally the real world) in order for the path to be able to represent it.

Key-words: - Representation, ER modeling, Information carrying, Connection traps, Constraint projection.

1 Introduction

Representation is a key concept of information systems as Shanks claims that representation is at the core of the discipline of information systems [1]. The notion of representation and representation systems have been studied in great extent by many scholars over the years [2, 3, 4, 5, 6].

We observe that a database is a specific type of representation systems from which data values carry information about and thus representing states of affairs in the real world. Although serious attempts of applying information theory to the database domain have been made over the years, most solutions proposed so far have focused on specific cases, and the database community does not seem to have a clear, generally accepted concept for the information carried by the data from a database [7]. In particular, we believe, due to the fact that the notion of representation is closely related to database systems, it would seem desirable to apply the notion of representation together with the theory of semantic information to tackle some fundamental problems in databases.

Following this line of thinking, we have attempted a representational analysis to databases as part of an ongoing PhD research project. We have so far successfully discovered that notions of representation, particularly the notion of *constraint projection*, is an intellectual tool to identify the mode of representation for relations as in relational databases, to uncover the fundamental reasons of some database problems within a relation, such as data redundancies and update anomalies. Now, we shift our attention to another important part of ER modelling [8], namely relationships between entities from the viewpoint of representation including the notions of *information carrying* and *nomic constraint*. In this paper, we focus on identifying the representation aspect of relationships. We aim to identify what constrains a path (i.e., a relationship or connected multiple relationships in an ER model) in its capability of representing something in the real world by looking at the graphical representation nature of a path in an

ER model, which accounts for both afore mentioned notions of *information carrying* and *nomic constraint* for relationships of an ER model. We find that what in the real world a path of length 2 or greater than 2 can represent has to do with whether there is an information carrying relationship between the entities involved in the path, which can be formulated as whether there is a *transitive nomic constraint* that governs the path. Connection traps can be accounted for in terms of the nomic constraint or the lack of it in a path: a fan trap is formed when a transitive structure contains no transitive nomic constraint and a chasm trap is the result of the partial participation of an entity in the nomic constraint of a relationship.

2 Database Constraints in a Relationship

We observe that *constraint projection* is essential for the notion of representation [3]. Constraints can be classified as *nomic* or *stipulative*. The former are due to natural laws or regularities and the latter business rules, social conventions and regulations.

The notion of constraint projection can be used for looking at some problems in databases, e.g., database constraints. Database constraints such as functional dependencies are indispensable elements of a relational database schema, which ensure its efficiency in data storage and manipulation. In databases, relationship constraints are the ones that may be placed on entity types within a relationship. Such constraints should match the respective constraints in the "real world". This is called constraint projection of a representation system. There are many kinds of constraints that fall into this category including, for example, the requirements that a property for rent must have an owner and each branch must have staff members. Among them, the most important constraint within a relationship is so called *multiplicity* constraint.

2.1 Multiplicity

Multiplicity is defined as the number (or range) of possible occurrences (in this paper we take that 'instance' and 'occurrence' are interchangeable) of an entity type that may be related to a single occurrence of an associated entity type through a particular relationship. Multiplicity constraint is made up of cardinality and participation constraints that are denoted by two integers such as 0..1, namely that the minimum and maximum number of possible relationship occurrences for an entity participating in a given relationship type are 0 and 1 respectively.

Multiplicity captures policies or business rules that determine how entities are related, which is set up by the users or enterprise. It forms an important part of data modelling of an enterprise. Three common types of cardinality ratio in multiplicity constraints are 1:1, 1:* and *:* which stand for one-to-one, one-to-many and many-to-many relationships respectively. We argue that multiplicity constraints are stipulative constraints as they are declared by the database designer to reflect business rules and policies of an enterprise. Therefore, adding or removing them will not result in any additional information to be added or deleted apart from the information associated with the constraints *per se*.

The same idea is applicable to other business rule based constraints involved in a relationship such as the requirements that a property for rent must have an owner. They are all used to control the certain values of a relationship type. Now we use an ER diagram to illustrate a relationship with its multiplicity constraints.

