Detecção de Fraudes em Redes de Transações Financeiras Utilizando Grafos

Andre Cota[1], Gustavo Prehl[2], Jhonatan Gutemberg[3], Natã Torres[4], Vinicius Salles[5]

[1]Pontifícia Universidade Católica de Minas Gerais (PUC Minas)
Belo Horizonte – MG – Brasil

[2]Instituto de Ciências Exatas e Informática (ICEI)

1202002[1], 1361441[2], 1464198[3], 1140500[4], 1444802[5]@sga.pucminas.br

Abstract. *Financial fraud represents a growing challenge, causing significant losses. Traditional detection methods often fail to identify complex fraudulent patterns involving multiple entities. This paper introduces FraudeDetector, a tool that models financial transactions as a graph network to identify suspicious activities. By representing clients and merchants as nodes and transactions as edges, the tool employs a suite of algorithms to detect anomalies and illicit patterns. These algorithms include cycle detection, statistical outlier analysis, identification of central hubs, community detection (clusters), and analysis of recurring transaction pairs. The proposed approach allows for a systemic view of the transaction network, revealing complex structures indicative of fraud that would be difficult to perceive with conventional methods.*

Resumo. *A fraude financeira representa um desafio crescente, gerando perdas significativas. Métodos tradicionais de detecção frequentemente falham em identificar padrões fraudulentos complexos que envolvem múltiplas entidades. Este artigo apresenta o FraudeDetector, uma ferramenta que modela transações financeiras como uma rede de grafos para identificar atividades suspeitas. Ao representar clientes e comerciantes como nós e transações como arestas, a ferramenta emprega um conjunto de algoritmos para detectar anomalias e padrões ilícitos. Tais algoritmos incluem detecção de ciclos, análise de outliers estatísticos, identificação de hubs centralizadores, detecção de comunidades (clusters) e análise de pares de transações recorrentes. A abordagem proposta permite uma visão sistêmica da rede de transações, revelando estruturas complexas e indicativas de fraude, difíceis de serem percebidas por métodos convencionais.*

1. Introdução

As transações financeiras digitais tornaram-se onipresentes, mas trouxeram consigo um aumento expressivo nas tentativas de fraude, como lavagem de dinheiro, contas ”laranja” e ataques automatizados. As abordagens convencionais de detecção, que analisam transações de forma isolada, são insuficientes para descobrir esquemas fraudulentos coordenados e complexos.

A teoria dos grafos oferece um paradigma poderoso para superar essa limitação. Ao modelar o ecossistema financeiro como uma rede — onde contas são nós e transações

são arestas —, é possível analisar as conexões e o fluxo de valores de maneira holística. Esta representação permite a aplicação de algoritmos capazes de identificar padrões estruturais e comportamentais que indicam atividades ilícitas.

Neste contexto, este artigo apresenta a ferramenta `FraudeDetector`, um sistema desenvolvido em Python para processamento e análise de dados de transações financeiras com o uso de grafos. O objetivo é fornecer um conjunto de metodologias robustas para a identificação de comportamentos suspeitos em redes de transações.

Análise foi conduzida utilizando um conjunto de dados sintéticos de transações financeiras, projetado especificamente para a detecção de fraudes, nos baseamos em [Lopez-Rojas, Edgar A. 2018] e [Young Lambert Officia 2024].

2. Modelagem de Transações com Grafos

Métodos de análise tradicionais, que geralmente avaliam transações de forma tabular e isolada, possuem limitações intrínsecas na detecção de esquemas fraudulentos sofisticados. Esses esquemas raramente envolvem uma única transação, mas sim uma rede complexa de interações projetada para ofuscar a origem e o destino dos fundos. É neste ponto que o paradigma dos grafos se torna uma ferramenta analítica superior.

