Detecting Evidence of Fraud in the Brazilian Government Using Graph Databases

Gustavo C.G. van Erven[1], Maristela Holanda[2(✉)], and Rommel N. Carvalho[1,2]

[1] Department of Research and Strategic Information (DIE),
Ministry of Transparency, Supervision and Office of the Comptroller General (CGU),
Brasília, Brazil
{gustavo.erven,rommel.carvalho}@cgu.gov.br
[2] Department of Computer Science (CIC), University of Brasília (UnB),
Brasília, Brazil
mholanda@unb.br

Abstract. In the *International Monetary Funding Staff Discussion Note No. 16/05* of May 11/2016, corruption was cited as one of the "most important problems facing the world today". This prompted agencies around the world to step up efforts on finding techniques to combat corruption in various contexts, such as fraud in government procurement processes. This type of fraud is usually orchestrated by groups of companies that manipulate competition so that processes are awarded to predetermined companies. Given this scenario, finding relationships between companies from linking information, such as partners or telephones, is essential to gathering evidence that can expose how the criminal activity is organized and carried out. Since relationships can be modeled as a network, graph databases prove to be an appropriate tool in finding these links. This paper presents a study on using graph databases to identify evidence of fraud in procurement processes. Firstly, the scope of the research and the model used are presented, and subsequently the queries and their results are shown and discussed, indicating possibles evidence of fraud in the real dataset.

Keywords: Graph database · Corruption control · Fraud detection · Government accountability

1 Introduction

The Ministry of Transparency, Supervision and Office of the Comptroller-General of Brazil[1] (CGU) is a government agency, currently regulated by the Law 13,341/2016, which carries out activities for preventing and combating corruption. In order to achieve these goals, the auditors in CGU try to identify people, companies, or any other entity that uses federal public money to find evidence of fraud or corruption. This evidence can come from different sources, such as operational reports from the field or by documentation and database analysis.

[1] http://www.cgu.gov.br.

Á. Rocha et al. (eds.), *Recent Advances in Information Systems and Technologies*,
Advances in Intelligent Systems and Computing 570, DOI 10.1007/978-3-319-56538-5_47

Particularly, in the case of database analysis, the main operation currently performed over records is a simple data match task. The data from one database is matched with data in another, to identify anomalous situations or irregularities. One of the areas in which this is frequently applied is in public procurement processes [5,8]. In this context, this paper presents a proposal for using graph databases to aid in fraud identification.

The remainder of this paper is organized as follows: Sect. 2 presents a brief introduction about the Brazilian procurement process. Section 3 presents the Related Works. Section 4 presents the data model to implement the graph database. Section 5 presents a sample for the current method used in CGU. Section 6 presents the queries and results obtained from a database of a specific target. Finally, Sect. 7 presents the final conclusions and future works.

2 Brazilian Procurement Process

In Brazilian procurement processes, specifically with regard to the Law 10,520/02 which provides for bidding in a reverse auction (*Pregão*), the company that wants to sell to the government must apply to an item in the process and offer an initial price when the bid starts. As the process evolves, companies can submit several bids, lowering price until the others give up.

Unfortunately, people can use several companies to participate in the same procurement process and use them to fake competition. In general, the fraud is operationalized by the *collusion* between them, which is "an illegal agreement between rivals that attempts to disrupt the market's equilibrium"[2]. In other cases, a closed group can open new companies they control instead of making deals with others they do not control.

A technique used to hide collusion is to create a company chain. This technique involves people in the group, or even their companies, opening other companies, and these companies opening other companies and so on. In this case, they register some or even all of the companies in the chain in the same procurement process to rig competition. Therefore, a possible method to detecting irregularity is by following the relationships between the companies and their partners.

To support these activities and investigations, the Department of Research and Strategic Information (DIE) in CGU uses several database queries. These queries focus mainly on data about joint participation in procurement processes where these companies share owners and/or own each other. However, these data can be structured more easily using a graph, where the entities are vertices and the relationships edges. Using these data as a graph, enables us to find relationships through graph algorithms, as shortest paths.

