Less is More: How to Tame a Very Large ER Diagram [1]

Yannis Tzitzikas[2] and Jean-Luc Hainaut[3]

[2] *University of Crete and FORTH-ICS, Heraklion, Greece*

[3] *Institut d'Informatique, University of Namur (F.U.N.D.P.), Belgium*

email: tzitzik@csi.forth.gr, jlh@info.fundp.ac.be

Abstract

Understanding a large schema without the assistance of persons already familiar with it (and its associated applications), is a hard and very time consuming task that occurs very frequently in reverse engineering and in information integration. In this paper we describe a novel method that can aid the understanding and the visualization of very large ER diagrams that is inspired by the link analysis techniques that are used in Web Searching. Specifically, this method takes as input an ER diagram and returns a smaller (top-k) diagram that consists of the major entity and relationship types of the initial diagram. Concerning the drawing in the 2D space of the resulting top-k graphs, we propose a force-directed placement algorithm especially adapted for ER diagrams. Specifically, we describe and analyze experimentally two different force models and various configurations. The experimental evaluation on large diagrams of real world applications proved the effectiveness of this technique.

1 Prologue

It has been recognized long ago that the usefulness of conceptual diagrams (e.g. ER/UML diagrams) degrades rapidly as they grow in size. Understanding a large schema without the assistance of persons already familiar with it, is often a nightmare. Unfortunately, large conceptual schemas are becoming more and more frequent. The integration of information systems, the development or reverse engineering of large systems, the usage of ERP (the SAP database includes 30.000 tables) and the development of the Semantic Web (structured into ontologies potentially including dozens of thousands of classes) naturally lead to the building of very large schemas. Although a good drawing of a conceptual schema could aid its understanding, and several approaches for automatic placement have been already proposed (e.g. see [33, 23, 8, 28, 10]), it is a widely accepted opinion that the automatic layout facilities offered by current UML-based CASE tools are not satisfactory even for very small diagrams (for more see [14])[2]. Consequently, the vast majority of layouts created today are done "by hand"; a human designer makes

[1] Extended version of paper accepted to be presented and published at ER'2005 [35] (24th Intern. Conf. on Conceptual Modeling, Klagenfurt, Austria, Oct. 2005) .

[2] General graph drawings algorithms (e.g. see [2]) usually make some assumptions that are not always valid in conceptual graphs.

most, if not all, of the decisions about the position of the objects to be presented [26]. The visualization and drawing of large conceptual graphs is even less explored. The classical hierarchical decomposition techniques that are used for visualizing large plain graphs (for a survey see Chapter 3 of [29]), have not been applied or tested on conceptual graphs. Consequently, only manual collapsing mechanisms (like those described in [22]) are currently available for decreasing the visual clutter and for aiding the understanding of big conceptual graphs. In addition, the techniques that have have been proposed for reducing the size of a conceptual graph (in order to aid its comprehension), specifically ER clustering, either require human input [15, 34, 18, 7], or they are automated but not tested on large conceptual schemas [30, 1].

We decided to devise an automatic technique for identifying the major entity and relationship types of a very large conceptual graph as a means for facilitating its understanding. As Link Analysis has been proved very successful in Web Searching [6, 25] and recently in several other application domains [19, 20], we decided to design a similar in spirit technique for one very common and important kind of conceptual schemas, namely Entity-Relationship (ER) diagrams [9]. Concerning the drawing in the 2D space of the resulting top-k graphs, we describe a force-directed placement algorithm especially adapted for ER diagrams. Specifically, we describe and analyze experimentally two different force models and various configurations.

Both of the techniques that are presented in this paper can be applied not only to ER diagrams, but also on other kinds of conceptual graphs. The Semantic Web is one interesting application area because it is founded on ontologies (potentially including dozens of thousands of classes) that are exchanged in a layout-missing format. In this context, the provision of top-k diagrams and automatic layout services is very important as they can aid understanding that is very important for accomplishing tasks like semantic annotation, creation of ontology mappings, ontology specialization, etc.

This paper is structured as follows: Section 2 describes Link Analysis for ER diagrams, Section 3 introduces force-directed placement algorithms for ER diagrams and finally, Section 4 concludes this paper.

2 Link Analysis for ER Diagrams

Our objective here is to identify the major entity and relationship types of a very large ER diagram in order to facilitate its understanding. We designed a PageRank [6] style scoring method because PageRank is described in terms on the entire Web, while HITS [25] is mainly applied on small collections of pages (say those retrieved in response to a query). We view an ER diagram as a triple (E, R, I) where $E = \{e_1, ..., e_N\}$ denotes the entity types, $R = \{r_1, ..., r_m\}$ denotes the relationship types, and I the isA relationships over E (i.e. $I \subseteq E \times E$). For any given $e \in E$, we shall use $conn_R(e)$ to denote those entity types that are connected with e through relationship types, $conn_{sb}(e)$ denote the direct subtypes of e, and $conn_{sp}(e)$ the direct supertypes of e. We shall also use the following shorthands: $conn_I(e) = conn_{sb}(e) \cup conn_{sp}(e)$ and $conn(e) = conn_R(e) \cup conn_I(e)$. Since two entity types may be connected with more than one relationship types we consider $conn_R(e)$ as a *bag* for being able to record duplicates. In addition, we shall use $attrs(e)$ to denote the attributes of an entity type e.

Now the *score* (or EntityRank) of an entity type e in E, denoted $Sc(e)$, can be defined as follows:

$$Sc(e) = q/N + (1-q) * \sum_{e' \in conn_R(e)} \frac{Sc(e')}{|conn_R(e')|} \qquad (1)$$

where q stands for a constant less than 1 (e.g. 0.15 as in the case of Google)[3]. One can easily see that the above formula simulates a random walk in the schema. Under this view each relationship type is viewed as a *bidirectional* transition and the probability of randomly jumping to an entity type is the same for all entity types (i.e. q/N). The resulting scores of the entity types correspond to the stationary probabilities of the Markov chain.

A rising question here is how we can incorporate n-ary ($n > 2$) relationship types into the aforementioned model. This can be achieved by replacing each n-ary relationship type ($n > 2$) over n entity types $e_1, ..., e_n$ by $n(n-1)/2$ binary relationship types that form a complete graph[4] over $e_1, ..., e_n$. Consequently, an n-ary relationship type is viewed as $n(n-1)/2$ binary relationships.