Ao modelar um sistema de transações financeiras como um grafo direcionado $G = (V, E)$, onde as entidades (clientes, comerciantes) são os **nós (vértices)** V e as transações entre elas são as **arestas** E, passamos a ter uma visão holística da rede. Essa representação transforma listas de transações em uma estrutura conectada, permitindo analisar a topologia e o fluxo de informações de maneiras que seriam impossíveis de outra forma. A principal vantagem dessa abordagem é a capacidade de revelar padrões relacionais e estruturais, tais como:

- **Movimentação Circular de Fundos:** Identificação de ciclos onde o dinheiro sai de uma conta e, após passar por múltiplos intermediários, retorna a ela. Este é um comportamento clássico em esquemas de lavagem de dinheiro.
- **Contas Centralizadoras (Hubs):** Detecção de nós que recebem fundos de muitas fontes distintas e os distribuem para muitos outros destinos (ou o inverso). Tais contas podem atuar como ”laranjas” ou pontos de concentração e pulverização de valores ilícitos.
- **Grupos Coordenados (Clusters):** Localização de comunidades de nós densamente conectados entre si, mas com poucas conexões com o resto da rede. Isso pode indicar quadrilhas ou grupos de contas agindo em conluio.

Dada a capacidade inerente dos grafos de expor essas estruturas ocultas, esta abordagem foi selecionada como a fundação para a ferramenta proposta neste artigo.

3. A Ferramenta FraudeDetector

A ferramenta foi projetada com uma estrutura modular para facilitar a manutenção e a extensibilidade. A seguir, detalha-se sua arquitetura e o modelo de dados utilizado.

3.1. Arquitetura do Projeto

O projeto está organizado nos seguintes diretórios principais:

- `domain/`: Contém as abstrações principais do modelo, como a classe que representa o grafo, encapsulando um objeto da biblioteca NetworkX.
- `services/`: Responsável pelos serviços de entrada e saída, notadamente a importação de dados de arquivos CSV.
- `algorithms/`: Contém a implementação de todos os algoritmos de análise e detecção de fraudes.
- `interface/`: Implementa a interface com o usuário via web (Flask), incluindo as rotas, as visualizações e a geração de grafos em SVG.
- `data/`: Repositório para os conjuntos de dados de entrada.
- `tests/`: Módulo dedicado aos scripts de teste para validação das funcionalidades.

3.2. Modelo de Dados e Importação

O processo inicia-se com a importação de um arquivo CSV de transações através da função `importar_transacoes_csv()` do módulo de serviços. Clientes e comerciantes são mapeados como nós, e cada transação é representada como uma aresta direcionada do cliente para o comerciante. Atributos como valor (‘amount‘) e um indicador de fraude (‘fraud‘) são armazenados nas arestas, permitindo análises ponderadas e a posterior validação dos algoritmos. O uso da biblioteca Pandas otimiza a leitura e o processamento inicial dos dados.

4. Metodologias de Detecção

A ferramenta implementa diversas técnicas de análise de grafos para identificar diferentes tipos de atividades suspeitas.

4.1. Detecção de Padrões Estruturais

4.1.1. Detecção de Ciclos

Transações circulares são um forte indício de lavagem de dinheiro. As funções `encontrar_ciclos()` e `encontrar_ciclos_fraude()` buscam por esses padrões, utilizando a implementação da biblioteca NetworkX e permitindo configurar o tamanho dos ciclos a serem investigados.

4.1.2. Pares Recorrentes

A função `pares_recorrentes()` identifica pares de nós (origem, destino) que realizam um volume de transações entre si acima de um limiar, o que pode indicar relações suspeitas ou tentativas de inflar artificialmente o volume de transações.

4.2. Detecção de Anomalias e Outliers

4.2.1. Volume Anômalo de Transações

As funções `nos_com_muitos_envios()` e `nos_com_muitos_recebimentos()` identificam nós cujo grau de saída ou de entrada, respectivamente, excede um limite predefinido, apontando para contas com atividade atípica.

4.2.2. Concentração Temporal de Transações (Burst)

A função `nos_com_burst_transacoes()` detecta nós que realizam um número elevado de transações em uma curta janela de tempo. Esse padrão pode sinalizar ataques automatizados ou fracionamento de valores.