Usually, data is stored in tables in relational database systems and then transformed into graphs to be processed. However, the growing technology of graph databases has brought support for storing and querying connected data

[2] http://www.investopedia.com/terms/c/collusion.asp.

more easily [1]. Several systems, such as Neo4j[3] and Titan[4], also have features to move between relationships.

Thus, this paper presents a study about using graph databases to detect evidence of public procurement fraud. In this paper, the procurement data was loaded into Neo4j, an attributed graph database.

3 Related Works

Graph structures are already used to study or identify criminal activities and relationships. A visualization approach was used in Sparrow [12] and McIllwain [10] to create and support analysis of relationships as networks. In Hulst [9], Social Network Analysis (SNA), a graph with people connected by resources and other people, is used as well, and enables the development of several studies about how to build scenarios, identify risks and destabilize criminal networks.

A network can include several suspects, each of them connected to another by information, such as telephones, companies, or bank accounts. It is easier, in most cases, to identify central agents and their communication flows in a visual way or even using graph algorithms and metrics [14]. Pandit [11] built a system that explores the graph structure of networks to detect fraud from collected online data. Other tools, as Pajek [3], are useful to work with SNA in a criminal context [6].

Graph databases [1] have been widely used in several areas as shown in [13]. In the context of fraud, attributed graph databases appear in real operations, such as the recent Panama Papers that "exposed highly connected networks of offshore tax structures used by the world's richest elites"[5]. Arora [2] presents techniques supported by graph databases to detect crimes. Branting [4] uses graph databases and other tools to process and estimate healthcare fraud risk.

4 Data Model of Procurement Process

This section presents a diagram that was used to build the public procurement graph database. The graph model used in this paper is composed of vertices and edges that can have properties or attributes. The domain chosen is relevant because it covers 2010 to 2014, in which more than USD 18 billions were spent in goods and services through 196.9 thousand processes, and are available in a Brazilian Government open data site[6].

The public procurement process was modeled to initially cover a few entities, such as partners, telephones, procurement items and companies, which is important to start an investigation task. Figure 1 presents the graph diagram used in this paper. The diagram notation used to drawn it is the MDG–NoSQL [7].

[3] https://neo4j.com/.
[4] http://titan.thinkaurelius.com/.
[5] https://neo4j.com/blog/analyzing-panama-papers-neo4j/.
[6] http://compras.dados.gov.br/docs/home.html.

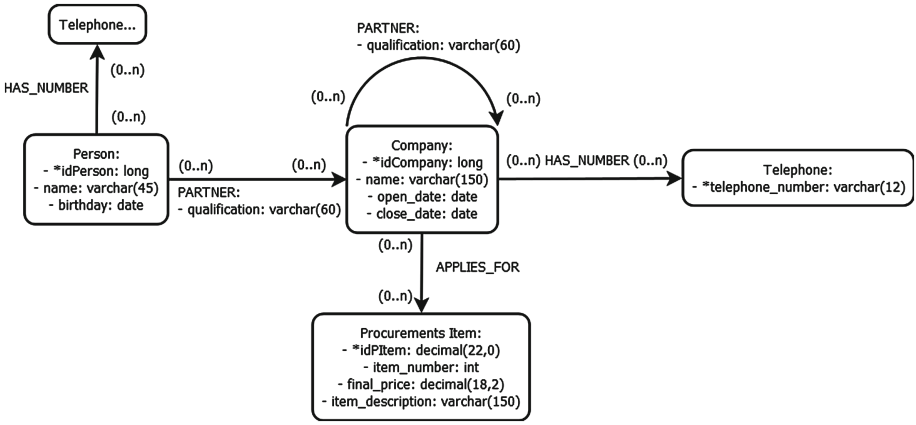

Fig. 1. Graph datamodel for procurement processes.

The model focuses on `Company`, which has a unique identifier, `idCompany`. The attribute `name` describes the company, the attribute `open_date` stores when the company began its operations and the attribute `close_date` holds the date when the company finished its activities or an invalid date.