An alternative approach is to assume that the probability of jumping to a random entity is not the same for all entities, but it depends on the number of its attributes. In this case we can define the *score* (or BEntityRank) of an entity type e in E as follows:

$$Sc(e) = q\frac{|attrs(e)|}{|Attr|} + (1-q) \cdot \sum_{e' \in conn_R(e)} \frac{Sc(e')}{|conn_R(e')|} \qquad (2)$$

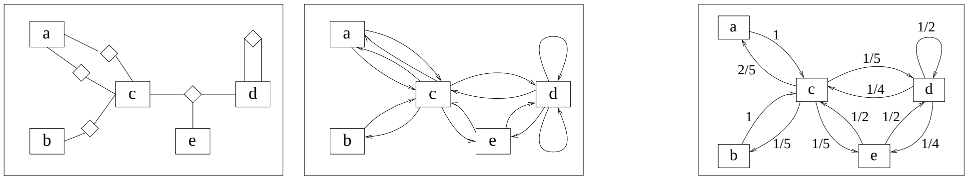
where $Attr$ denotes the set of all attributes of all entity types (i.e. $Attr = \cup\{\ attrs(e) \mid e \in E\}$). This particular formula simulates a user navigating randomly in the schema who jumps to a random entity e with probability $\frac{q|attrs(e)|}{|Attr|}$ or follows a random relationship type (on the current entity). The probability $\frac{|attrs(e)|}{|Attr|}$ corresponds to the probability of selecting e by clicking randomly on a list that enumerates the attributes of all entity types of the schema.

The linear algebra version of EntityRank and BEntityRank is given in Appendix A.

Let's now discuss the differences between link analysis for ER diagrams and link analysis for the Web. Firstly, Web links are directed, while relationship types are not directed thus the latter are considered as bidirectional transitions. Secondly, we do not collapse all relationships types between two entity types into one (as it is done with Web links), and this is the reason why we consider $conn_R$ as a bag. Thirdly, in ER diagrams we should count "self hyperlinks" (i.e. cyclic relationship types), although the Web techniques ignore them. For example, consider a schema consisting of two entity types $\{\texttt{Person}, \texttt{City}\}$

[3]If we set $q = 0.15$ or below then an iterative method for computing the scores (e.g. the Jacobi method) requires at most 100 iterations to convergence.

[4]A complete graph is a graph in which each pair of graph vertices is connected by an edge.



(a) (b)

Figure 1: An ER diagram and the corresponding transitions

and two relationship types {`Person` *lives* `City`, `Person` $fatherOf$ `Person`}. If we ignore the relationship type $fatherOf$, then both `Person` and `City` would be equally scored, a not so good choice. At last, although in the Web link analysis is exploited mainly for ranking the results of retrieval queries, in our case we don't need just a ranked list of entity types, but rather another *diagram* that consists of the major entity and relationship types.

As example, Figure 1 shows an ER diagram, the corresponding states and transitions and finally, the transition probabilities (the probabilities of selecting a link) if we ignore the "teleporting" transitions.

We have implemented and evaluated the above scoring schemes into the $\texttt{DB} - \texttt{MAIN}$ CASE tool (for more see [17, 21]). The designer provides a threshold *per* between 0 and 100. Subsequently, all entity types with score lower than $per\% * ScMax$, where $ScMax$ denotes the highest score, disappear. Controlling the visibility of entity types according to their score, and not according to their rank, is preferred as it better handles ties. Concerning relationship types, only those that connect the visible entity types are displayed. The computation of the scores takes only some seconds on a conventional PC. Specifically, to compute the scores we use the Jacobi iterative algorithm. We have noticed that 50 iterations give quite stable orderings and their application on schemas with 1000 entity and relationship types takes less than 2 seconds in a conventional PC.

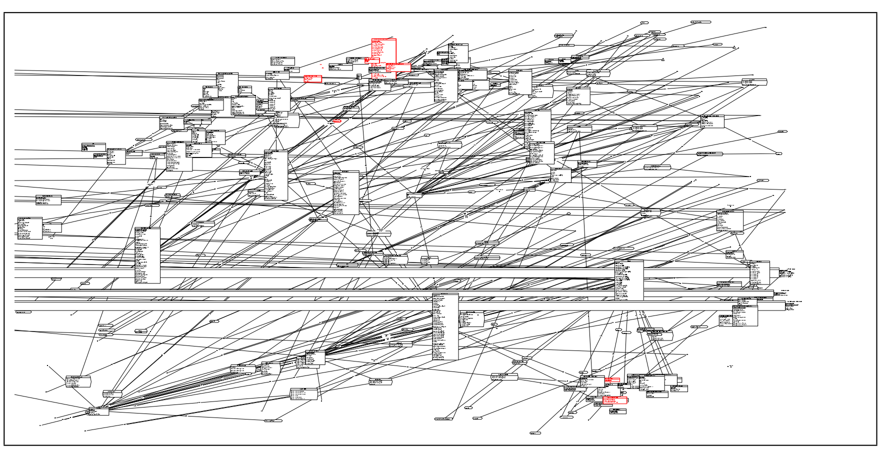


Figure 2: Excerpt (< 10%) of a large ER diagram drawn using a force-directed placement algorithm

Figure 1 shows a very small part of the ER diagram of a Belgian distribution company. Though the schema comprises about 450 nodes and 800 edges only, the layout is definitely useless for understanding the schema and the corresponding application.

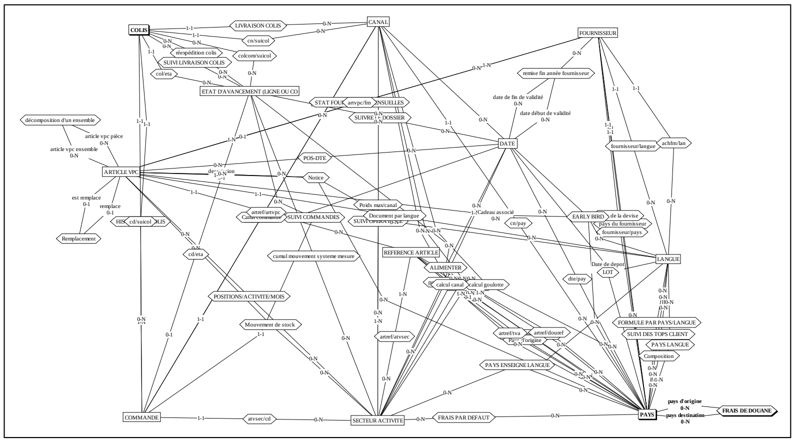


Figure 3: The diagram of the top-11 entity types of the schema of Figure 1

Now Figure 2 shows in micrography the diagram of the top-11 entity types of the schema of Figure 1 according to EntityRank (for reasons of space, the attributes are not displayed in this figure). Although this schema has 54 relationship types it is extremely more easy to visualize, and thus to understand, than the original schema. Of course, one user could start from even smaller diagrams. For instance, Figure 3 shows the graph of the top-5 entity types of the same schema. It indeed contains the major entity types of this application and a user can immediately understand the application domain of this schema.