4.2.3. Transações com Valores Outliers

São implementadas duas abordagens para detectar transações com valores anômalos. A primeira, `transacoes_valor_alto()`, utiliza um valor monetário absoluto. A segunda, `transacoes_outliers()`, emprega uma abordagem estatística, identificando transações cujo valor excede a média mais um múltiplo do desvio padrão (ex: $\mu + 3\sigma$).

4.3. Análise Estrutural da Rede

4.3.1. Identificação de Nós Centrais (Hubs)

As funções `hubs_envio()` e `hubs_recebimento()` identificam nós que interagem com um número elevado de destinos ou origens únicas, respectivamente. Tais nós são candidatos a contas de lavagem ou pontos de redistribuição de fundos.

4.3.2. Ranking de Atividade

Para priorizar investigações, a função `ranking_nos_ativos()` classifica os nós com base no número total de transações, permitindo focar a análise nos participantes mais ativos da rede.

4.3.3. Detecção de Clusters (Comunidades)

A função `encontrar_clusters()` utiliza o algoritmo de Kosaraju para encontrar Componentes Fortemente Conectados (SCCs). Um SCC representa um grupo de nós onde cada um pode alcançar qualquer outro nó do mesmo grupo, sugerindo uma possível coordenação entre eles.

5. Interface e Funcionalidades do FraudeDetector

Para demonstrar todos os artefatos produzidos, esta seção apresenta as principais telas da interface web da aplicação, que permitem ao usuário final interagir com os algoritmos e visualizar os resultados. Cada funcionalidade gera uma representação visual do subgrafo de interesse, além de apresentar os dados em formato tabular para facilitar a análise.

5.1. Detecção de Padrões Estruturais

Este grupo de funcionalidades foca na topologia da rede para encontrar esquemas de fraude coordenados.

5.1.1. Detecção de Ciclos

A tela de detecção de ciclos (Figura 1) é fundamental para identificar padrões de lavagem de dinheiro, destacando os nós e as transações que formam uma rota circular de fundos.

