A Model Checking Tool

to detect and resolve potential Redundancies

in Entity Relationship models.

Frédéric Maréchal – 25 March 2008

(Word Count: 1496)

1. ABSTRACT

A fundamental measure of database design success lies in the production of an accurate conceptual model that does not exhibit unnecessary relationships. Researchers have proposed numerous approaches to discover potential redundant relationships in an Entity Relationship Diagram (ERD). However, they reveal restrictions that limit their usage in real life models.

The aims of this research seeks to prototype an enhanced version of a discovery strategy proposed by Bowers (2002), and verify how efficient and correct this solution is in discovering potential redundant relationships. I based the research on the creation of controlled test suites. I ran the prototype against these tests to gather the necessary qualitative and quantitative information that form the foundation of the analysis. The experiment showed the strengths and weaknesses of the discovery ability of the enhanced algorithm, and its execution run time growth rate.

The result of this research suggests that the discovery of potential redundant relationship rate failure is relatively high. However, failures only occur in case of multiple relationships. This is due to a limitation in the current implementation when handling multiple relationships between the first and second entity in an equivalent composed path. This study also demonstrates that the current execution growth rate become exponential when I double the number of entities.

2. REASONS FOR THE RESEARCH

The literature review showed that:

- the functional dependency graph approach (Ausiello, d'Atri and Sacca, 1983) systematically removes longest paths - whether they are potentially redundant or not
- the 'real-world' knowledge discovery strategy success rate is highly correlated to the quality of the relationships name dictionary definition (Storey, 1993 and Azman and Noah, 2000 and Noah and Williams, 2000)
- the "cardinality constraints" (Dullea and Song, 1997), and the "shared attributes and inclusion constraints" (Rosenthal and Reiner, 1994) methodologies are restricted to single relationships in a cycle
- although the ERD adjacency-matrix multiplication strategy (Bowers, 2000 and Bowers, 2002) is limited to single relationships without a loop or a cycle, it seems to be the most promising in systematically discovering potential redundant relationships with more complex ERDs

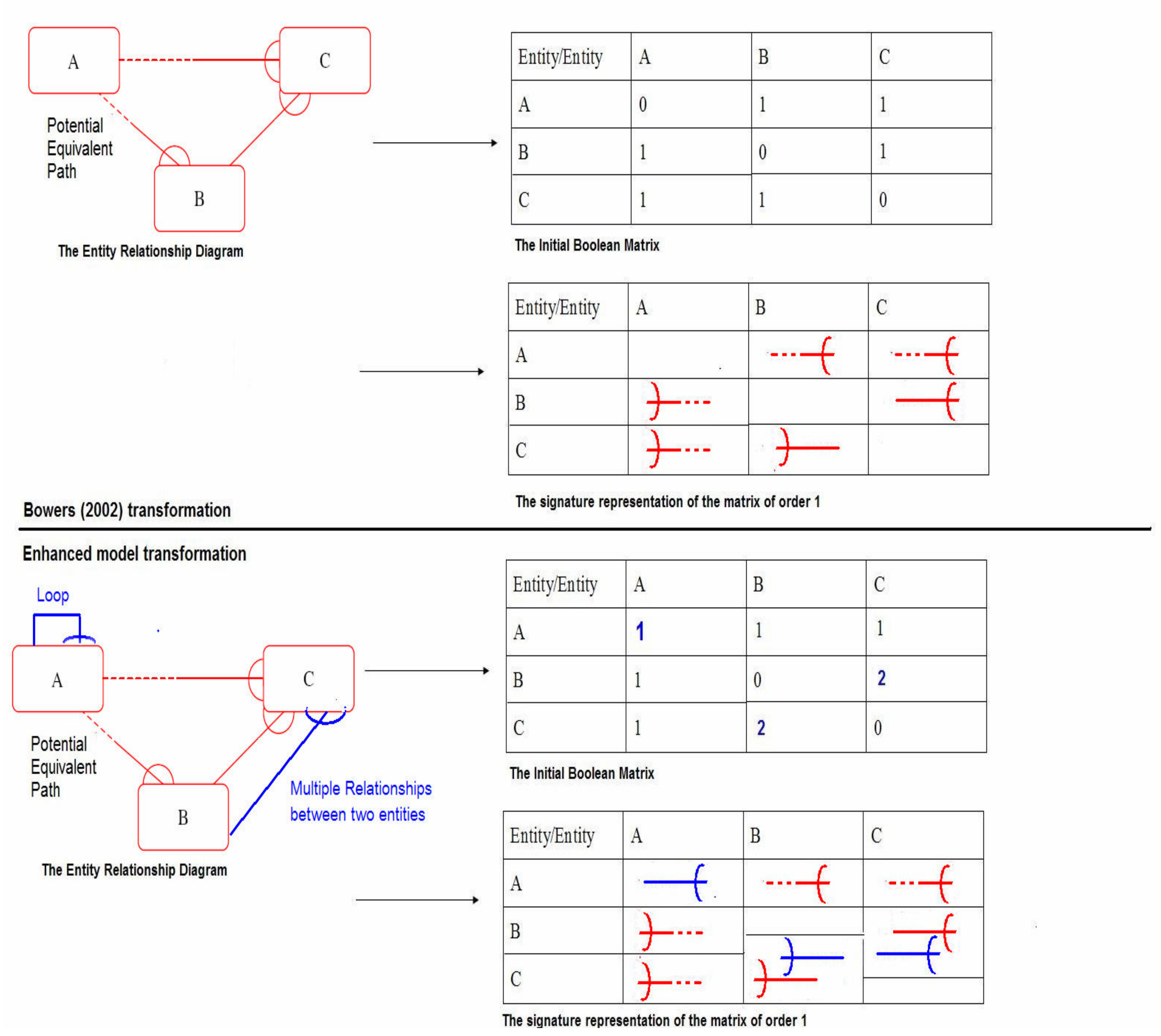
However, none of these methodologies provide an all encompassing solution for discovering potential redundant relationships in ERDs, which contain single/multiple relationships as well as loops and cycles.