The `Procurements_Item` keeps data about items from a procurement process. In the data model, the procurement process is not defined as an entity to group the items of the same process, however the information can be derived from the attribute `item_number`. The attributes `final_price` and `item_description` detail the item price at the end of the process and the description explains what was bought.

As described in Sect. 2, companies can apply for several procurement items and each item can have several companies competing for it. This relationship is represented by the `APPLIES_FOR` edge and the cardinality (0..n). As companies can create situations for collusion or groups to fraud the procurement processes, an approach to finding evidence is looking for the connections between these companies. The connections chosen as the primary focus of this paper are between `Person` and `Telephone`. These two entities are the building blocks to finding evidence of fraud in this work. If the companies competing for the same item are connected through partners or telephones, then there is evidence of collusion.

`Person` entity are people who can have a relationship with the companies. Other relationships, such as relatives and friends would be added in the future. The main information are `name` and `birthday`. `Person` may have a relationship with `Company` through the `PARTNER` edge. As a `Person` can be `PARTNER` in several companies, and they can have several instances of `Person` as partners, the cardinality is (0..n). The last entity is `Telephone`, which is connected to `Person` and `Company` through `HAS_NUMBER`. Similar to `PARTNER`, the `HAS_NUMBER` cardinality is (0..n). The unique attribute in `Telephone` is `telephone_number`, which works as an identifier as well.

This diagram was implemented in Neo4j, which uses a specific language called `CYPHER`[7] to query the data. `CYPHER` is easy to use and allows us to search using a description of graph patterns, shown later in the text.

Finally, the queries were defined together with the CGU's specialists and tested over real data, but the effective results (indicating evidence of fraud or not) based on data obtained are still under investigation.

5 Current Method in CGU

Currently, CGU specialists execute several queries in relational databases to extract information about targets. The CGU specialists are fundamental in this work because they are directly involved with auditing and fraud detection activity. They are the final users of the information extracted from theses queries.

The specialists chose some targets to analyze. Considering that there was no priority order nor a red alert for any company, the auditors prefer to start the analysis by searching for the top ten companies with the highest summation for unitary item values or number of items the company registered to bid. The hypothesis is that the higher the item's value, the higher the likelihood of the fraud. To support the analysis process, there are several sources that are grouped and then consulted by SQL queries.

Another important query, and our final milestone for exploration analysis in the next section, is the question: *In the procurement items that the target company has applied for are there other companies with whom the target company has relationships?*

6 Querying for Evidence

This section presents an exploration analysis and results in the graph database for the queries used by the CGU specialist, as well as some findings that point to suspicious scenarios.

The data was loaded into Neo4j by an ETL process (Extract, Transformation and Load). Since the data is originally in a relational database, each instance had to be mapped to a vertex and its relationships recreated as edges with the respective label. The final database is about 58 GB, with almost 270 millions vertices and 184 millions edges.

After the data was loaded, we started with the first query to explore the data as defined by the specialists. The query in Listing 1.1 merely retrieves this information, but in descending order of total price. In this paper, the identification numbers (id) of the companies in question were changed to preserve confidentiality.

Listing 1.1. Top 10 companies that apply for procurement's items.

```
match (c:Company)-[r:APPLIES_FOR]->(i:Procurements_Item)
return c, count(r) as total_items, sum(i.price) as sum_prices
order by sum_prices desc limit 10
```

[7] https://neo4j.com/docs/developer-manual/current/cypher/.

With this first set of companies in hand, the auditor can now use a company id from the top ten list to retrieve the procurement items that it applied for by using the query from Listing 1.2. The query can be combined with totals, sums or other aggregates and filters.