In case of diagrams with big *isA* hierarchies, some entity types, although major, may not receive high scores because their relationship types are scattered in several subentity types. To handle this case, we introduced a (optional) preprocessing step in which each *isA* hierarchy of the schema is collapsed into one entity type that collects all the attributes and relationship types of its subentity types.

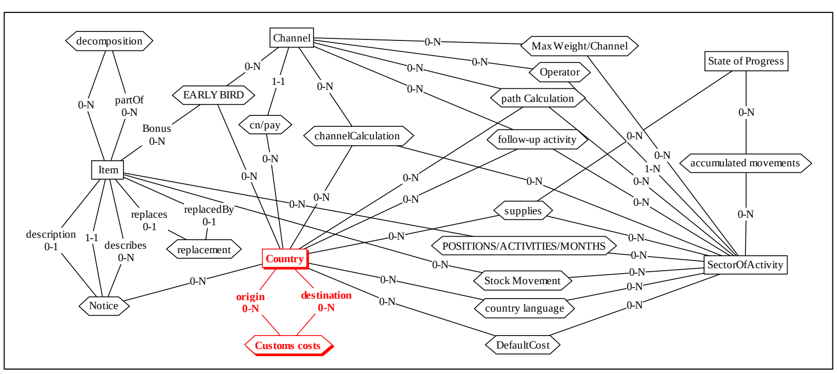
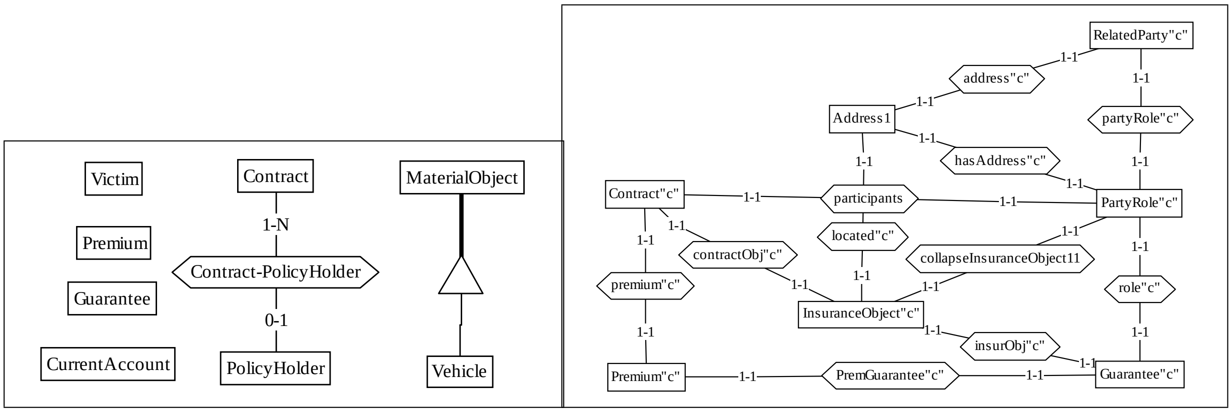


Figure 4: The diagram of the top-5 entity types of the schema of Figure 1

As another example, TELEBIB2 is a national standard (for Belgium) for exchanging messages between insurance companies. Its schema[5] consists of 339 entity types and 232 relationship types. Figure 5.(a)

[5] which can be found in http://www.telebib2.org/test/DataModel.htm



(a) (b)

Figure 5: (a) shows the top-8 diagram of the schema of TELEBIB2, while (b) shows the top-7 diagram of the same schema but after collapsing *isA* hierarchies.

shows the diagram that contains the top-8 diagram of TELEBIB2 according to EntityRank. Notice that only one glance to this diagram is enough in order to understand the domain of this application (here the triangle-marked line segment denotes an *isA* link). Now, Figure 5.(b) shows the top-7 diagram of the same schema after first collapsing each *isA* hierarchy into one entity type (this explains the suffix "c" of the relationship types). In contrast to the diagram of Figure 5.(a), this diagram is nicely connected. The two diagrams have 3 entity types in common: `Guarantee`, `Premium` and `Contract` (the suffix "c" of the names in the second diagram indicates that collapsing has taken place). The entity types `MaterialObject` and `Vehicle` of the first diagram have been replaced in the second by the entity type `InsuranceObject` which is the top element of the corresponding *isA* hierarchy.

Concerning evaluation, at first we have to note that the evaluation of the effectiveness of link analysis techniques for ER diagrams (for conceptual graphs in general), is more difficult than in the case of Web. In the latter case, it is not so hard to judge whether the top ranked pages are indeed relevant to the submitted query. However, in the case of conceptual graphs, one has to know well in advance the conceptual graph in order to judge whether the resulting small graph indeed contains the major concepts of the conceptual graph and of its underlying domain. Inevitably, the most reliable evaluation of such techniques can be done only in already known conceptual graphs. For this reason we applied this method to almost every conceptual schema that the DBMAIN group has produced the last 3 years. This was a quite representative test bed as it includes big schemas of existing (and non artificial) real world applications. We always obtained surprisingly good results. For reasons of space we cannot report here the exact results of the evaluation of EntityRank (and BEntityRank) using specific metrics coming from the area of IR (for more [36]). In addition, it is an advantage that the proposed formulas for link analysis are mathematically founded and that the underlying model (the random walk model) is quite relevant to *browsing*, i.e. to the most widely used method for understanding a conceptual graph. At last, another evidence that link analysis is indeed appropriate for ER diagrams is that all large schemas that we have tested have a small set of elements, usually less than 5% of the total ones, whose scores are significantly higher than the rest. This at least indicates that big ER diagrams tend to have a well connected kernel which, at least in our experiments, always comprised the more important concepts of the application domain.