This research sought to establish whether:

- the solution proposed by Bowers (2002) can be enhanced to support loops, cycles and multiple relationships,
- this new solution has a higher rate of success in discovering potential redundant relationships for complex ERDs, in comparison to the above mentioned methodologies
- the proposed solution has a 'relatively' efficient execution time, especially for sparse schemas

3. RESEARCH METHOD

I started my research by extending the Bowers (2002) methodology. Bowers (2002) presented a general algorithm, using Boolean logic, which enabled the composition of transitive relationships in an ERD. In a nutshell, the graph entities and relationships are stored in a matrix. The entities are listed on both the x- and y-axis of the matrix, while the relationships signature (i.e. cardinality and participation) are represented in the elements of the matrix, as shown in Figure 1. The proposed enhanced version of the Bowers (2002) model does support diagonal elements (i.e. loops) and elements containing multiple relationships.


A
C
B
Potential Equivalent Path
The Entity Relationship Diagram
The Initial Boolean Matrix
The signature representation of the matrix of order 1
Bowers (2002) transformation
Enhanced model transformation
Loop
Multiple Relationships between two entities

Entity/Entity	A	B	C
A	0	1	1
B	1	0	1
C	1	1	0

The Initial Boolean Matrix

Entity/Entity	A	B	C
A	1	1	1
B	1	0	2
C	1	2	0

The Initial Boolean Matrix

Figure 1 – A simple example of the enhanced Bowers (2002) matrix representation for loops and multiple relationships.

I then implemented my model in a prototype, and tested my model against a series of controlled experiments. The first experiment consisted in the creation of a set of selected ERD test cases that organised schema types in five different groups. Each type contained a combination of each of the following variable (c.f. Table 1):

- single or multiple relationship
- recursion (i.e. loop)
- cycle

The minimum ERD test contained two entities/two relationships, while the largest ERD test contained five entities eight relationships. I loaded each test into the prototype, namely Graph Edge Validator 2008 (GEV2008).