Listing 1.2. Procurement's items for a company.

```
match (c:Company{idComp:0001})-[r:APPLIES_FOR]->(i:
    Procurements_Item) return c, r, i
```

Up to this point, these queries, although relevant to help in a fraud investigation, do not use specific features that a graph database offers. Since graph structures simplify moving between vertices, a graph can be used to find relationships between them. The Listing 1.3 presents a query to retrieve all common items between two companies. The idea is to describe the sub graph pattern using `CYPHER` so that the engine can try to find it in the graph data. For this example, the sub graph is comprised of two companies, identified by `idComp` 0001 and `idComp` 0002, which applied for the same items, evidenced by the procurement item generalized in the middle of the relationships.

Listing 1.3. Common Items applied by two companies.

```
match (c1:Company{idComp:0001})-[r1:APPLIES_FOR]->(i:
    Procurements_Items)<-[r2:APPLIES_FOR]-(c2:Company{idComp
    :0002}) return c1, r1, r2, i, c2
```

This query is equivalent to searching for relationships between companies with 2-hop distances. Therefore, the query could be rewritten to use some powerful functions common in graph databases that take advantage of the graph structure, namely, the short path function. In Listing 1.4 the query was modified to use the `allShortestPaths` function, which searches for all the shortest paths between the vertices. An advantage of using this function is that it retrieves only the sub graph with the shortest paths. If the pattern such as Listing 1.3 is used, all possible nodes and edges in the subgraph are retrieved and the query can take more time to finish.

Listing 1.4. Common items using all shortest path function.

```
match sp = allShortestPaths((c1:Company{idComp:0001})-[r:
    APPLIES_FOR*..2]-(c2:Company{idComp:0002})) return sp
```

Although this approach does not retrieve all paths, it may seem that relevant information is lost at first, however, this approach can be faster, considering that all possibilities would not have to be searched, and a connection would be found between the companies, if one exists. From this information, an auditor can expand the network to other relevant links as more information is needed during the investigation process.

These queries shown from Listing 1.1 to Listing 1.4 helped to find companies that applied for the same item in a procurement process. Nevertheless, the

results from these queries are not only expected, but also desired, since increasing the companies that participate increases the competition. Thus, this is not necessarily an evidence of fraud. However, as explained earlier, companies owned by the same group can be used to simulate competition, which is a type of fraud. Therefore, it is essential to retrieving information on the relationship between these companies, in order to verify if there is a possibility of collusion.

Therefore, if companies have paths passing through the same partners or telephones, there is a chance that these companies are controlled by the same group. As shown in Listing 1.4, the telephone relationship can be retrieved using the shortest path function, as presented in Listing 1.5.

Listing 1.5. Querying for direct relationships by telephones.

```
match sp = allShortestPaths((c1:Company{idComp:0001})-[r:
    HAS_NUMBER*..2]-(c2:Company{idComp:0002}))
return sp
```

The partners related to a target company can be found using a direct query presented in Listing 1.6. This information is another building block in the search for the relationship between companies.

Listing 1.6. Querying for partners of a target company.

```
match (c:Company{idComp:0001})<-[r:PARTNER]-(p) return c, r, p
```

Although companies used in fraud schemes belong to a group, the relationship between them are often hidden by a sequence of partnerships. Therefore, when companies apply for items, there is no evidence of influence from one to another on the first level. The partners' influence can flow from one company to another through a chain of partnerships. Thus, as the objective is to find some evidence of these chains, a next step is to search for deeper relationships between the partners.

The CGU specialist defined that 6-hops from one target to another are enough to find relevant relationships. As several vertices can exist between the targets, a query for a subgraph pattern can take a long time. Thus, the shortest path function can improve the search in the graph database to find a first link, because it returns the information after finding the relationships on the lower level, as presented in Listing 1.7.

Listing 1.7. Shortest path for partners.

```
match sp = allShortestPaths((c1:Company{idComp:0001})-[r:PARTNER*
    1..6]-(c2:Company{idComp:0002})) return sp
```

The query for the shortest path in Listing 1.7 does not consider the direction of the edges, because companies have partners only as incoming edges. Considering the graph as undirected, enables the algorithm to reach all connected vertices moving through the `PARTNER` relationships, regardless from where it starts.

