Design and Implementation of a Graph-based Solution for Tracking Manufacturing Products

Jorge Martinez-Gil[1], Reinhard Stumpner[2], Christian Lettner[1], Mario Pichler[1], Werner Fragner[3]

[1]Software Competence Center Hagenberg GmbH
Softwarepark 21, 4232 Hagenberg, Austria
[2]FAW GmbH
Softwarepark 35, 4232 Hagenberg, Austria
[3]STIWA Group
Softwarepark 37, 4232 Hagenberg, Austria
e-mail: jorge.martinez-gil@scch.at

Abstract. One of the major problems in the manufacturing industry consists of the fact that many parts from different lots are supplied and mixed to a certain degree during an indeterminate number of stages, what makes it very difficult to trace each of these parts from its origin to its presence in a final product. In order to overcome this limitation, we have worked towards the design of a solution aiming to improve the traceability of the products created by STIWA group. This solution is based on the exploitation of graph databases which allows us to significantly reduce response times compared to traditional relational systems.

Keywords: Data Engineering, Graph Databases, Manufacturing

1 Introduction

Quality related errors in manufacturing create a lot of problems for the industrial sector mainly because they lead to a great waste of resources in terms of time, money and effort spent to identify and solve them [4]. For this reason, researchers and practitioners aim to find novel solutions capable of tracking and analyzing manufacturing products in an appropriate and easy to use manner.

One of the challenges of modern manufacturing is that a final product normally consists of different components which themselves can also consist of different components, and so forth. At the lowest level, there is raw material, like steel coils, which is also very relevant to the quality of the final product. Tracking and connecting all the data of the different manufacturing stages is crucial to finding the causes to quality related errors in the final product.

In case the components and raw materials have a one-to-one or one-to-many relationship to the final products, the tracking is quite straightforward and is already implemented satisfactorily in the existing solutions. However, manufacturing products can also be made up of parts that are in lots, and lots can be combined to make other lots. In addition, many different lots can be combined into a new one that can be part of many other ones. This means that we have to deal with a problem involving many-to-many relationships with blurred relationships among the single parts. But providing these relationships is crucial to the end-users, so they can drill down from the final product to the assembled components and ultimately to the used raw materials and can identify causes to problems which are not obvious at the first glance. Unfortunately, the existing solutions based on relational databases are not very useful for the people who are in charge of examining these lots. On the one hand, the existing solutions have bad response times and on the other hand, there are no meaningful probability values available. So the end-users must invest a lot of time to analyze all possibly related data and cannot focus on the data which is really relevant.

In order to alleviate this problem, we have looked for a solution so that it can be possible to track all items from different lots that were used in final products. Our proposed solution is based on the exploitation of graph databases. The major advantage of our approach is that it allows for informed queries, i.e. queries that can lead to an early termination if nodes with no compatible outgoing relations are found. As a result, we have got a system that presents lower execution time for a number of use cases concerning the tracking of items. Therefore, the major contribution of this work can be summarized as the design and implementation of a solution for tracking the manufacturing products created by STIWA. This solution is intended to outperform traditional systems based on relational databases in the specific context of tracking defective items in lots of manufacturing products.

The rest of this work is structured in the following way: Sec. 2 presents the state-of-the-art concerning the current solutions; Sec. 3 describes the design and implementation of our solution and a use case whereby our solution outperforms the traditional tracking systems. Finally, we remark the conclusions and lessons learned from this work.

2 Preliminaries

There is a lack of solutions in this domain despite the fact that most of quality assurance processes require controlling and supervising the whole production chain in order to timely detect human errors and defective materials. But even in the case of passing all quality assurance controls, products can be rejected by end users or other manufacturers if unknown problems appear. Therefore, the capability to track each part of a lot from its origin is very important.

Traditional relational database models are unable to tackle this problem in an effective manner. Therefore, we have focused our research on graph databases [3]. The idea behind graph databases is their capability to store data in nodes and edges versus tables, as found in relational databases. Each node represents an entity, and each edge represents a relationship between two nodes.

It is generally assumed that graph databases have some key advantages over relational databases in this context. The reason is that unlike relational databases, graph databases are designed to store interconnected data what makes it easier to work with these data by not forcing intermediate indexing at every time, and also making it easier to facilitate the evolution of the data models.