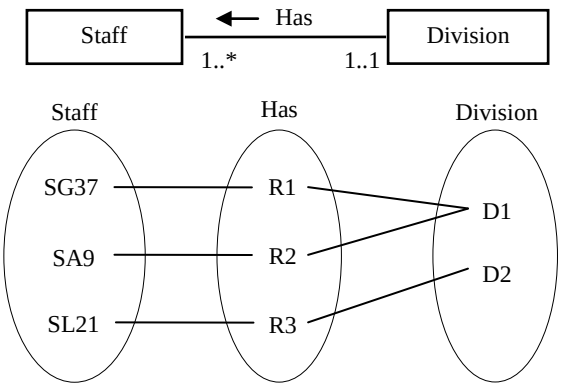

Fig. 1 the Relationship Model of Division-Has-Staff

Fig. 1 shows a binary relationship of Division-has-Staff in two different models, the ER model and semantic net. As we can see, the multiplicity constraints of this relationship are 1..* and 1..1, meaning that one member of staff must belong to only one division and a division can have one or many members of staff.

2.2 IIR

Like functional dependency that governs relationships between attributes within a relation on the type level, multiplicity specifies the association between occurrences of two different entities on the instance level. An entity occurrence may be seen as an event, for example, SG37 is the event that "SG37 happens to be a member of Staff". Certain types of multiplicity constraints ensure informational relationships between instances of two relations. This means that the existence of one instance may carry information that another instance exists, for example, if SG37 in Fig. 1 is certain, than D1 is certain. In other words, the information content of SG37 includes D1. We call such a relationship *information content inclusion relation* (IIR). More detailed analysis of IIR including all the IIR rules can be found in [9].

After establishing IIRs between instances of two entities that form a relationship, we will examine the IIRs on relationships of three possible cardinality ratios respectively. We will still use the example in Fig. 1 to illustrate our ideas.

1) *:1 and 1:1

For example, entity Staff and entity Division have a *:1 relationship, namely a member of staff works for only one division. Therefore, given a member of staff, say SL21, there is only one division D2 corresponding to it. This is, whenever SL21 is certain, D2 is certain. In term of IIR, we have I(SL21) ∋ D2. Moreover, following the idea of [10, p109], we say that there is a function from Staff to Division.

When "0" appears in the multiplicity of a relationship, it indicates that there is no connection available on some of the instances of an entity type. There is no IIR, therefore no information connections for those instances.

2) *:*

Nothing is certain in this relationship. Therefore, no IIR can be established.

Proposition 1

For two distinct sets of (random) events A and B, I(A) ∋ B, if every subset of B is in the information content of at least one subset of A.

Proof

Suppose A = $\{a_1, a_2, \ldots .. a_m\}$

B = $\{b_1, b_2, \ldots .. b_n\}$

(Comment: a_m is a subset of A, b_n is a subset of B.)

I(A) = $\sum_{i=1}^{m} I(a_i)$, I(B) = $\sum_{i=1}^{n} I(b_i)$ (1)

(Comment: Information content of the whole set is the summation of information content of its subset.)

b_i is in the information content of one subset of A given (2)

$\sum_{i=1}^{m} I(a_i)$ ∋ b_i (1), (2) (3)

I(b_i) ∋ B IIR sum rule* (4)

$\sum_{i=1}^{m} I(a_i)$ ∋ B (3), (4) and IIR transitive rule (5)

I(A) ∋ B (1), (5) (6)

*****IIR sum [9]: If $Y = X_1 \cup X_2 \ldots \cup X_n$, then I($X_i$) ∋ Y for i = 1,…, n

This rule states that if a random event Y is the disjunction of a number of random events X_i, then it is in the information content of any of the latter. For example, given two random events, ψ1 someone *a* holds a coloured (green, red, blue or yellow) ball in her hand and ψ2 someone *a* holds a blue ball in her hand, then I(ψ2) ∋ ψ1, namely, whenever ψ2 happens, ψ1 happens. More detailed analysis regarding to IIR sum rule can be found in [9].

Based on this, we observe another proposition as follows.

Proposition 2

*For two distinct entities of A and B participating in a relationship, I(A) ∋ B if the cardinality ratio of the multiplicity of the relationship is *:1 or 1:1 with a total participation on both sides.*

This is because given the multiplicity of *:1 or 1:1 from A to B, every occurrence in B is in the information content of at least one occurrence of A. According to Proposition 1, the whole set of B is in the information content of A. By examining the example shown in Fig. 1, we observe that I(Staff) ∋ Division is true.