3 Automatic ER Drawing

For drawing automatically the top-k ER diagrams that are derived by the previous technique, we shall view them as mechanical systems. Below we present two force models that combine the *spring-model* (proposed and developed in [13, 24, 16]) with the *magnetic-spring model* (proposed in [32, 31]) in a way that is appropriate for ER diagrams.

3.1 Force Model A

Here entity types are viewed as equally charged particles which *repel* each other. Relationship types and *isA* relationships are viewed as springs that *pull* their adjacent entity types. Moreover, we assume that the springs that correspond to *isA* links are all magnetized and that there is a global magnetic field that acts on these springs. Specifically, this magnetic field is parallel (i.e. all magnetic forces operate in the same direction) and the *isA* springs are magnetized unidirectionally, so they tend to align with the direction of the magnetic field, here upwards. Figure 6 illustrates this metaphor.

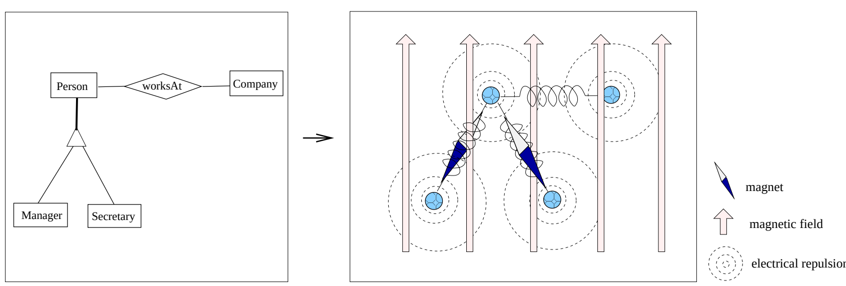


Figure 6: Viewing an ER diagram as a mechanical system

Under the above force model, the force on a entity type e_i is given by:

$$F(e_i) = \sum_{e_j \in conn(e_i)} f(e_j, e_i) + \sum_{e_j \in E, e_j \neq e_i} g(e_j, e_i) + \sum_{e_j \in conn_I(e_i)} h(e_j, e_i) \quad (3)$$

where: $f(e_j, e_i)$ is the force exerted on e_i by the spring between e_j and e_i (note that e_i and e_j are connected by a relationship type or an *isA* link), $g(e_j, e_i)$ is the electrical repulsion exerted on e_i by the entity type e_j, and $h(e_j, e_i)$ is the rotational force exerted on e_i by the entity type e_j (here e_i and e_j are connected by an *isA* link).

Figure 7 gives some indicative examples that explain the role of the forces f, g and h. Specifically, figure (a) justifies the spring force, figure (b) justifies the electrical repulsion and shows that high electrical repulsion (high K^e) results in symmetrical drawings, and figure (c) illustrates how the magnetic field can be used in order to obtain the classical top-down drawings for isA hierarchies.

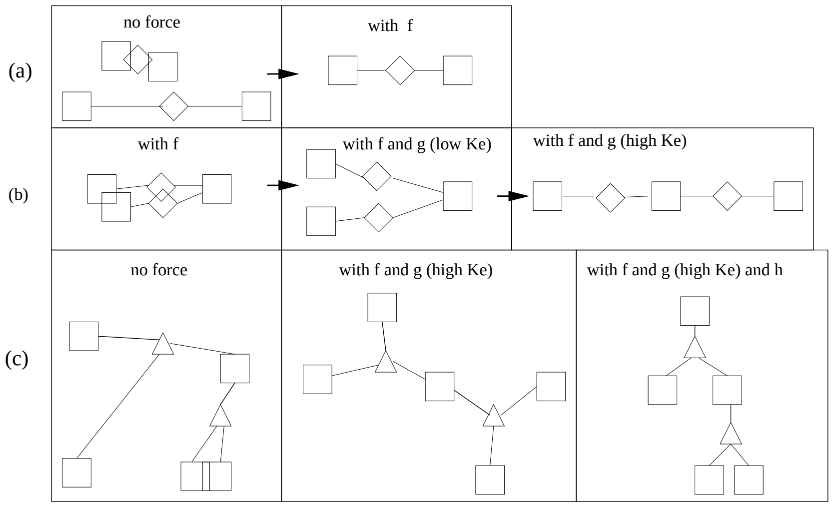


Figure 7: Forces and ER Drawings

The spring force $f(e_j, e_i)$ follows Hooke's law, i.e. it is proportional to the difference between the distance between e_j and e_i and the zero-energy length of the spring. Let $d(p, p')$ denote the Euclidean distance between two points p and p' and let $p_i = (x_i, y_i)$ denote the position of an entity type e_i. The x component of the force $f(e_i)$ is given by:

$$f_x(e_i) \quad = \quad \sum_{e_j \in conn(e_i)} K^s_{i,j}(d(p_i, p_j) - L_{i,j}) \frac{x_j - x_i}{d(p_i, p_j)}$$

where $L_{i,j}$ denotes the natural (zero energy) *length* of the spring between e_i and e_j. This means that if $d(p_i, p_j) = L_{i,j}$ then no force is exerted by the spring between e_i and e_j. Now $K^s_{i,j}$ denotes the *stiffness* of the spring between e_i and e_j. The larger the value of $K^s_{i,j}$, the more tendency for the distance $d(p_i, p_j)$ to be close to $L_{i,j}$. The y component of the force $f(e_i)$ is defined analogously.

The electrical force $g(e_j, e_i)$ follows an inverse square law. The x component of the force $g(e_i)$ is given by:

$$g_x(e_i) \quad = \quad \sum_{e_j \in E, e_j \neq e_i} \frac{K^e_{i,j}}{d(p_i, p_j)^2} \frac{x_i - x_j}{d(p_i, p_j)}$$

where $K^e_{i,j}$ is used to control the *repulsion strength* between e_i and e_j. The y component of the force $g(e_i)$ is defined analogously.

The magnetic force $h(e_j, e_i)$ depends on the angle between the isA spring (that connects e_j and e_i) and the direction of the magnetic field and it induces a rotational force on that spring. For example, Figure 8 shows an isA link between e_i and e_j and the exerted forces on e_i and e_j due to the magnetic field. The x and y components of the magnetic force $h(e_i)$ are given by:

$$h_x(e_i) \quad = \quad \sum_{e_j \in conn_{sp}(e_i)} K^m \frac{x_j - x_i}{L_{i,j}} + \sum_{e_j \in conn_{sb}(e_i)} K^m \frac{x_j - x_i}{L_{i,j}}$$

$$h_y(e_i) \quad = \sum_{e_j \in conn_{sp}(e_i)} K^m \frac{L_{i,j} + y_j - y_i}{L_{i,j}} - \sum_{e_j \in conn_{sb}(e_i)} K^m \frac{L_{i,j} + y_i - y_j}{L_{i,j}}$$

where K^m is used to control the strength of the magnetic field.