In fact, we have identified three major disadvantages of traditional relational databases in comparison with graph databases to tackle this problem:

(1) One of the advantages when dealing with a graph is that in the relational world, foreign key relationships are not relationships in the sense of edges of a graph.
(2) Another drawback of relational databases is that it is not possible to assign properties or labels to relationships. It is possible to give them a name in the database, but it is not possible to visualize them.
(3) Last, but not least, relational databases are not able to scale well when dispatching relationship-like queries [2].

The application domain of graph databases is wide [1]. In fact, many organizations are already using databases of this kind for detecting fraud in monetary transactions, providing product and service recommendations, documenting use cases and lessons learned in a wide range of domains, managing access control in restricted places, network monitoring to identify potential risks and hazards, and so on.

3 Contribution

In order to illustrate the problem with an example, let us think on a situation where a finished part is rejected by the customer because of product quality errors. In that case, the manufacturer of the finished part must make a statement within 24 hours if this error can be restricted to the single rejected product or if a greater amount of parts is affected. In case a greater amount of parts is affected, then the exactly affected lots must be reported to the customer.

The first task of the manufacturer is to find the cause of the quality error. Therefore it must analyze the captured data of the finished part but also the captured data of all assembled components and raw materials. In case the cause of the quality error lies in a specific component or raw material lot the manufacturer must find all finished parts that contain this lot. So finding the right affected lots in a short time can decide whether the manufacturer must recall millions of finished parts or just a few. This means that if we are able to identify the affected lots quickly, it is not necessary to recall all delivered lots.

In order to perform this analytic task, the manufacturer needs to search back and forth across all the data of the finished parts. If a relational database is used, this means that many queries and their corresponding responses would need to be combined. In our approach, the solution is much more intuitive since it is in general possible to easily write queries capable of running over the data in any direction. In fact, the capability to discover and see the connections between different parts of a product allows us performing this tracking in an effective manner.

3.1 Notation

Let L_m^t be a lot of parts produced on machine m at time step t. Every machine m has a buffer b_m where lots to be processed at this machine are poured into, i.e. they are getting blurred. Then, the relation $usage : (L_m^{t2}, L_n^{t1})$ defines that at time slice $t2$ the lot L_m^{t1} has been poured into the buffer of machine m for processing. So beginning at time slice $t2$ parts of lot L_n^{t1} are installed with a certain probability into parts of lot L_m^{t2}. The relation $pred : \{(L_m^{t2}, L_m^{t1})\}$ defines that lot L_m^{t1} is produced before lot L_m^{t2} on machine m and the buffer b_m of machine m was not empty when the production of L_m^{t2} started. Lots that are delivered by other suppliers, i.e. raw materials, are treated the same way. They will be assigned a virtual production machine number and time slice which uniquely identifies the batch number from the supplier.

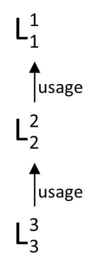

Fig. 1: One-to-one relationship between lots.

Using this notation, simple examples for one-to-one and one-to-many relationship between lots without blurring are depicted in Fig. 1 and Fig. 2. An example with a many-to-many relationship that includes blurring of lots is given in Fig. 3. By following the edges of the graph, the lots that are built in other lots can be determined easily, i.e. for lot L_3^4 parts of the lots $\{L_1^1, L_1^2, L_2^2, L_2^3\}$ may be included, or for lot L_3^5 parts of the lots $\{L_1^2, L_1^3, L_2^3, L_2^4\}$ may be included. The missing relation *pred* between L_3^4 and L_3^5 indicates that the buffer of machine 3 was empty before the production of lot L_3^5 started so no blurring of lots could have occurred. The distance between lots can be used as a basis to determine the probability a certain part of a lot is used in another lot. A more precise determination of probabilities would also require to consider buffer levels during manufacturing but that is beyond of the scope of this work.

3.2 Implementation

A prototypical solution to the proposed manufacturing product tracking system has been implemented using the graph database OrientDB[1]. OrientDB is a multimodal NoSQL database that combines properties of document-oriented and graph databases. It allows to define graph structures using concepts for nodes

[1] https://orientdb.com/

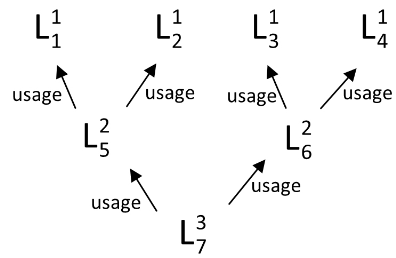

Fig. 2: One-to-many relationship between lots.