Next, we will show how the concept of IIR copes well with our representation framework in solving connection problems of ER modelling.

3 Transitive Relationships and Transitive Nomic Constraints

So far we have learned that the multiplicity constraint of a relationship governs how occurrences of entities are related to each other across entities. Most importantly, we have established that by means of the multiplicity constraint of a relationship between two entities, we can establish information content inclusion relations between occurrences of the entities. We observe that the concept of IIR combined with the notion of representation may be proven an intellectual tool in deriving hidden information relations, as well as solving particularly the connection problems of ER modelling such as connection traps.

3.1 Transitive Relationships

In database design, after having identified entities and attributes, an ER schema is usually formed by establishing direct links between entities, which are called "non-transitive" relationships. The term of path can be used to describe the links between entities and the length of a path indicates how many shorter paths are involved in the path. The length of a path in a non-transitive relationship is 1.

In ER modelling, an entity can be involved in two or more "non-transitive (direct linkages of entities)" relationships. We call such an ER structure a "transitive relationship". The length of a path in a transitive relationship is greater than 1. An example of such a structure is illustrated below.

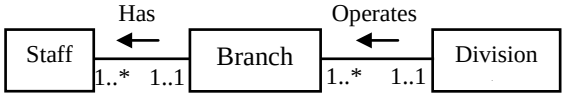

Fig. 2 an Example of a Transitive Relationship

A transitive relationship could be complex and problematic. According to [11], there may be an implied connection, i.e., the connection between "Staff" and "Division" that is inferred from two relationships "Has" and "Operates". Such connections, which we call transitive connections, are a result of two or more connected paths of length 1.

As part of an ER schema, a transitive connection is capable of storing certain information. In order to retrieve such information, a database transaction may have to navigate through part or whole of the ER schema. A problem arising here is that it is not always certain if the navigation is legitimate and if the results of the transaction are trustworthy.

Proposition 3

*A transitive connection contains a nomic constraint if the cardinality ratios of the multiplicities of its non-transitive relationships are *:1 or 1:1.*

Proof

Suppose that there are two non-transitive relationships: R_1 between entity types A and B, and R_2 between entity types B and C.

Multiplicity of R_1(A, B) and R_2(B, C) are *:1 or 1:1	given	(1)
I(A) ϶ B, I(B) ϶ C	Proposition 2	(2)
I(A) ϶ C (2)	and transitive rule of IIR	(3)
A ⊢ C	(3) and Definition of IIR	(4)

Here A ⊢ C is a *constraint* in that if there is an *a* being of A, then there must be a *c* that is of C. Such an extremely general definition of constrains are not the same as database constraints as widely accepted in the database literature. The latter is concerned with logical limitations on data values and associations between data values.

Moreover, the above constraint A ⊢ C is derived by means of the mathematically sound IIR transitive rule, which is in turn based on some inherent topological characteristics of a path of length 2 in an ER diafram, and therefore it is a nomic constraint. There could be many types of nomic constraints based on natural laws including mathematics and inherent geometric or topological characteristics of a representation system. In this paper, we are only looking at nomic constraints based on mathematics that are concerned with database constraints.

3.2 Nomic Constraints due to transitive connections

A nomic constraint in representations may indicate something in the real world that is actually not true. For example, if we want to express information of "a house is located between two roads" by using a diagram, we have to draw a symbol of the house between the symbols of the roads in the diagram. This would generate many more pieces of 'information' inevitably than the desired information "a house is located between two roads" due to the inherent geometric and topological characteristics of the diagram.

Such 'information' may include 'the house is closer to road A than road B', which may not be true, and thus is not information at all as information requires to be contingently true according to [12]. In our situation however, as nomic constraints like the A ⊢ C above are there due to mathematically sound IIR rules, what they indicate is always true.

Definition
*We call a nomic constraint that is derived from *: 1 or 1: 1 multiplicities of the transitive connection of a relationship a transitive nomic constraint. That is, we wish to call a constraint that is due to the nomic nature of transitive connection of a relationship a 'transitive nomic constraint'.*

Proposition 4
The existence of a transitive nomic constraint across a path of length 2 is equivalent to that there is an information carrying relationship between the two entities in the path that are involved in a transitive relationship. In other words, the former is a necessary and sufficient condition for the later.