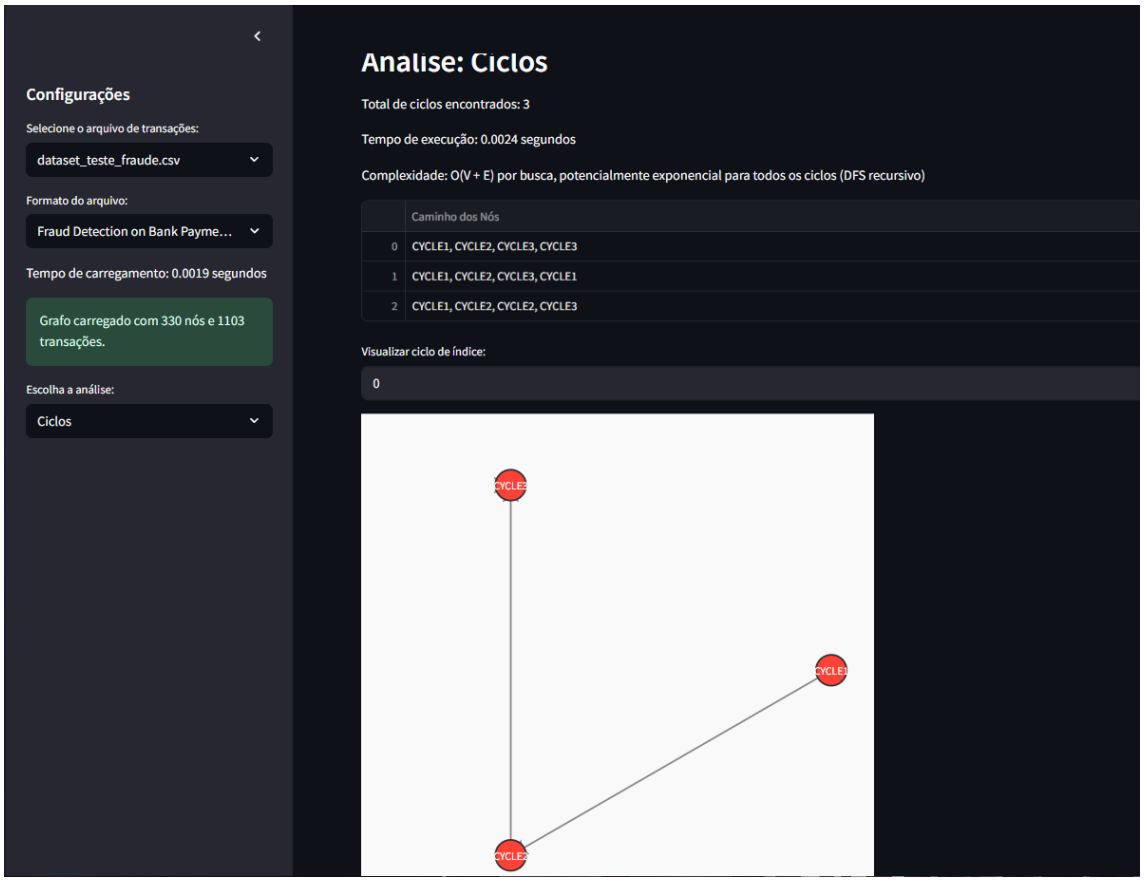

Figure 1. Tela de visualização de ciclos fraudulentos.

5.1.2. Detecção de Clusters (Comunidades)

A Figura 2 demonstra a saída da detecção de clusters (Componentes Fortemente Conectados). Grupos de nós densamente interconectados são exibidos, sugerindo a existência de redes que podem estar agindo em conluio.

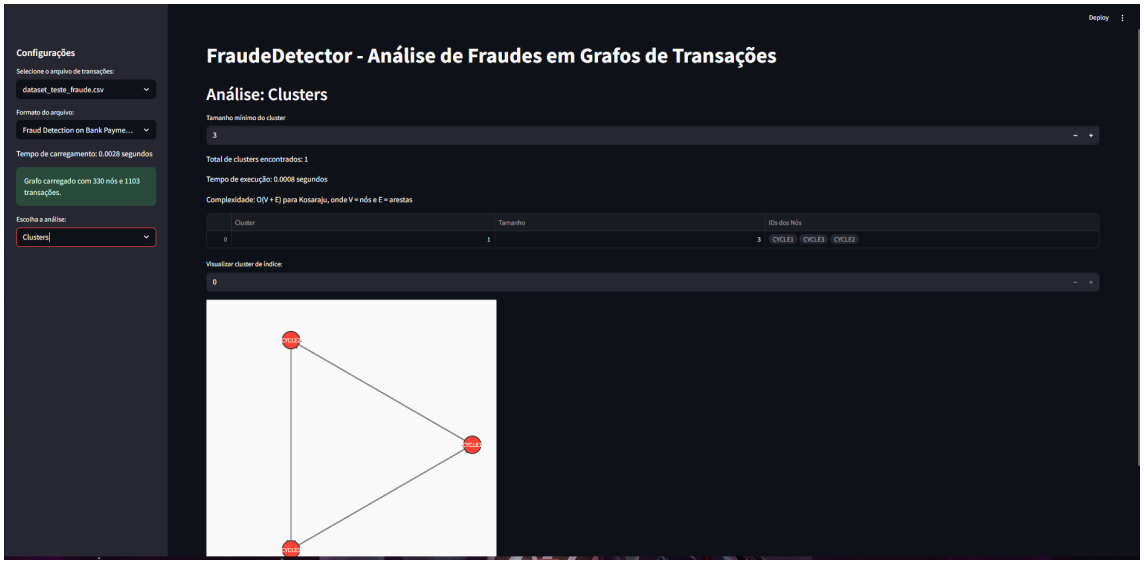

Figure 2. Exibição de um cluster identificado na rede de transações.

5.2. Análise de Centralidade e Volume

Este conjunto de análises identifica nós que possuem um comportamento atípico em termos de volume de transações ou centralidade na rede.

5.2.1. Análise de Hubs

A identificação de hubs (Figura 3) é crucial para encontrar contas ”laranja”. A tela permite analisar tanto hubs de recebimento (muitas entradas) quanto de envio (muitas saídas), destacando o nó centralizador e suas conexões.

