A Graph Theoretical Approach for Identifying Fraudulent Transactions in Circular Trading

Priya, Jithin Mathews, K. Sandeep Kumar, Ch. Sobhan Babu
Department of Computer Science
Indian Institute of Technology Hyderabad
India
Email: {cs15resch11007, cs15resch11004, cs15mtech11017, sobhan} @iith.ac.in

S.V. Kasi Visweswara Rao
Department of Commercial Taxes
Government of Telangana
India
Email: svkasivrao@gmail.com

***Abstract*—Circular trading is an infamous technique used by tax evaders to confuse tax enforcement officers from detecting suspicious transactions. Dealers using this technique superimpose suspicious transactions by several illegitimate sales transactions in a circular manner. In this paper, we address this problem by developing an algorithm that detects circular trading and removes the illegitimate cycles to uncover the suspicious transactions. We formulate the problem as finding and then deleting specific type of cycles in a directed edge-labeled multigraph. We run this algorithm on the commercial tax dataset provided by the government of Telangana, India, and discovered several suspicious transactions.**

***Keywords*—social network analysis; fraud detection; circular trading; value added tax.**

I. Introduction

A tax is a mandatory financial charge or some other type of levy imposed upon an individual or a legal entity by a state in order to fund various public expenditures. There are mainly two types of tax structures, direct tax and indirect tax. In this paper, we focus on the indirect tax structure. The indirect tax (Value added Tax (VAT) [3] and Goods and Services Tax (GST) [13]) is a tax collected by an intermediary (such as a retail store) from the person who bears the ultimate economic burden of the tax (such as the consumer).

A. Value Added Tax

VAT is a consumption tax that is collected incrementally based on the value added to the goods at each stage of production. Figure 1 explains the transfer of tax from the consumer to the government.

- The manufacturer purchases raw material of value ₹ 1000 from the seller (Here, we represent currency in the form of Indian currency denoted by the symbol ₹ or *Rs.*). He pays ₹ 1100 (₹ 1000 towards raw material and ₹ 100 towards tax, at the tax rate of 10%) to the seller. The seller remits the tax he collected from the manufacturer (₹ 100 rupees) to the government.
- The retailer purchases goods of value ₹ 1200 from the manufacturer. He pays ₹ 1320 (₹ 1200 towards the goods and ₹ 120 towards tax, at the tax rate of 10%) to the manufacturer. The manufacturer remits the difference between the tax he collected from the retailer and the one paid to the raw material seller, which is ₹ 120 - ₹ 100 = ₹ 20, to the government.
- The end user purchases goods of value ₹ 1500 from the retailer. He pays ₹ 1650 (₹ 1500 towards goods and ₹ 150 towards tax, at the tax rate of 10%) to the retailer. The retailer remits the difference between the tax he collected from the end user and the one paid to the manufacturer, which is ₹ 150 - ₹ 120 = ₹ 30, to the government.

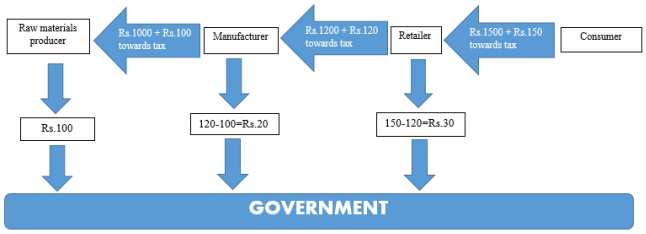

Figure 1. Cash flow diagram of VAT

Note that the total tax received by the government is ₹ 150, which is paid by the consumer.

B. VAT Evasion Methods

Illegal evasion of VAT often entails taxpayers deliberately misrepresenting the true state of their business affairs to the tax authorities to reduce their tax liability and includes dishonest tax reporting. The dealers will try to reduce their liability to pay tax by indulging in some of the following practices.

- Do not collect the tax.
- Evade tax by collecting the tax but not reporting it to the government and thus pocketing the tax.
- Show fake exempt transactions, i.e., branch transfer.
- Show fake sales to other states/countries to claim lower tax rate.