I subsequently compared the prototype output to the expected potential redundant relationships. I also compared the generated equivalent composed paths, against a set of expected results derived from visual inspection. An equivalent composed path corresponds to any paths that joins an entity A to an entity B, with the same signature and semantic as the original path. The signature of a path is a multi-function of the cardinality and participation of all the entity relationships starting from an entity A to an entity B.

Type #	**Description**
Type 1	Contains only single relationships, at least one recursion and no cycle
Type 2	Contains only single relationships, no recursion and at least one cycle
Type 3	Contains multiple relationships, at least one recursion and at least one cycle
Type 4	Contains multiple relationships, no recursion, and at least one cycle
Type 5	Contains single relationships, recursions, and at least one cycle

Table 1 – Test type description

I classified the output raw data in terms of correctness, incorrectness and missed metrics. Each metric indicated respectively the percentage of correct, incorrect and missed potential redundant relationships generated by the prototype.

My second experiment consisted in creating a set of test cases representing sparse and dense diagrams. I aimed at studying the prototype execution time to complete the analysis of each sparse and dense ERD. A sparse (dense) diagram is an ERD with a relatively small number of entities (large number of entities).

4. RESULTS

4.1 The ‘Quality Discovery’ Results

The results show that the percentages of incorrect potential redundant relationship discoveries are significant for ERDs that supports cycle and/or multiple relationships. The prototype seems to miss potential redundant relationships in rare cases (c.f. Figure 1). However, my case by case analysis indicates that the percentage of errors in cycles is due to the ‘*many:many:many*’ cycle. The other single source of failure for multiple relationships is due to the inability of the proposed algorithm to handle correctly multiple relationships between the first and the second entity in an equivalent composed path.

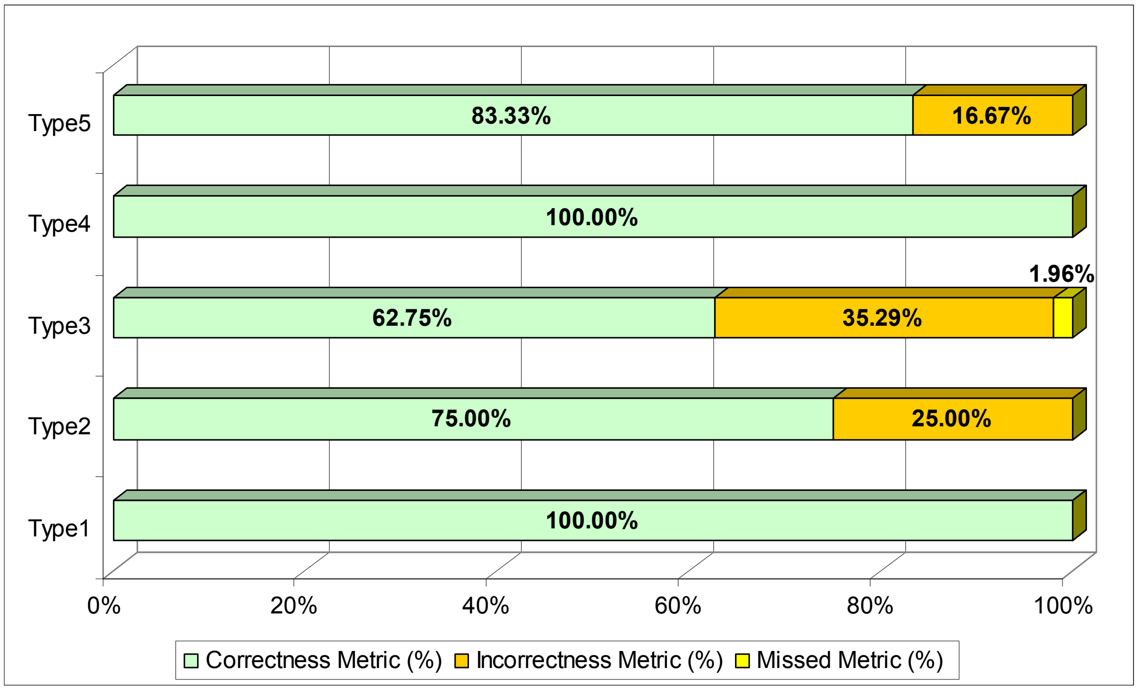


Figure 1 - Potential Path Redundancy Discovery Comparison

Figure 2 shows the result corresponding to the creation of the equivalent composed paths for each discovered potential redundant relationship. The resulting picture is equivalent to the one in Figure 1. The causes of failures are due to the same reasons as the ones described in Figure 1.