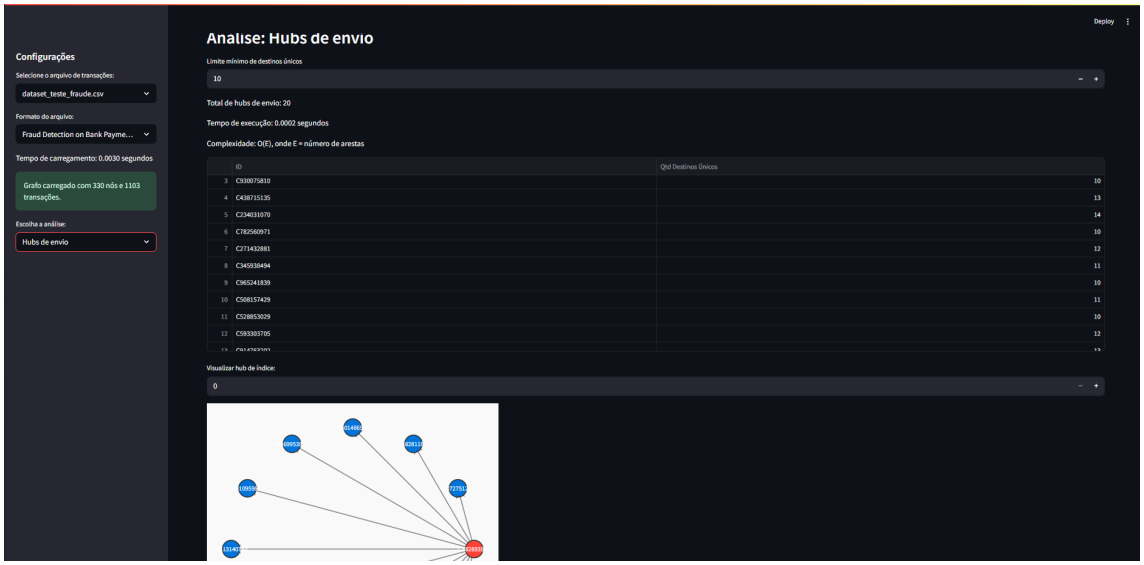

Figure 3. Identificação de um Hub de Recebimento e suas conexões.

5.2.2. Nós com Alto Volume de Transações

De forma similar aos hubs, esta funcionalidade (Figura 4) identifica nós que ultrapassam um limiar de transações enviadas ou recebidas, independentemente do número de parceiros distintos. É útil para detectar contas com atividade anormalmente alta.

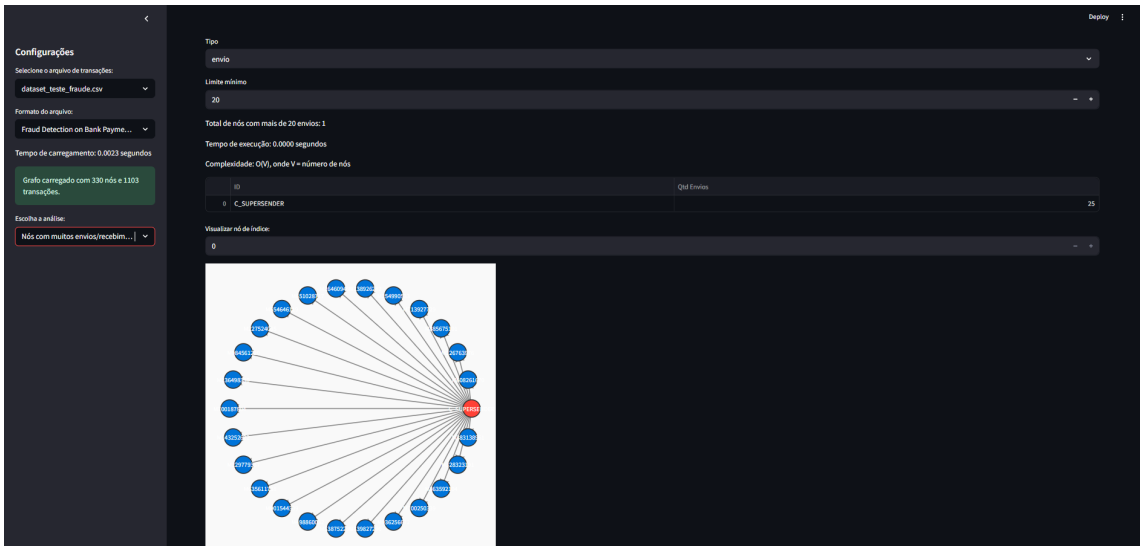

Figure 4. Tela de análise para nós com alto volume de envios ou recebimentos.