 ISBN: 978-1-61208-603-3

- Bill trading is an organized crime [10]. In this approach, one dealer sells goods to a buyer without issuing an invoice but collecting tax. He then issues a fake invoice to another dealer, who uses it to minimize his tax liability.

To hide these manipulations, which are easily detectable by tax authorities, dealers exchange goods (can be fictitious goods) among themselves without any value-add. This tax evasion technique is called *circular trading* [2] [4] [8].

C. Circular Trading

One of the major malpractices to VAT evasion is *circular trading*. The motivation for circular trading is to hide suspicious sales/purchase transactions, which can be detectable by sales authorities. They hide these suspicious transactions by doing several illegitimate sale transactions among themselves in a short duration in a circular manner without any *value-add*, as shown in Figure 2. Since there is no value-add for the illegitimate transactions, they do not pay VAT on these illegitimate transactions and confuse the tax authorities about suspicious transactions. In this manner, they complicate the process of identifying suspicious transactions.

Some of these fraudulent traders who are part of circular trading can be fictitious (dummy) traders created by malicious real traders.

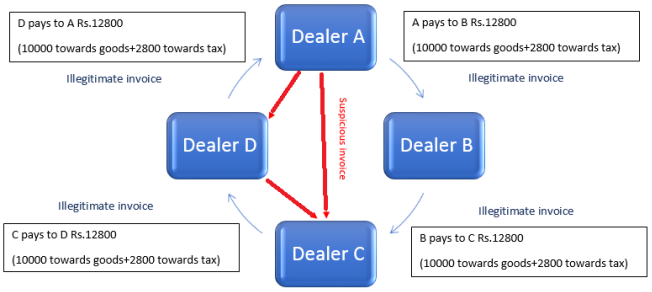

Figure 2. circular flow of sales/purchases

In Figure 2, transactions shown in thick line from A to D, A to C and D to C are suspicious transactions. In order to complicate the process of detection of these suspicious transactions by tax authorities, dealers superimpose illegitimate transactions on the suspicious transaction, as shown using thin lines. Note that the superimposition of thin lines transactions did not change the tax liability of any dealer.

The problem that we address in this paper is to identify suspicious sales transactions, which are superimposed by several illegitimate sales transactions.

The difficulties in identifying suspicious sales transactions are the large size of sales database, complex sequence of illegitimate sales transactions, large number of traders, unknown number and identity of traders in the circular trading. In this paper, we propose algorithms to remove illegitimate sales transactions, which are superimposed on suspicious sales transactions. This allows tax authorities to identify suspicious transactions in an easy manner.

The structure of the paper is as follows. In Section II, several approaches which are available in literature to detect circular trading have been described. In Section III, the problem is formulated as finding specific type of cycles in directed edge-weighted multigraph and an overview of the solution is given. In Section IV, a detailed algorithm along with its proof of correctness is given. In Section V, we have taken up a case in which eight dealers are doing heavy circular trading among themselves and analyzed the case in detail using our proposed algorithm.

II. Related Work

Most approaches for detecting circular trading are concentrated on stock market trading. In [4], a graph clustering algorithm specially tailored for detecting collusion sets in stock market is given. A novel feature of this approach is the use of Dempster–Schafer theory of evidence to combine the candidate collusion sets. In [7], a method is proposed to detect the potential collusive cliques involved in an instrument of future markets by first calculating the correlation coefficient between any two eligible unified aggregated time series of signed order volume, and then combining the connected components from multiple sparsified weighted graphs constructed by using the correlation matrices where each correlation coefficient is over a user-specified threshold. In [5], the authors proposed an approach to detect collusion sets using Markov Clustering Algorithm (MCL). Their method can detect purely circular collusions as well as cross trading collusions. They have used MCL at various strength of "residual value" to detect different cluster sets from the same stock flow graph. Classic centrality functions for graphs are able to identify the key players of a network or their intermediaries. However, these functions provide little information in large and heterogeneous graphs. Often the most central elements of the network (usually too many) are not related to a collections of actors of interest, such as a group of drug traffickers or fraudsters. Instead, its high centrality is due to the good relations of these central elements with other honorable actors. In [14], the authors introduced complicity functions, which are capable of identifying the intermediaries in a group of actors, avoiding core elements that have nothing to do with this group. These functions can classify a group of criminals according to the strength of their relationships with other actors to facilitate the detection of organized crime rings. In [11], the authors presented a statistics-based method for detecting value-added tax evasion by Kazakhstani legal entities. Starting from features selection they performed an initial exploratory data analysis using Kohonen self-organizing maps; this allowed them to make basic assumptions on the nature of tax compliant companies. Then they selected a statistical model and proposed an algorithm to estimate its parameters in an unsupervised manner. In [9], the authors presented a case study of a pilot project that