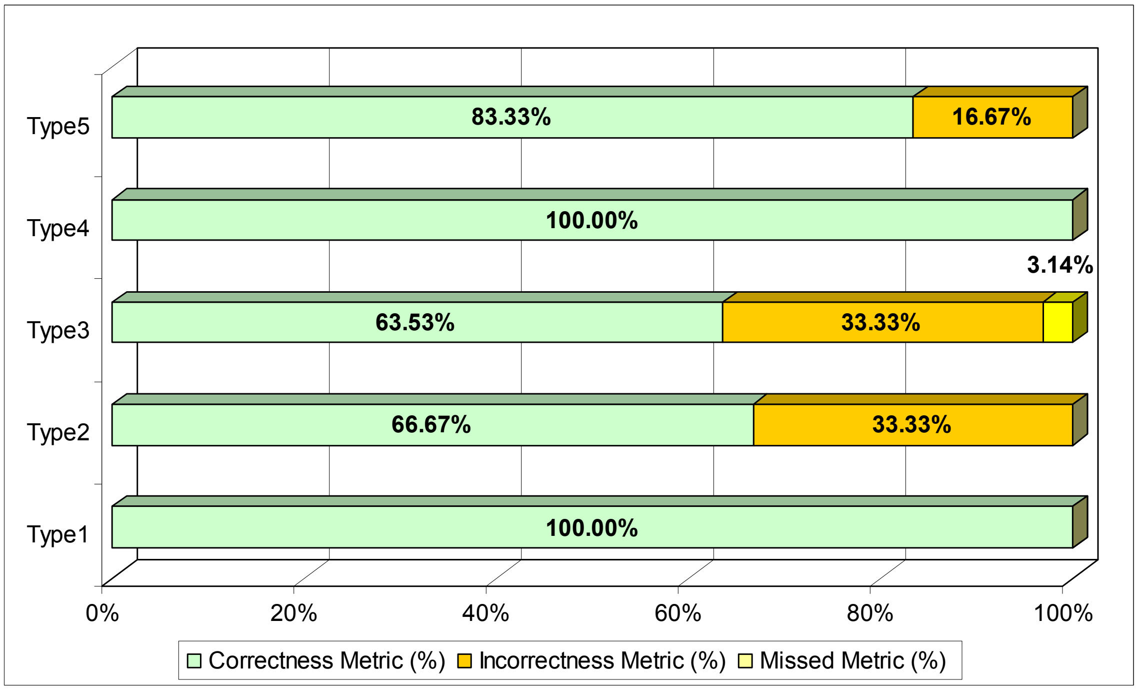


Figure 2 - Potential Equivalent Path Redundancy Discovery Comparison

4.2 The Execution Time Measurement Results

Figure 3 and Figure 4 show the execution times in milliseconds for sparse and dense ERDs, which do not support recursive relationships. The prototype running time tends to grow by a factor of 8 when the number of relationships is doubled. This supports the conclusion that the algorithm is time proportional to $O(R^3)$, where R stands for the number of relationships. However, the picture is not as clear when the number of entities is doubled. The growth rate seems to be proportional to E^3 up to a 40 entities ERD, with E representing the number of entities. When I double the current number of entities to 80, then the time is multiplied by a factor of 15. These results show degradation in performance compared to the results obtained by Bowers (2002) and Sedgewick (2004) in the case of dense ERDs.