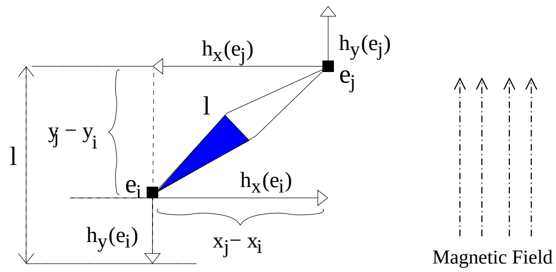


Figure 8: Magnetic forces and isA links

The x and y components of the composed force $F(e_i)$ on an entity type e_i are obtained by summing up, i.e.: $F_x(e_i) = f_x(e_i) + g_x(e_i) + h_x(e_i)$ and $F_y(e_i) = f_y(e_i) + g_y(e_i) + h_y(e_i)$.

As in the link analysis technique, we view an n-ary relationship type as $n(n-1)/2$ springs.

3.2 Force Model B

One weakness of the above model is that the resulting drawings can have several overlaps. The reason is that: (a) there is no repulsion among relationship types, and (b) there is no repulsion between entity and relationship types. Figure 9 illustrates this problem. This drove us to introduce a different force model where each relationship type is viewed as a particle too. Clearly, the resulting electrical repulsion discourages the creation of overlaps (between entity and relationship types, or between relationship types themselves). Notice that according to this view, a relationship type does no longer correspond to one spring. Specifically, the particle of a relationship type over k entity types, is connected with one spring with each one of them. The forces on entity types and relationship types are computed analogously to the force model A.

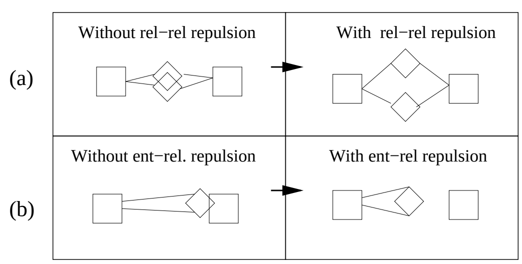


Figure 9: Forces and ER Drawings

3.3 The Drawing Algorithm

We can reach a drawing by an algorithm that simulates the mechanical system. Such a algorithm would seek for a configuration with locally minimal energy, i.e. a drawing in which the forces on each node is zero. A variety of numerical techniques can be used to find an equilibrium configuration, and thus the final drawing. We have adopted the iterative method based on the method proposed in [13]. At first the nodes are placed at random positions. At each iteration, the force on each node is computed and then the node is moved towards the corresponding direction by a small amount proportional to the magnitude of the force. This can be continued until convergence, but we can also limit the number of iterations.

Note that if we would like to find a drawing that corresponds to a state with globally minimal energy, then we would have to resort to very general optimization methods. For instance, a method based on simulated annealing is proposed in [11], while an approach based on genetic algorithms is described in [4]. However the computational complexity of these techniques turns them not very appropriate for interactive design systems. In addition, and according to the results of the extensive empirical analysis of several force-directed algorithms (including globally minimal energy algorithms) upon plain graphs that are reported in [3], there is no universal winner and the general approach is to try several methods and choose the best.

3.4 Experimental Evaluation

We have investigated and evaluated all these issues in the context of the CASE tool DB-MAIN. The specification of the parameters L, K^s, K^e and K^m is not a trivial task as these parameters determine in a high degree how the final drawing will look like. One flexibility of the proposed approach is that we can adjust the spring length ($L_{i,j}$), spring stiffness($K^s_{i,j}$) and electrical repulsion($K^e_{i,j}$), in order to customize the appearance of the drawing according to the semantics of the ER diagram constructs. For instance, as it is desirable to keep the nodes of an *isA* hierarchy close enough and since between any two isA-related entity types we only have to draw a line (and not any hexagon-enclosed string), we can use a smaller length for *isA*-springs than that of relationship-springs. In any case, the user can change their value at run-time.

Figure 10 shows one drawing obtained by the algorithm using low electrical repulsion. Although the *isA* hierarchy is drawn as a top-down drawing and we have no overlaps, this drawing is not satisfying because a designer would hardly manually place into the space occupied by an *isA* hierarchy an entity type that does not belong to that hierarchy. After we increased the repulsion and the magnetic field we never faced again such a drawing. The lesson learned is that high repulsion not only results in symmetrical drawings but its combination with a strong magnetic field results in clear *isA* drawings. Another drawing of a diagram with 4 *isA* hierarchies that is derived by the algorithm according to force model A, is shown in Figure 11.

A more complex case is shown in Figure 12. Figure 12.(a) shows a manually placed diagram where all subentity types have been placed at the outer part of the drawing. Figure (b) shows the drawing obtained according to force model B. Notice that every *isA* hierarchy now corresponds to a top-down drawing and that the entire drawing is symmetrical and satisfying.

The experimental evaluation showed that the drawings according to force model A suffer from overlaps,