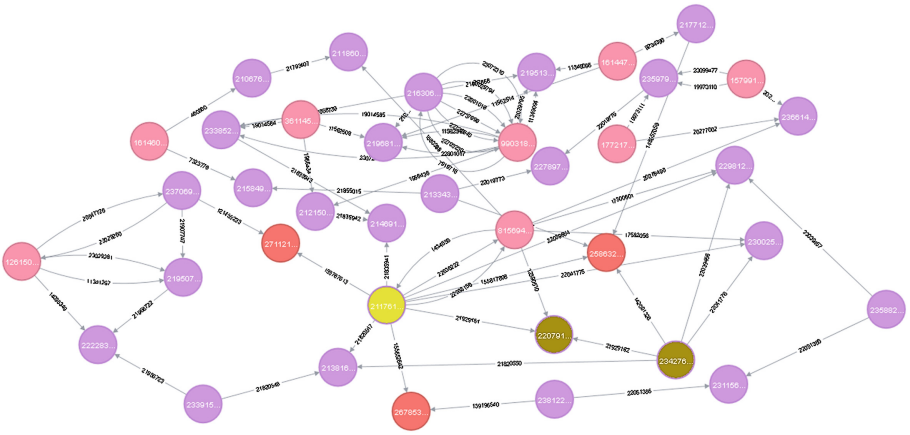

Fig. 2. Companies with at most 6 hop relationships in the same procurement's items from a selected target.

In any case, at the very least, the queries about procurement items in Listing 1.3 and partners relationship in Listing 1.7 can be nested to answer the question describe in Sect. 5. The Listing 1.8 presents the query used to search for this information.

Listing 1.8. Relationship between a target and companies.

```
match (c1:Company{idComp:0001})-[r1:APPLIES_FOR]->(i:
    Procurements_Item)<-[r2:APPLIES_FOR]-(e2:PessoaJuridica)
match sp = allShortestPaths((c1)-[r:PARTNER|HAS_NUMBER*1..6]-(c2)
    )
return sp, r1, r2, i
```

The variables `c1` is bound to the target company and the variable `c2` is used to bind each company that applied for the same item searched in `allShortestPath` function. Therefore, the graph database takes advantage of the connected data structure to answer the question. A sample this type of query is presented in Fig. 2. The number of vertices was limited to 25 nodes for better visualization.

The red circles are procurement items, and the pink ones are partners. The companies are the purple circles, except for the target, which is represented by the yellow circle and two companies painted with dark yellow, highlighting an example of a path shared to a common item. Items link the companies that have applied to the same item as the target. The companies and people between them are the partner chain that implies a possible flow of influence among the partners, and, thereby, collusion. Therefore, this would be a situation where these companies are hiding their relationship to simulate or compromise the competition in the process.

Another situation is presented in Fig. 3 using the same target, but with less vertices and only connections that are 3-hop deep.

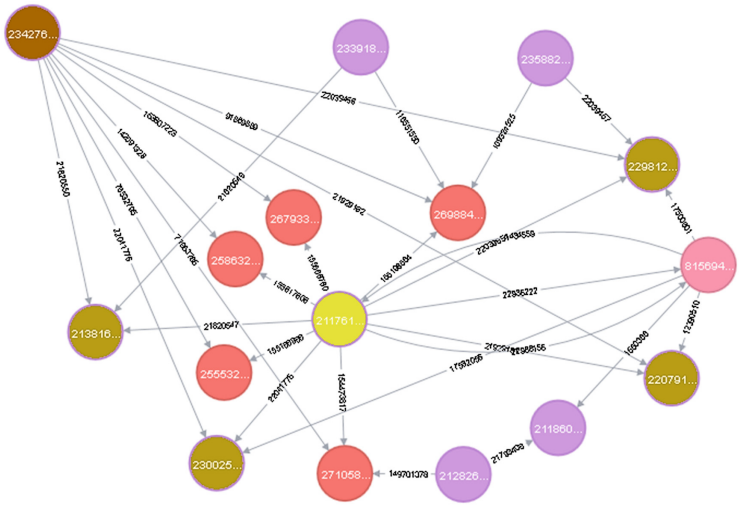

Fig. 3. Companies with at most 3 hop relationships in the same procurement's items from a selected target.