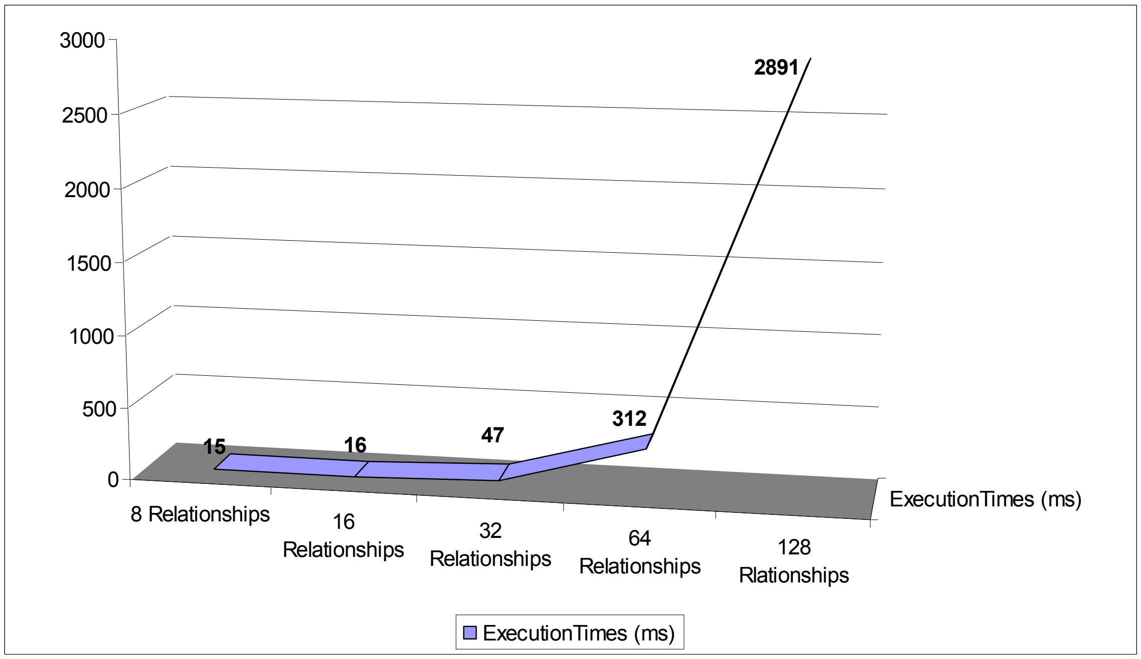


Figure 3 - Execution time in milliseconds (ms) for a sparse ERD (5 entities)