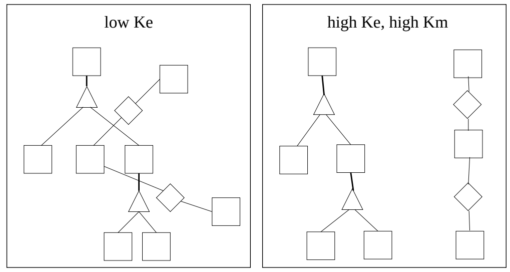


Figure 10: How to obtain clean isA drawings

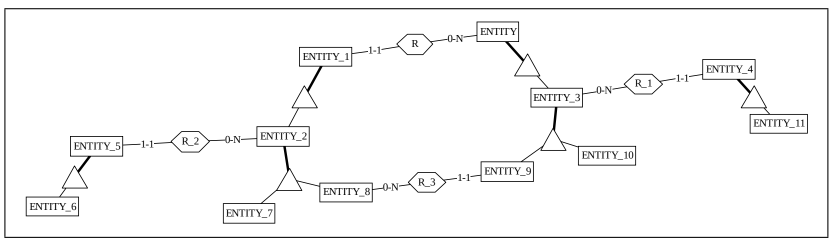
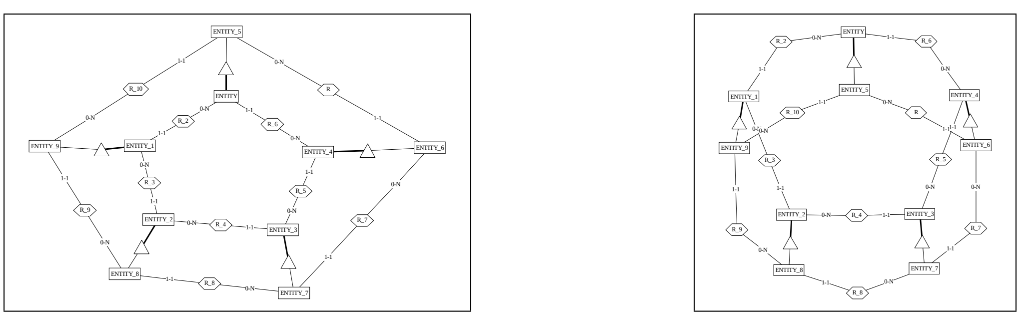


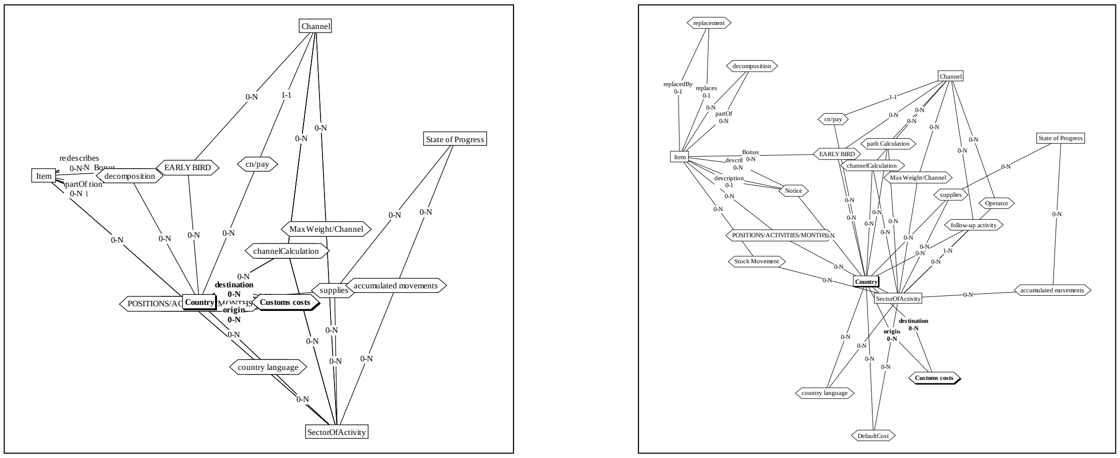
Figure 11: A drawing of a diagram with 4 isA hierarchies according to force model A



(a) (b)

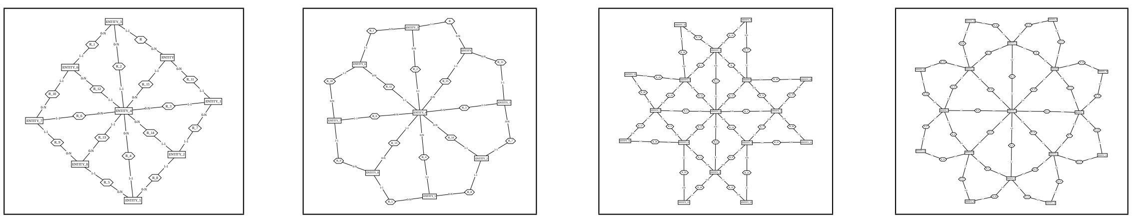
Figure 12: Drawing of a diagram with several IsA hierarchies

(a): manual drawing where isA links are not vertical. (b): drawing obtained according to force model B



(a) (b)

Figure 13: Force model A vs force model B on a diagram with $|E| = 5$ and $|R| = 17$



(a) (b) (c) (d)

Figure 14: Force model A vs force model B

(a): force model A. (b): force model B. (c): force model A. (d): force model B.

while those according to force model B have a few (or none) overlaps. The difference between force model A and force model B is even more evident in dense diagrams. Figure 13 shows the drawings obtained by these two models when applied on the top-5 ($|E| = 5, |R| = 17$) diagram of Figure 3. Again, the second drawing is evidently better. A noteworthy remark here is that the second diagram is more clear and intuitive than the manually specified layout that is shown in Figure 3. This indicates that in certain cases (at least when the diagram is very dense) the automatically-derived drawings can be better than the manually drawn.

However, we have to note that force model B has two weaknesses comparing to force model A: (i) it is computational more expensive, and (ii) in the resulting drawings the tentacles of binary relationship types are in many cases unnecessarily not aligned. This is evident in Figure 14. Although this is not a major problem it is an issue for further research.

Figure 15 shows the automatic layout obtained for the top-11 diagram (that was presented in Figure 2). The high relative number of relationships makes the drawing almost unreadable. This example suggests that we should take into account the density of a diagram, in order to reach readable and clear drawings.

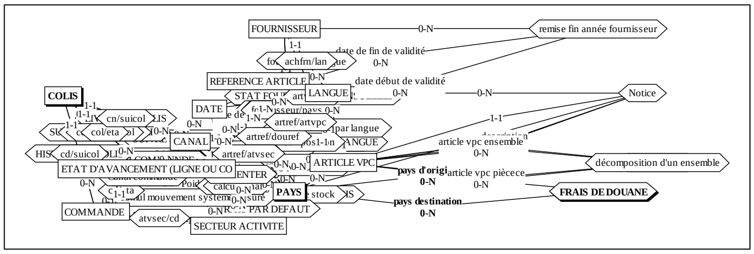
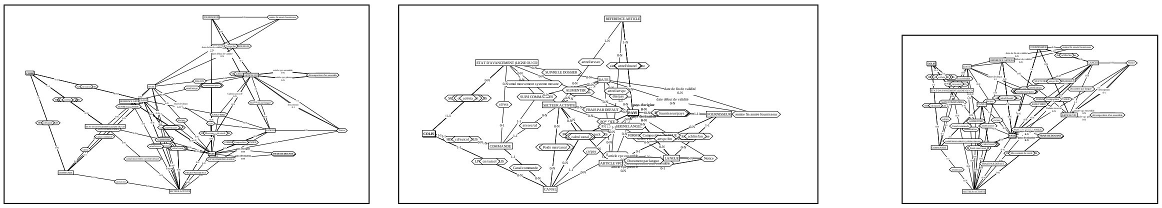