In this case, it is clear that the target company applied for five distinct items highlighted in red, and in all of them, companies related with it also participated in the procurement process. However, in this subgraph, the target has four companies highlighted with dark yellow in common with the company painted in brown. The subgraph exposes a possible fraud scenario of the target, as well as the brown company that appears to have similar resources and influence.

7 Conclusions and Future Works

This paper presents the uses of a graph database to search for evidence of fraud in public procurement processes, a relevant activity in government agencies. As the CGU specialists explain, searching for evidence of fraud can start either from a specific complaint or from a target selected in a group of high risk. Nevertheless, it will eventually evolve to several tasks where finding relationships between targets, and other entities is of major concern. A graph database can address this problem, supporting the search for relevant links, such as a chain of partners and/or telephones.

In this paper, the public procurement process data were implemented in the Neo4j and queried for relationships between companies that applied for the same procurement items using `CYPHER`. An investigation sample was used, starting from the top 10 companies with the highest application values. As a result, some companies were identified and a scenario with a central target was presented as being a possible fraud situation, namely collusion.

Future studies include other techniques such as visualization, graph data mining and large graph data processing, which can be used together with graph databases to develop a more complete fraud detection system.

References

1. Angles, R.: A comparison of current graph database models. In: 2012 IEEE 28th International Conference on Data Engineering Workshops (ICDEW), pp. 171–177, April 2012
2. Arora, K., Bhargava, D.S., Srivastava, A.: GMT (Graph Mining Techniques) for crime detection, comparison with the proposed algorithm. Int. J. Adv. Res. Comput. Sci. Electron. Eng. (IJARCSEE) **3**(2), 108–112 (2014)
3. Batagelj, V., Mrvar, A.: Pajek - analysis and visualization of large networks. In: Jünger, M., Mutzel, P. (eds.) Graph Drawing Software, pp. 77–103. Springer, Heidelberg (2004)
4. Branting, L.K., Reeder, F., Gold, J., Champney, T.: Graph analytics for healthcare fraud risk estimation. In: 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), pp. 845–851, August 2016
5. Carvalho, R.N., Laskey, K.B., Costa, P.C.D.: Uncertainty modeling process for semantic technology. PeerJ Comput. Sci. **2**, e77 (2016)
6. Cheong, T.-M., Si, Y.-W.: Event-based approach to money laundering data analysis and visualization. In: Proceedings of the 3rd International Symposium on Visual Information Communication, VINCI 2010, pp. 21:1–21:11. ACM, New York (2010)
7. van Erven, G.C.G.: MDG-NoSQL: modelo de dados para bancos NoSQL baseados em grafos, December 2015
8. Frizzo, H., Oliveira, P.: PLC - Public procurement in Brazil: overview, October 2014
9. van der Hulst, R.C.: Introduction to Social Network Analysis (SNA) as an investigative tool. Trends Organized Crime **12**(2), 101–121 (2009)
10. McIllwain, J.S.: Organized crime: a social network approach. Crime Law Soc. Change **32**(4), 301–323 (1999)
11. Pandit, S., Chau, D.H., Wang, S., Faloutsos, C.: Netprobe: a fast and scalable system for fraud detection in online auction networks. In: Proceedings of the 16th International Conference on World Wide Web, WWW 2007, pp. 201–210. ACM, New York (2007)
12. Sparrow, M.K.: The application of network analysis to criminal intelligence: an assessment of the prospects. Soc. Netw. **13**(3), 251–274 (1991)
13. Srinivasa. S.: Data, storage and index models for graph databases. In: Sakr, S., Pardede, E. (eds.) Graph Data Management, pp. 47–70. IGI Global (2011)
14. Xu, J., Chen, H.: Criminal network analysis and visualization. Commun. ACM **48**(6), 100–107 (2005)