To justify the "sufficient condition", we assume that there is a transitive nomic constraint A ⊢ C existing in a relationship between entities A, B and C. By the afore-mentioned definition of constraint, if there is an *a* being of A, then there must be a *c* that is of C, that is, C is in the information content of A. In other words, A carries information that C.

To prove the "necessary condition", we assume that A carries information that C in a path that involves two relationships between three entities A, B and C. By definition of information content, if there is an *a* being of A, then there must be a *c* that is of C. That is, A ⊢ C.

By virtue of above ideas, we can check paths in an ER diagram. For example, after having checked the path shown in Fig. 2, we would know that the path is sound in that if a database query asks for staff and divisions for which they work, the path is capable of providing a correct answer. This is partly because there is a transitive nomic constraint in the path, which enables the path to give a certain division for a given member of staff. However, a path of length 2 may well be unsound in terms of information provision. One type of such problems is called *connection traps*.

4 Connection Traps

Connection traps are common problems in constructing an ER schema, in obtaining information by querying database, and in entering data into a database to represent certain information. Any ER schema could contain potential connection traps. Some of them might be trivial to users and some others can be avoided by restructuring the ER schema. In either way, their existence must be recognised.

Although the problem of connection traps has been studied by many scholars over the years from different angles, some issues relating to connection traps are still ambiguous. For example, how to systematically identify a system having connection traps has not been fully studied in the literature. After establishing links between the notion of representation and the database theory, we can look at the problems of connection traps from the prospective of the representation especially nomic constraints.

4.1 What Causes Connection Traps

In the literature, there are two common views on why connection traps happen. Howe [10] suggests that connect traps are due to a misinterpretation of the meaning of certain relationships. Feng [11] extends Howe's idea and points out the cause of connection traps is that a database is mistakenly taken to be able to represent (or to have represented) some information that it is unable to represent.

Although Howe provides what the misinterpretation of the meaning means in general sense, he failed to address why it is the case for each individual type of connection traps. Feng observes the weakness of Howe's interpretation on connection traps and examines the root of connection problems in a detailed and boarder sense. He uses the notion of (true or false) *semantic connection* as an intellectual tool to identify connection traps. For example, fan traps are due to false semantic connections not being recognised and

chasm traps are the victims of the true semantic connections that are taken to have been represented or can be represented in fact not being represented and cannot be represented.

We are convinced that Feng's approach on connection traps is probably more insightful than many others. Some of the concepts he puts forward are based on semantic information theories, which makes his ideas somewhat a little hard to understand. To alleviate this and to extend his work (ibid.), we now attempt to use the notion of *transitive nomic constraint* to re-look at the same problem with a view to gaining further insights.

Fan Traps

Two well-known types of connection traps are fan traps and chasm traps. A relationship fan exists where two or more relationship occurrences of the same type fan out from the same entity occurrence. When two relationship fans connect back to back to the same instance of an entity, the relationship types and the entity types in question are said to constitute a "coupling fans" structure, which may constitute a fan trap. The characteristic of coupling fans may be formulated as the multiplicity of a path of length 2 being *:1/1:*.

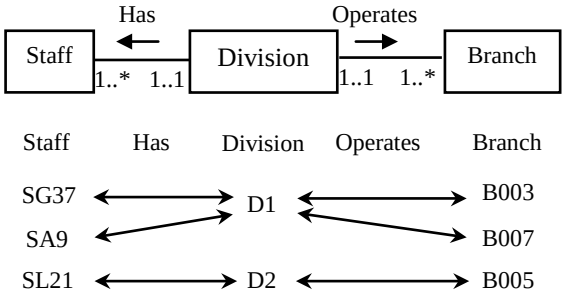

Fig. 3 an Example of Fan Trap with Occurrence Diagram

If we swap the position of "Branch" and "Division" in Fig. 2, the resultant structure is shown in Fig. 3. The multiplicity of relationship is changed from *:1/*:1 to *:1/1:* which results in the path being of a fan trap structure. A fan trap may bring undesirable consequences to database queries. For instance, if we want to know at which division staff member SG37 works, the database would be unable to provide an answer from the above structure.