 ISBN: 978-1-61208-603-3

was developed to evaluate the use of data mining in audit selection for the Minnesota Department of Revenue (DOR). They described the manual audit selection process used at the time of the pilot project for Sales and Use taxes, discussed the data from various sources, addressed issues regarding feature selection, and explained the data mining techniques used.

Several approaches are available in the literature to detect cycles in directed graphs. Traditional methods rely on depth-first searches (DFS)[1], exploiting the fact that a graph has a cycle if and only if the DFS finds a so-called back edge. Recent works on the topic include distributed algorithms that aim at maintaining an acyclic graph when new links are added to an initially acyclic graph [6]. In [15], the authors introduced a new problem, cycle detection and removal with vertex priority. It proposes a multithreading iterative algorithm to solve this problem for large-scale graphs on personal computers. In [12], the authors considered the problem of detecting a cycle in a directed graph that grows by arc insertions, and the related problems of maintaining a topological order and the strong components of such a graph.

III. Problem Formulation

A. Sales Database Format

Table 1 contains a few fields of the sales transaction database.

Table I. SALES TRANSACTIONS DATABASE

S.NO	Seller	Buyer	Time of Sales	Sales in Rupees
1	Dealer A	Dealer B	2017/01/03/10:30	10000
2	Dealer C	Dealer D	2017/01/03/12:00	15000
3	Dealer A	Dealer D	2017/01/04/09:00	12000
4	Dealer B	Dealer C	2017/01/04/10:00	14000
5	Dealer C	Dealer A	2017/01/04/10:30	10000

The actual sales transactions database contains many other details like, quantity of sales, vehicle used for transporting, etc. Each record in this sales transactions database refers to a single sales transaction.

B. Time-stamped directed graphs

Using the sales database, we construct a directed edge-labeled multigraph called *sales flow graph*, denoted by $G_s = (V, E, \Phi)$, where V is the set of vertices(each vertex is labeled by a dealer name), E is the set of directed edges and Φ is the function that associates a 2-tuple for each edge, where first element of the tuple is the time of sales of this transaction and second element is the value of sales of this transaction. Figure 3 shows the sales flow graph of Table I.

Let us define a few notations. Note that each edge in the graph has two parameters, one is the time of sale and the other is the value of sales. The *end time* of a cycle is defined as the time of most recent transaction among all the transactions corresponding to edges in the given cycle. The end time of the cycle *ABC* in Figure 3 is *2017/01/04/10:30* of the edge

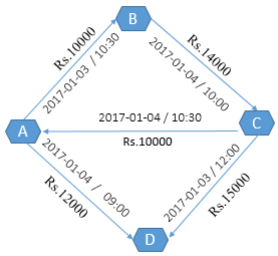

Figure 3. Sales Flow Graph

CA. The *start time* of a cycle is defined as the time of least recent transaction among all the transactions corresponding to edges in the given cycle. The start time of the cycle *ABC* in figure 3 is *2017/01/03/10:30* corresponding to the edge AB. The *span* of a cycle is defined as the difference between *end time* and *start time*. The span of the cycle *ABC* in figure 3 is *2017/01/04/10:30 - 2017/01/03/10:30 = 24:00*. The *bottle neck value* of a path is defined as the time of least recent transactions among all transactions corresponding to edges in the given path. The *bottleneck* value of the path *BCAD* in Figure 3 is *2017/01/04/9:00*.