Figure 15: Dense diagram drawing



(a) (b) (c)

Figure 16: (a): larger springs; (b): higher repulsion; (c) less stiff springs

Roughly, we could handle dense diagrams by considering: (i) larger springs, (ii) higher electrical repulsion, (iii) less stiff springs. For example, and assuming force model A, Figure 16.(a) shows the drawing obtained with spring length $L' = 5L$, Figure 16.(b) shows the drawing obtained with $K^{e'} = 100K^e$, and Figure 16.(c) shows the drawing obtained with $K^{s'} = K^s/10000$. Indeed, all are better than the original drawing shown in Figure 15. Another simple method that is both effective and efficient is to scale up the entire drawing (i.e. multiply each coordinate by a constant $c > 1$). Nevertheless, an issue that is worth further research is to investigate the effectiveness of local-density adaptations, e.g. to adapt the spring lengths according to the local density of the graph. EntityRank and BEntityRank scores could be exploited for this purpose.

As a final remark note that the above drawing techniques can be applied for drawing the structural part of ontologies expressed in RDFS [5] and OWL [12]. The only difference is that RDFS supports property specialization which however will be handled correctly due to the magnetic field that is applied on specialization/generalization links (also indicated by Figure 11).

4 Conclusion

We described a novel method for identifying the major elements of an ER diagram that is based on link analysis. This method can significantly aid (a) the *understanding*, (b) the *visualization*, and (c) the *drawing* of very large schemas. The proposed technique can elevate automatically the major elements and allows exploring the schema gradually: from the more important elements to the less. Consequently, it can be very useful in reverse engineering and in information integration. Moreover, the scores can be exploited for ordering the schema elements that match a keyword query of the user. In addition, and given the inability to produce automatically aesthetically satisfying layouts for large schemas, the small (top-k graphs) that can be derived by this technique can be visualized effectively and this is very useful during communication (e.g. between designers and application programmers or in requirements engineering and training). For this purpose we investigated a force-directed drawing algorithm and evaluated two different force models upon several conceptual schemas of real applications. For small and medium sized diagrams the results were satisfying in most of the cases. In the rest cases, human intervention (moving, nailing) and rerun of the drawing algorithm could rectify the problems.

References

[1] Jacky Akoka and Isabelle Comyn-Wattiau. "Entity-Relationship and Object-Oriented Model Automatic Clustering". *Data and Knowledge Engineering*, 20(2):87–117, 1996.

[2] Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis Tollis. *"Graph drawing: algorithms for the visualization of graphs"*. Prentice Hall Englewood Cliffs (N.J.), 1999. ISBN/ISSN : 0-13-301615-3.

[3] F. J. Braedenburg, M. Himsolt, and C. Rohrer. "An Experimental Comparison of Force-Direceted and Randomized Graph Drawing Algorithms". In *Procs of Graph Drawing, GD'95*, pages 76–87, 1996.

[4] J. Branke, F. Bucher, and H. Schmeck. "Using Genetic Algorithms for Drawing Undirected Graphs". In *Procs of the 3rd Nordic Workshop on Genetic Algorithms and Their Applications, 3NWGA*, pages 193–2005, 1997.

[5] Dan Brickley and R. V. Guha. "Resource Description Framework (RDF) Schema specification: Proposed Recommendation, W3C", March 1999. http://www.w3.org/TR/1999/PR-rdf-schema-19990303.

[6] Sergey Brin and Lawrence Page. "The Anatomy of a Large-scale Hypertextual Web Search Engine". In *Proceedings of the 7th International WWW Conference*, Brisbane, Australia, April 1998.

[7] L. J. Campbell, Terry A. Halpin, and Henderik Alex Proper. "Conceptual Schemas with Abstractions: Making Flat Conceptual Schemas More Comprehensible". *Data and Knowledge Engineering*, 20(1):39–85, 1996.

[8] Rodolfo Castello, Rym Mili, and I. Tollis. "A Framework for the Static and Interactive Visualization of Statecharts". *Journal of Graph Algorithms and Applications*, 6(3):313–351, 2002.

[9] P. Chen. "The Entity-Relationship Model - Toward a Unified View of Data". *ACM Transactions on Database Systems*, 1(1):9–36, March 1976.

[10] Richard Cole. "Automatic Layout of Concept Lattices using Force Directed Placement and Genetic Algorithms". In *Proc. of the 23th Australiasian Computer Science Conference*, pages 47–53. Australian Computer Science Communications 1, IEEE Computer Society, 2000.

[11] R. Davidson and D. Harel. "Drawing Graphics Nicely Using Simulated Annealing". *ACM Trans. Graph.*, 15, 1996.

[12] M. Dean, D. Connolly, F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, and L.A. Stein. "OWL Web Ontology Language 1.0 Reference", 2002. (http://www.w3c.org/TR/owl-ref).

[13] P. Eades. "A Heuristic for Graph Drawing". *Congressus Numerantium*, 42, 1984.

[14] Holger Eichelberger and Jurgen Wolff von Gudenberg. "UML Class Diagrams - State of the Art in Layout Techniques". In *Proceeding of Vissoft 2003, International Workshop on Visualizing Software for Understanding and Analysis*, pages 30–34, 2003.

[15] P. Feldman and D. Miller. "Entity Model Clustering: Structuring a Data Model by Abstraction". *The Computer Journal*, 29(4):348–360, 1986.

[16] T. Fruchterman and E. Reingold. "Graph Drawing by Force-directed Placement". *Software - Practice and Experience*, 21(11):1129–1164, 1991.

[17] F.U.N.D.P. "DB-MAIN". (http://www.info.fundp.ac.be/∼dbm/).

[18] Munish Gandhi, EdwardL Robertson, and Dirk Van Gucht. "Levelled Entity Relationship Model". In *Procs of the 13rd Intern. Conf. on the Entity Relationship Approach, ER'94*, pages 420–436, Manchester, U.K., December 1994.

[19] Floris Geerts, Heikki Mannila, and Evimaria Terzi. "Relational Link-based ranking". In *Procs of the 30th Intern. Conference on Verly Large Data Bases, VLDB'2004*, Toronto, Canada, August 2004.