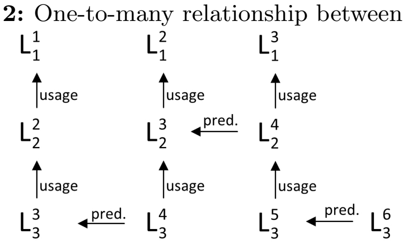

Fig. 3: Many-to-many relationship between lots, i.e. blurred lots.

and edges, but also allows to append complex data to nodes in the form of documents. Nodes and edges can have attributes (e.g. edge weight or similar). Moreover, inheritable classes can be defined for the nodes and edges which can be extended flexibly. The query language is an adapted form of SQL. Compared to implementations based on relational systems, using a graph database leads to an efficient and scalable solution in which the problem at hand can be modeled easily. Traversing graphs modeled in relational systems would require to write nested and recursive queries that are difficult to maintain and provide bad comparable performance.

3.3 Use Case

Based on the implementation that has been described above, such as graph-based approach is considered to have a positive impact on the operations of STIWA group. In particular, the depth calculation, i.e. the distance between lots, could be used as a basis for the construction probability of a part in a product. The search with BREADTH FIRST returns the real depth in the graph. The greater the depth, the less likely it is that the lot has been incorporated into the end product.

An example query to get the shortest path between lot 33 : 27050 and 29 : 2667 can be easily written as:

```
SELECT expand(path) FROM (
SELECT shortestPath(#33:27050, #29:2667, 'OUT') AS path UNWIND path);
```

The result of this query is depicted in Fig. 4. The number of edges between the lots gives a basic probability that parts of lot 29 : 2667 are built in parts of lot 33 : 27050.

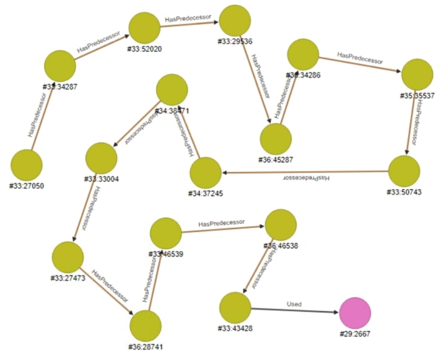

Fig. 4: Shortest path between nodes #33:27050 and #29:2667 in edge direction

4 Conclusions and Future Work

We have presented a solution for tracking each component that comprises manufacturing products from STIWA group through diverse stages of the production chain. Our solution has been modeled using a graph database. In this way, our approach provides an improved level of both transparency and traceability. These factors are able to facilitate the analysis of all manufacturing products as well as the capability to look for final products that could be affected by some specific problem. In this way, our approach presents more efficient modeling and querying mechanisms than traditional approaches. We envision that graph databases holds lot of unrealized potential in the next years as companies will be moving towards approaches being able of better data analysis and exploration.

Acknowledgements

This work has been supported by the Austrian Ministry for Transport, Innovation and Technology, the Federal Ministry of Science, Research and Economy, and the Province of Upper Austria in the frame of the COMET center SCCH.

References

1. M. El Abri. *Probabilistic relational models learning from graph databases. (Apprentissage des modèles probabilistes relationnels à partir des bases de données graphe)*. PhD thesis, University of Nantes, France, 2018.
2. R. Angles, M. Arenas, P. Barceló, A. Hogan, J. L. Reutter, and D. Vrgoc. Foundations of modern query languages for graph databases. *ACM Comput. Surv.*, 50(5):68:1–68:40, 2017.
3. D. Fernandes and J. Bernardino. Graph databases comparison: Allegrograph, arangodb, infinitegraph, neo4j, and orientdb. In *Proceedings of the 7th International Conference on Data Science, Technology and Applications, DATA 2018, Porto, Portugal, July 26-28, 2018.*, pages 373–380, 2018.
4. Kuo-Ren Lou and Lu Wang. Optimal lot-sizing policy for a manufacturer with defective items in a supply chain with up-stream and down-stream trade credits. *Computers & Industrial Engineering*, 66(4):1125–1130, 2013.