One question arising here is why a path with an embedded fan trap is problematic on information carrying. In order to answer this question, we need first to analyse the constraint that governs information transmission in such a path. To reiterate, a transitive connection is a derived relationship due to two non-transitive relationships joining each other, for example, the relationship between entity types "Staff" and "Branch" in Fig. 3 is such a relationship. In order to find an answer of the above question, we present another proposition below.

Proposition 5. *The cause for a fan trap is that there is no transitive nomic constraint between the end nodes* (*entities*) *of a path.*

Justification. The multiplicity constraints of a fan trap structure is *:1/1:*, which does not satisfies the condition for transitive nomic constraints, which we discussed in Proposition 3. Moreover, according to Proposition 4, whenever there is no transitive nomic constraint, there is no information carrying relationship between end nodes. Therefore a database query concerning the links between the two end nodes would give ambiguous answers, which is why such a structure is called a 'fan trap'.

In ER schema design, the existence of fan traps can be avoided by restructuring the sequence of entities to ensure a transitive nomic constraint. For example, we swap the position of "Division" and "Branch" in Fig. 3 back to the original structure shown in Fig. 2, and it is free of fan traps.

It should be noted that a path free of fan traps does not guarantee successful representation. The reason for this is as follows: Having a transitive nomic constraint would give us an information carrying relation, for example, a path may be capable of giving a certain division for a given member of staff. That is, there is an association between the certain division and the given member of staff. However, 'information carrying' gives an association, but does not concern itself with the meaning of such an association. For example, the path shown in Fig. 3 would fail to represent 'staff work for divisions' if there is no business rule that if a branch, say B003, has a member of staff, say SG37, and a division, say D1, operates B003, then SG37 works for D1. In such a case, we say that there is a mis-match between the transitive nomic constraint and a constraint in the real

world. This is called 'constraint mis-projection' [3].

Therefore we conclude that the sufficient condition for a path to be capable of representing something concerning a transitive relationship in the real world is that *there is a transitive nomic constraint in the path* and furthermore *this constraint projects to a real world constraint.*

Chasm Traps

A chasm trap occurs where an ER schema suggests the existence of a relationship between entity types, but such a relationship might not actually exit between certain entity occurrences [13]. It happens due to one or more relationships with a minimum multiplicity of zero forming part of the pathway between related entities. A typical chasm trap is illustrated in Fig. 4.

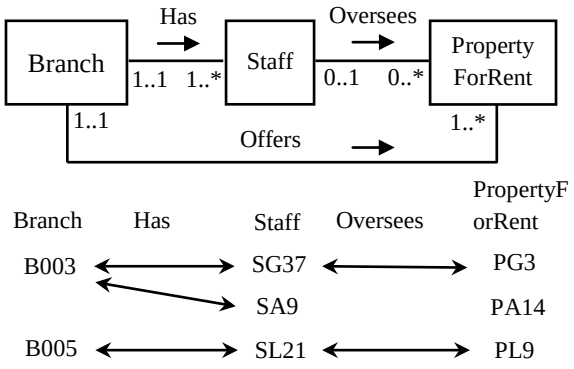

Fig. 4 an Example of Chasm Trap with Occurrence Diagram

In order to understand why chasm traps bring problems to a database, we shall again examine the multiplicity constraints embedded in a path in order to identify possible nomic constraint projections.

By looking at the above diagram at the schema level, we find the combined multiplicity constraint of 1:*/1:* from "Branch" to "PropertyForRent". Conversely, the multiplicity constraint from "PropertyForRent" to "Branch" is *:1/*:1, which satisfies the condition of a transitive nomic constraint according to Proposition 3. That is, there exists a transitive nomic constraint in the transitive connection between "PropertyForRent" and "Branch" at the type level. However, the partial participation constraint (indicated by '0' in the minimum side) reveals that some occurrences of "PropertyForRent" cannot connect to occurrences of "Branch" via occurrences of "Staff" due to the partial participation. The partial participation of an entity in a relationship entails that some entity instances are not involved in the transitive nomic constraint, and thus are not involved in the information carrying relationship formulated as the transitive nomic constraint. Because it is the individual representations (tokens) that carry information about other objects (targets), databases queries that rely on such information carrying relationship would fail on those entity instances that do not participate in one or more relationships in the path. Therefore we conclude that the sufficient condition for every instance of a path to be capable of representing something concerning a transitive relationship in the real world is that *there is a transitive nomic constraint in the path, this constraint projects to a real world constraint,* and furthermore, *the every instance of the path participate in the transitive nomic constraint*.