5.2.3. Ranking de Nós Ativos

Para direcionar a investigação, a tela de ranking (Figura 5) lista os nós mais ativos da rede, seja por número de transações enviadas, recebidas ou pelo total. Esta tela não costuma gerar um grafo, mas sim uma tabela ordenada para análise prioritária.

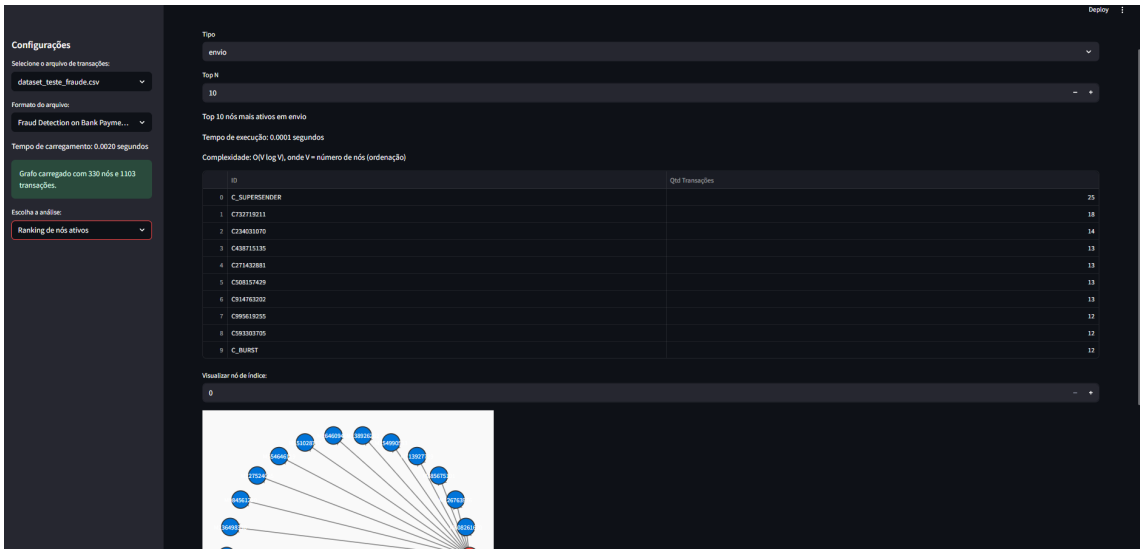

Figure 5. Interface exibindo o ranking de nós mais ativos da rede.

5.3. Análise de Comportamento e Anomalias

Este grupo de funcionalidades foca em padrões de transações que destoam do comportamento esperado, seja em valor, frequência ou temporalidade.

5.3.1. Análise de Valores de Transação

A ferramenta oferece duas abordagens para valores: outliers estatísticos (Figura 6), que identifica transações que destoam da média ($\mu + N\sigma$), e transações de valor absoluto alto (Figura 7), que sinaliza qualquer transação acima de um limiar monetário pré-definido.