C. Overview of the solution

From the in-depth research by taxation authorities, it is observed that dealers do illegitimate sales transactions among themselves in a very short period of time. Any illegitimate cycle should satisfy the following conditions since for any dealer the net tax liability due to illegitimate transactions should be zero.

- For a dealer in any illegitimate cycle the tax he pays on illegitimate purchase transaction should be same as the tax he collects on illegitimate sales transaction. This makes net tax liability due to illegitimate transactions zero.
- The quantity of goods involved in an illegitimate purchase should be the same as the quantity of goods of illegitimate sale. Otherwise, it is easy for tax authorities to detect these illegitimate transactions.

To meet the above two conditions the price of one unit of goods should be the same in every illegitimate transaction. Since the market price of goods can vary from day to day dealers perform all illegitimate transactions in a very short period of time. This means that the *span* of any illegitimate sales cycle is very small.

Our objective is to remove all illegitimate cycles from the sales flow graph. Then the remaining graph is a directed acyclic graph(DAG). Note that the resultant DAG contains all suspicious transactions. This make the fraud detection process easier and we can perform an in-depth analysis on suspicious transactions to identify tax evaders.

 ISBN: 978-1-61208-603-3

In the proposed algorithm, we remove illegitimate cycles from the *sales flow graph* one after the other in increasing order of their *end time*. If two cycles have the same *end time*, then we remove them in the increasing order of their *span*. Following is a brief sketch of the algorithm.

1) Select a cycle C in sales flow graph G_s with the following conditions:
   - *Condition 1: end time* of C is minimum among all the cycles in G_s
   - *Condition 2:* With respect to the condition one, *span* of C is minimum
2) Let m be the minimum of the values of sales of all the edges in C. Subtract m from values of sales of all edges in C.
3) Remove any edge from C whose value of sales becomes zero.
4) Repeat steps one to three, as long as G_s contains a cycle.

IV. ALGORITHM

A. Bottleneck Edge Computation Algorithm

The objective of Algorithm 1 is to find a path between the given source and destination vertices in a sales flow graph such that the *bottleneck* value is maximized.

Time Complexity: Assuming that there are V vertices in the graph, the queue Q may contain $O(V)$ vertices. Every time the *while* loop executes, one vertex is extracted from the queue Q and added to S. This operation takes $O(logV)$ time assuming the heap implementation of priority queues. So the total time required to execute the *while* loop itself is $O(VlogV)$.

Assuming that there are E edges in the graph,the inner *for* loop will be executed E times. Each iteration takes $O(logV)$ time assuming the heap implementation of priority queues. So, time complexity of this algorithm is $O(E * logV)$.

Proof Of Correctness:

Theorem 4.1: Let S be the set of visited vertices.

- For every vertex $v \in S$, bt[v] is the maximum among bottleneck values of all the paths from vertex a to vertex v.
- For every vertex $v \in V(G_s) - S$, if we consider only those paths from vertex a to vertex v where predecessor of v is in S, bt[v] is the maximum among bottleneck values of all such paths.

Proof Proof by induction on $|S|$.

Base case ($|S|$= 1): the only time $|S|$= 1 is when $S = \{a\}$. In this case

- bt[a] $= \infty$.
- For every $v \in V(G_s) - S$, which is adjacent to a, bt[v] is the time of sales of the edge av.

Inductive hypothesis: Let u be the last vertex moved from Q to S. Assume that theorem is true for the set $S' = S - \{u\}$.