[20] Zoltan Gyongyi, Hector Garcia-Molina, and Jan Pedersen. "Combating Web Spam with TrustRank". In *Procs of the 30th Intern. Conference on Verly Large Data Bases, VLDB'2004*, Toronto, Canada, August 2004.

[21] Jean-Luc Hainaut. "Transformation-based Database Engineering". In *Transformation of Knowledge, Information and Data: Theory and Applications*. IDEA Group Pub., 2004.

[22] Jouni Huotari, Kalle Lyytinen, and Marketta Niemela. "Improving Graphical Information System Model Use with Elision and Connecting Lines". *ACM Transactions on Computer-Human Interaction*, 10(4), 2003.

[23] Yannis E. Ioannidis, Miron Livny, Jian Bao, and Eben M. Haber. "User-Oriented Visual Layout at Multiple Granularities". In *Proc. of the 3rd International Workshop on Advanced Visual Interfaces*, pages 184–193, Gubbio, Italy, May 1996.

[24] T. Kamada. *"On Visualization of Abstract Objects and Relations"*. PhD thesis, Dept. of Information Science, Univ. of Tokyo, Dec 1988.

[25] Jon Kleinberg. "Authoritative Sources in a Hyperlinked Environment". In *Proceedings of 9th ACM-SIAM Symposium on Discrete Algorithms*, San Francisco, USA, 1998.

[26] Simon Lok and Steven Feiner. "A Survey of Automated Layout Techniques for Information Presentations". In *Procs of the 1st. Int. Symp. on Smart Graphics*, Hawthorne, NY, 2001.

[27] Rajeev Motwani and Prabhakar Raghavan. *Randomized algorithms*. Cambridge University Press, 1995.

[28] Arthur Ouwerkerk and Heiner Stuckenschmidt. "Visualizing RDF Data for P2P Information Sharing". In *Procs of the workshop on Visualizing Information in Knowledge Engineering, VIKE'03*, Sanibel Island, FL, 2003.

[29] Aaron J. Quigley. *"Large Scale Relational Information Visualization, Clustering, and Abstraction"*. PhD thesis, University of Newcastle, Australia, August 2001. (http://www.it.usyd.edu.au/∼aquigley/thesis/aquigley-thesis-mar-02.pdf).

[30] O. Rauh and E. Stickel. ”Entity Tree Clustering: A Method for Simplifying ER Design”. In *Procs of the 11th Int. Conf. on Entity-Relationship Approach, ER'92.*

[31] K. Sugiyama and K. Misue. “A Simple and Unified Method for Drawing Graphs: Magnetic-Spring Algorithm”. In *Procs of Graph Drawing Conference, GD'94*, pages 364–375, 1994.

[32] K. Sugiyama and K. Misue. “Graph Drawing by Magnetic-Spring Model”. *Journal on Visual Lang. Comput.*, 6(3), 1995.

[33] R. Tamassia, C. Batini, and M. Talamo. “An algorithm for automatic layout of entity-relationship diagrams”. In *Procs of the 3rd International Conference on Entity-relationship approach to software engineering*, pages 421–439. Elsevier North-Holland, Inc., 1983.

[34] T. J. Teory, W. Guangping, D. L. Bolton, and J. A. Koenig. “ER Model Clustering as an Aid for User Communication and Documentation in Database Design”. *Communications of the ACM*, 32(8):975–987, 1989.

[35] Yannis Tzitzikas and Jean-Luc Hainaut. ”How to Tame a Very Large ER Diagram (using Link Analysis and Force-Directed Placement Algorithms)”. In *Proceedings of 24th Int. Conf. on Conceptual Modeling, ER'2005*, Klagenfurt, Austria, October 2005.

[36] Yannis Tzitzikas and Jean-Luc Hainaut. ”Ranking the Elements of Conceptual Diagrams”, 2005. (submitted for publication).

A Linear Algebra Version of (B)EntityRank

Let A be the generalized adjacency matrix of an ER diagram where $A[e_i, e_j]$ equals the number of transitions from e_i to e_j. Now the probability transition matrix M is obtained by normalizing each row of A to sum to 1 (an example of M was shown in Figure 1.(b)). EntityRank is based on a Markov chain on the entity types with transition matrix

$$q \cdot \mathsf{U} + (1-q) \cdot \mathsf{M}$$

where U is the transition matrix of uniform transition probabilities i.e. $\mathsf{U}[e_i, e_j] = 1/N$ for all i, j. The vector of the EntityRank scores, denoted by Sc, is then defined to be the stationary distribution of this Markov chain. Equivalently, Sc is the principal right eigenvector of the transition matrix $(q \cdot \mathsf{U} + (1-q) \cdot \mathsf{M})^T$, since by definition the stationary distribution satisfies $(q \cdot \mathsf{U} + (1-q) \cdot \mathsf{M})^T \mathsf{Sc} = \mathsf{Sc}$. On the other hand, BEntityRank (the *biased* version of EntityRank) is based on the transition matrix:

$$q \cdot \mathsf{B} + (1-q) \cdot \mathsf{M}$$

where $\mathsf{B}[e_j, e_i] = \frac{|attrs(e_i)|}{|Attr|}$ where $Attr$ denotes the set of all attributes of all entity types (i.e. $Attr = \cup\{\ attrs(e) \mid e \in E\}$).

As another remark note that in an undirected (strongly connected and non-bipartite) graph $G = (V, R)$, the stationary probability of a node u is given by $P(u) = \frac{deg(u)}{2|R|}$ where $deg(u)$ is the degree of u [27]. This means that in an undirected graph (or multigraph) the stationary probabilities can be computed very efficiently and without the need of an iterative algorithm. In our case we cannot employ the above method due to the ”teleporting” transitions which are indispensable in our case for ensuring that the transition graph is strongly connected (note that large ER diagrams are not always connected). Specifically, the ”teleporting” transitions of BEntityRank are not symmetric and this cannot be captured by an undirected graph. For instance, consider the case of an ER diagram consisting of two entity types $e1$ and $e2$ and one relationship type between them, where $e1$ has one attribute and $e2$ has two attributes. According to a random walk on the undirected graph both entity types have probability 1/2. According to BEntityRank if $q = 0$ then $P(e_1) = P(e_2) = 1/2$, if $q = 1$ then $P(e_1) = 1/3$ and $P(e_2) = 2/3$, and if $q = 0.5$ then $P(e_1) = 0.44$ and $P(e_2) = 0.55$.