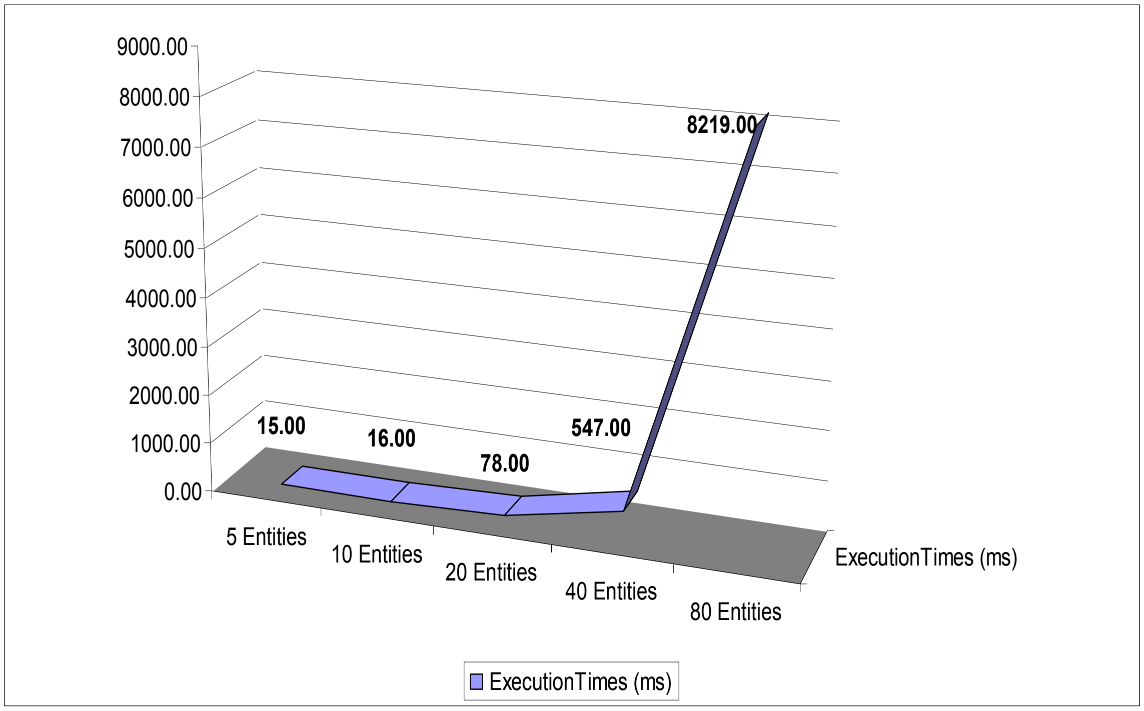


Figure 4 - Execution time in milliseconds (ms) for a dense ERD, where the number of relationships is the double of the number of entities

5. CONCLUSION

I have demonstrated that the first two objectives of this thesis have been attained. The proposed algorithm has a higher rate of success in discovering potential redundant relationships for complex ERDs, compared to solutions studied in the literature review. However, my analysis revealed that there are a number of failures that are due to an oversight in the creation of the partitions in the matrix element. The current incarnation of my implementation loses the relationship context between the first entity in an equivalent path (stored as a partition in the matrix element) and the source entity on the y-axis of the matrix. This purely a code implementation issue, this is not shortcoming within the theoretical model. My proposed solution to tackle this problem is to link this first entity to each of the equivalent composed path in the matrix element. The formal analysis of my model indicates that most of the failures witnessed in this experiment should disappear when the implementation limitation is removed.

The result of the quantitative analysis indicates that the algorithm is scalable with sparse ERDs, but not in the case of dense graphs or graphs with many loops. Furthermore, this test shows a limitation with regards to storing the ERD in memory. The prototype generates an “out of memory exception” for a relatively small ERD (i.e. 5 entities and 8 recursive relationships).

Consequently, I have proposed a number of heuristics to improve the current architecture correctness, scalability, reliability and performances. For example, matrices data could be stored in a local storage, instead of keeping the data in memory. These enhancements demand a substantial revision of the architecture. Due the nature of the modifications and the need for regression testing the entire set of test cases, I decided not to implement the required changes. This was necessary to meet the project deadline. However, these changes could represent a program of further work for researchers who are interested in studying this topic in more depth.

6. REFERENCES

1. Ausiello G., d'Atri A. and Sacca D. (1983), 'Graph Algorithms for Functional Dependency Manipulation', *Journal of the Association for Computing Machinery*, *Vol 30, No 4*, pp 752-766.

2. Azman, S. and Noah M. (2000), 'Intelligent System for Conceptual Modelling of Databases', *TENCON Proceedings*, *Vol2*, pp. 504-508.

3. Bowers D.S. (2000), 'Database schema Improvement Techniques for CASE Tools', *proc. UKAIS conference*, Cardiff, UK.

4. Bowers D.S. (2002), 'Detection of Redundant Arcs in Entity Relationship Conceptual Models', *proc. Int. Workshop on Conceptual Model Quality*, Tampere.

5. Dullea, J. and Song, I-Y. (1998), 'An Analysis of the structural validity of ternary relationships in entity relationship modelling', *Proc. 7th CIKM*, pp. 331-339.

6. Noah, S. and Williams M. (2000), 'Exploring and Validating the Contributions of Real-World Knowledge to the Diagnostic Performance of Automated Database Design Tools', IEEE Computer Society Washington, USA, p177-189.

7. Storey, V. (1993), 'Understanding Semantic Relationships', *Very Large Data Bases (VLDB) Journal, vol. 2, no. 4*, pp. 455-488.

8. Rosenthal, A. and Reiner, D. (1994), 'Tools and Transformation – Rigorous and Otherwise – for Practical Database Design', *ACM Transactions on Database Systems 19(2)*, pp. 167-211.

9. Sedgewick, R. (2004), *Algorithm in Java Third Edition,* USA, Addison Wesley, pp. 1-31 and pp. 149-227.