Data: sales flow graph G_s and two vertices a, b in G_s
Result: A path from a to b such that *bottle neck value* is maximum, and the corresponding bottle neck value.

```
# Set the bottleneck value of source vertex a to ∞;
bt[a] = ∞;

# Set the predecessor of source vertex to NULL;
pred[a] = NULL;

# Set the bottleneck values of all other vertices to −∞;
For (all v ∈ V(G_s) − a){
bt[v] = −∞;
pred[v] = NULL}

# S be the set of visited vertices, initially it is empty;
S = ∅;

# Q be the queue of unvisited vertices, initially it contains all vertices;
Q = V(G_s);

while Q ≠ ∅ do
    Let u be a vertex in Q with maximum bottleneck value ;

    # Move the vertex u from queue Q to processed set S;
    S = S ∪ {u} ;
    Q = Q − u;

    For (all v ∈ neighbours(u))
    {
     # Note that there can be more than one edge from vertex u to vertex v. We take the edge whose time of sales is most recent;
    # If the bottleneck value of vertex v is less than the minimum among the bottleneck value of vertex u and the time of sales of the edge uv, then replace the bottleneck value of vertex v;

    if ((min(bt[u],time of sales of the edge uv) > bt[v])
     then
        bt[v]= min(bt[u],time of sales of the edge uv);
        pred[v]=u;
    end
    }
end
```

Algorithm 1: Bottleneck Edge Computation Algorithm

Lemma 4.2: bt[u] is the maximum among bottleneck values of all paths from a to u

Proof Let $a, s_1, s_2, \ldots, s_k, q_1, \ldots, u$ be a path from a to u in G_s such that bottleneck value is maximum. Assume that $\{a, s_1, s_2, \ldots, , s_k\} \subseteq S'$ and $q_1 \in V(G_s) - S'$. Note that the bottleneck value of path $a, s_1, s_2, \ldots, s_k, q_1, \ldots, u$ is less than or equal to the bottleneck value of the path $a, s_1, s_2, \ldots, s_k, q_1$, which is less than or equal to $bt[q_1]$. Since we selected u such that $bt[u] \geq bt[v]$, for all $v \in V(G_s) - S'$, $bt[q_1] \leq bt[u]$. So, the bottleneck value of the path $a, s_1, s_2, \ldots, s_k, q_1, \ldots, u$ is less than equal to $bt[u]$. Since $a, s_1, \ldots, s_k, q_1, \ldots, u$ is a path from a to u in G_s such that bottleneck value is maximum, bottleneck value of the path $a, s_1, s_2, \ldots, s_k, q_1, \ldots, u$ is equal to $bt[u]$

Lemma 4.3: For every vertex $v \in V(G_s) - S$, if we consider only those paths from vertex a to vertex v where predecessor of v is in S, $bt[v]$ is the maximum among bottleneck values of all such paths.

Proof The proof is based on the fact that every vertex $v \in S$, bt[v] is the maximum among bottleneck values all path from a to v.

This proves the theorem.

B. Minimum Span Cycles Removal Algorithm

The objective of this algorithm is to remove illegitimate cycle in sales flow graph. This algorithm uses the Algorithm 1 to find a cycle with minimum span.

Data: sales flow graph G_s
Result: Forest G_t, which is obtained by removing all cycles in G_s

G_t = Edgeless graph whose vertex set is $V(G_s)$;
Let $l_1, l_2, \ldots, l_m$ be a sequence of all edges in G_s ordered by non decreasing order of time of sales;
for *i = 1...m* **do**
  insert the edge l_i in the graph G_t;
  while *(G_t contains cycle)* **do**
    Assume that the edge l_i is from vertex b to vertex a in G_t;
    Let $a, v_1, \ldots, v_k, b$ be a path from a to b in G_t such that the bottleneck value is maximum;
    Let cycle $C = a, v_1, \ldots, v_k, b$;
    Let p be the minimum among price of sales of all edges in C;
    Subtract p from price of sales of all edges in C;
    Remove all edge from G_t whose price of sales is zero;
  end
end

Algorithm 2: Cycle Removal Algorithm

V. CASE STUDY

We had taken up a synthetic data set in which few dealers are doing heavy circular trading among themselves. Figure 4 shows the details of this circular trade. The value of each edge represents the value of sales in lakhs (one *lakh* is equal to 0.1 million), and note that it is the sum value of several transactions between two particular dealers directed from one to the other.