The existence of chasm chaps has to be recognised and eliminated. We can build a direct relationship between two entities that are transitively connected to bypass the chasm trap. For example in Fig. 4, a new relationship "Offers" is created between "Branch" and "PropertyForRent".

5 Conclusions

In this paper, we have presented our work on some connection problems concerning relationships between entities in an ER model by applying notions of representation.

The database multiplicity constraints in a path were examined in order to identify possible transitive nomic constraints. Two typical types of connection traps, fan traps and chasm traps were then investigated.

A transitive connection contains a transitive nomic constraint if the multiplicities of its non-transitive relationships are *:1 or 1:1, which makes information carrying between nodes (representations) of a path possible. A fan trap is formed when a transitive structure contains no transitive nomic constraint. A chasm trap is formed due to a partial participation of entity instances in the nomic constraint of a path. Furthermore, even when a path is free of connection traps, the transitive nomic constraint in the path has to project to (i.e., match) a constraint in the target domain (normally the real world) in order for the path

to be able to represent it. These we believe shed some light on the representational capability of a relationship in an ER model.

This paper has been focusing on relationships of an ER model. The ideas can be extended however to other problems, for example, connection problems on a non-transitive relationship or more complex relationships. They should also be applicable to other database models, such as Object-Oriented, Network and Document-Oriented models.

6 Acknowledgements

This work has been funded by Scientific Research Common Program of Beijing Municipal Commission of Education (No. KM201311417011), the Importation and Development of High-Caliber Talents Project of Beijing Municipal Institutions (No. CIT&TCD201404089) and Funding project of Beijing Philosophy and Social Science Research Program (No. 11JGB039).

References

[1] G. Shanks, Semiotic approach to understanding representation in information systems. *Proceedings of the information Systems Foundations Workshop, Ontology, Semiotics and Practice*. Macquarie University, 1999.

[2] R. Weber, Ontological Foundations of Information Systems, Coopers & Lybrand Accounting Research Methodology. Monograph No. 4. Melbourne, 1997.

[3] A. Shimojima, On the Efficacy of Representation, Ph.D. Thesis, the Department of Philosophy, Indiana University, 1996.

[4] J. Barwise, J. Seligman, Information Flow: the Logic of Distributed Systems. Cambridge University Press, Cambridge, 1997.

[5] B. R. Sinha, P. P. Dey, M. N. Amin, G. W. Romney, Database modeling with Object Relationship Schema. In Proceedings of International Conference on Information Technology Based Higher Education and Training (ITHET), Antalya, IEEE, October 2013, pp. 1-7.

[6] A. Pipitone, R. Pirrone, A Hidden Markov Model for Automatic Generation of ER Diagrams from OWL Ontology. In Proceedings of IEEE International Conference on Semantic Computing (ICSC), Newport Beach, CA, June 2014, pp. 135-142.

[7] D. Ungureanu, A New Definition for The Information Content of Databases. *U.P.B. Scientific Bulletin*, Series C, Vol.74, Iss.3, 2012, pp. 35-44.

[8] K. Moss, The Entity-Relationship Model. In Proceedings of Global Engineering Education Conference (EDUCON), Marrakech, IEEE, April 2012, pp. 1-6.

[9] K. Xu, J. Feng, M. Crowe, Defining the Notion of 'Information Content' and Reasoning about it in a Database. *Knowledge and Information Systems*, 18(1), 2009, pp. 29-59.

[10] D. Howe, Data Analysis for Database Design. 2nd ed., London: Edward Arnold, 1989.

[11] J. Feng, M. Crowe, The Notion of "Classes of a Path" in ER schemata. *In Proceedings of Third East European Conference on Advances in Databases and Information Systems*, ADBIS'99, LNCS 1691, Springer, 1999, pp. 218-231.

[12] F.I. Dretske, *Knowledge and the Flow of Information*. Cambridge University Press, 1999.

[13] T. M. Connolly, C. E. Begg, *Database Systems: a practical approach to design, implementation, and management*. 5th ed., 2010.