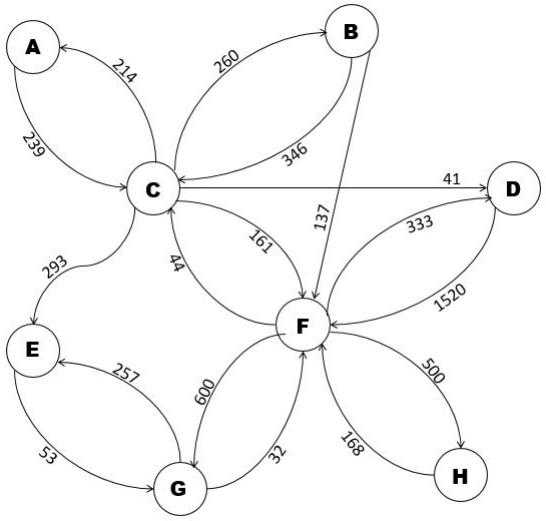

Figure 4. Sales flow graph

As given in Figure 4, there are numerous cycles among this group of dealers and these cycles are considered as undesirable patterns by domain experts. Cycles are undesired in these transactions since a cycle indicates the buying of the same goods by a dealer which (s)he has previously sold. According to domain experts, this kind of flow of goods is not valid for the particular commodity which these people are trading among themselves. These cycles indicate that there is a great chance for tax evasion. As mentioned in Sub-section C of Section III, the dealers involved in a circular trade need to fabricate these cycles in a short duration of time. We used the proposed algorithm to delete the illegitimate cycles. Figure 5 shows the directed acyclic graph obtained after deleting all cycles from the graph given in Figure 4. The novelty of our technique over the others in deleting cycles is that since we delete cycles that forms in a short duration of time, with high accuracy we delete the illegitimate edges used to form these cycles. This would not be the case if we had deleted the cycles using other techniques. Recall that the objective of the illegitimate transactions were to hide these suspicious transactions, and the directed acyclic graph (DAG) given in Figure 5 contains the suspicious transactions. Analysis of this DAG gives significant information which can be used by the taxation authorities for conducting further investigations.

By studying the purchases/sales they did it is observed that they should have paid huge tax, buy they paid only negligible amount of tax by showing fictitious exports and inter state

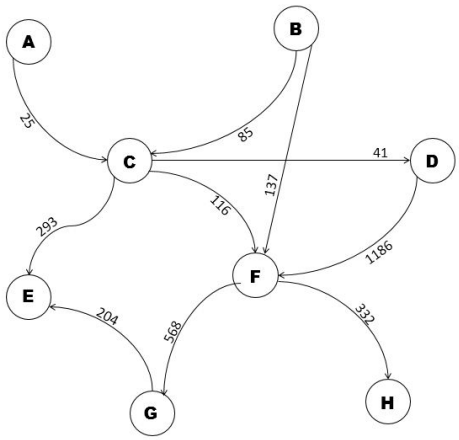

Figure 5. Suspicious transactions

sales. To hide this fictitious sales they created a web of sales and purchase transactions among themselves.

VI. Conclusion

In this paper, we stated and formalized an important practical problem in the commercial taxation system called *circular trading*. *Circular trading* is the method in which a set of traders do heavy illegitimate sales/purchase transactions in a circular manner among themselves in a short duration without any *value-add*. The problem of removing these type of cycles is important because, tax authorities can easily detect suspicious transactions once the cycles are removed. Here, we proposed an algorithm to remove such type of illegitimate cycles. As further work, we are investigating to find more effective ways of removing cycles.

VII. Acknowledgement

We would like to express our deep thanks towards the government of Telangana, India, for allowing us to use the Commercial Taxes Data set and giving us constant encouragement and financial support.

References

[1] E. W. Dijkstra and S. S. Scholten. “Termination detection for diffusing computations”. In: *Information Processing Letter* 11 (1980), pp. 1–4.

[2] M. Franke, B. Hoser, and J. Schröder. “On the analysis of irregular stock market trading behavior”. In: *Data Analysis, Machine Learning and Applications*. ISBN: 978-3-540-78239-1, URL: https://link.springer.com/chapter/10.1007/978-3-540-78246-9_42. Springer, Jan. 2007, pp. 355–362.

[3] Alan Schenk and Oliver Oldman, eds. *Value Added Tax: A Comparative Approach*. ISBN: 978-1107617629. Cambridge University Press, Jan. 2007.

[4] G. K. Palshikar and M.M. Apte. “Collusion set detection using graph clustering”. In: *Data Mining and Knowledge Discovery*. ISSN: 1384-5810, URL: https://link.springer.com/article/10.1007/s10618-007-0076-8. Springer, Apr. 2008, pp. 135–164.

[5] N. Md. Islam et al. “An approach to improve collusion set detection using MCL algorithm”. In: *Computers and Information Technology*. ISBN: 978-1-4244-6284-1, URL: http://ieeexplore.ieee.org/abstract/document/5407133/. IEEE, Dec. 2009, pp. 237–242.

[6] B. Haeupler et al. “ Incremental cycle detection, topological ordering, and strong component maintenance”. In: *ACM Transactions on Algorithms* vol. 8, no. 1 (2012), pp. 1–33.

[7] J. Wang, S. Zhou, and J. Guan. “Detecting potential collusive cliques in futures markets based on trading behaviors from real data”. In: *Neurocomputing* 92 (2012), pp. 44–53.

[8] K. Golmohammadi, O.R. Zaiane, and D. Díaz. “ Detecting stock market manipulation using supervised learning algorithms”. In: *Data Science and Advanced Analytics*. ISBN: 978-1-4799-6991-3 , URL: http://ieeexplore.ieee.org/document/7058109/. IEEE, Nov. 2014, pp. 435–441.

[9] Hsu KW.and Pathak N. et al. “Data Mining Based Tax Audit Selection: A Case Study of a Pilot Project at the Minnesota Department of Revenue”. In: *Real World Data Mining Applications*. ISBN: 978-3-319-07811-3, URL: https://doi.org/10.1007/978-3-319-07812-0_12/. Springer, Nov. 2014, pp. 221–245.

[10] B. Baesens, V.V. Vlasselaer, and W. Verbeke, eds. *Fraud Analytics Using Descriptive, Predictive, and Social Network Techniques: A Guide to Data Science for Fraud Detection*. ISBN: 978-1-119-13312-4. Wiley, Aug. 2015.

[11] Zhenisbek Assylbekov et al. “Detecting Value-Added Tax Evasion by Business Entities of Kazakhstan”. In: *Intelligent Decision Technologies 2016*. ISBN: 978-3-319-39629-3, URL: https://doi.org/10.1007/978-3-319-39630-9_4/. Springer, 2016, pp. 37–49.

[12] Michael A. Bender, Jeremy T. Fineman, and Robert E. Tarjan Seth Gilbert. “A New Approach to Incremental Cycle Detection and Related Problems”. In: *ACM Transactions on Algorithms* Volume 12 Issue 2 (2016), p. 22.

[13] S. Dani. “A Research Paper on an Impact of Goods and Service Tax(GST) on Indian Economy”. In: *Business and Economics Journal* 7 (2016). ISSN: 2151-6219, p. 264.

[14] E. Vicente, A. Mateos, and A. Jiménez-Martín. “Detecting stock market manipulation using supervised learning algorithms”. In: *Modeling Decisions for Artificial Intelligence*. ISBN: 978-3-319-45655-3, URL: https://link.springer.com/chapter/10.1007/978-3-319-45656-0_17. Springer, Sept. 2016, pp. 205–216.

[15] Huanqing Cui et al. “A Multi-Threading Algorithm to Detect and RemoveCycles in Vertex- and Arc-Weighted Digraph”. In: *Algorithms 2017* 10 (2017), p. 115.

 ISBN: 978-1-61208-603-3