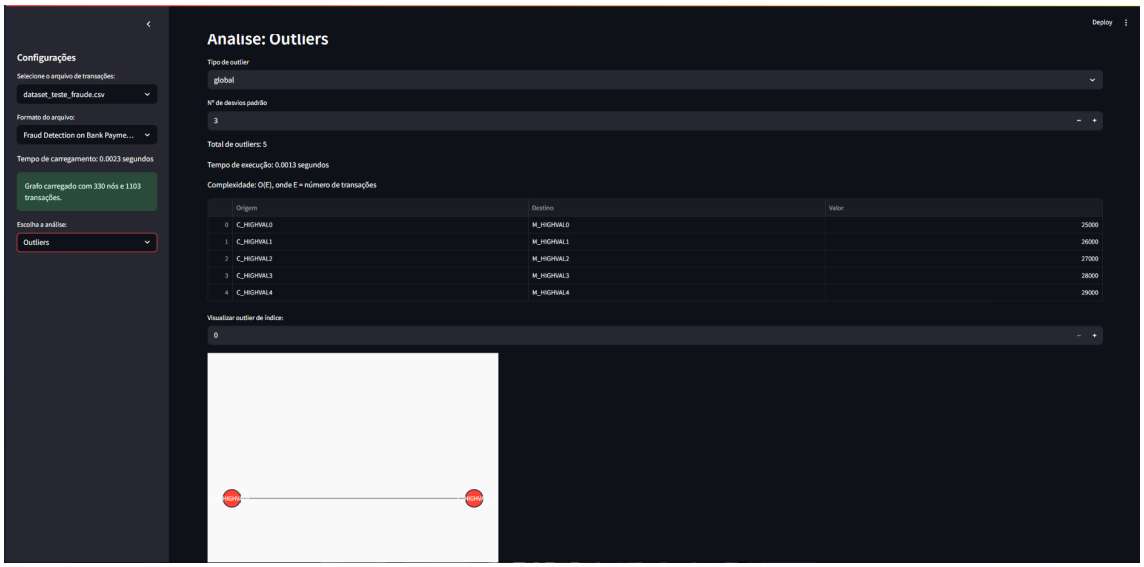

Figure 6. Visualização de uma transação outlier (anomalia estatística).

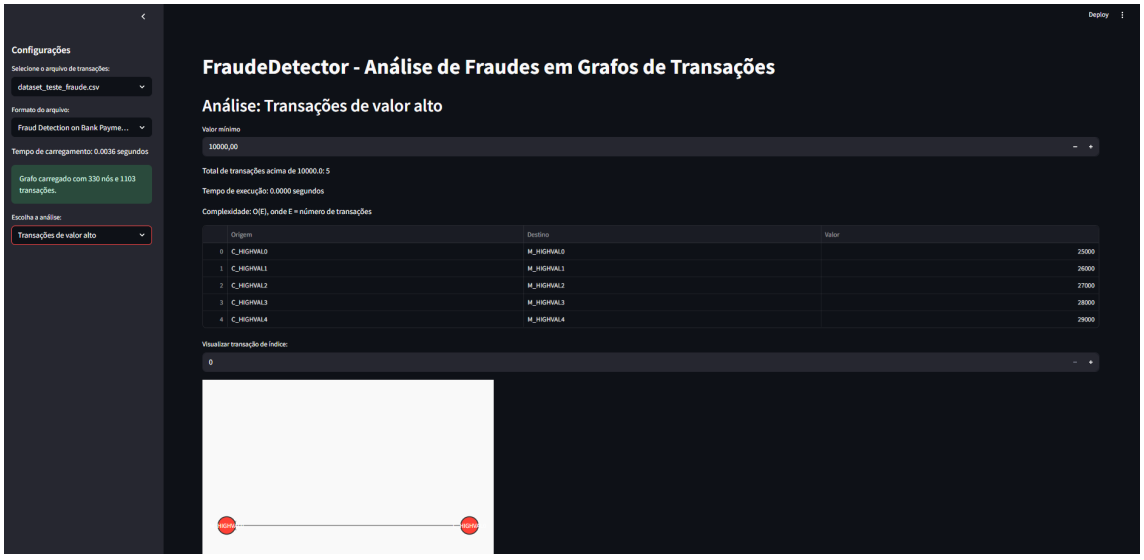

Figure 7. Detecção de uma transação com valor absoluto acima do limiar.

5.3.2. Análise de Pares Recorrentes

A Figura 8 exibe o resultado da análise de pares recorrentes, destacando duas entidades que transacionaram entre si um número de vezes acima do normal, o que pode indicar tentativas de inflar volumes ou outras atividades suspeitas.

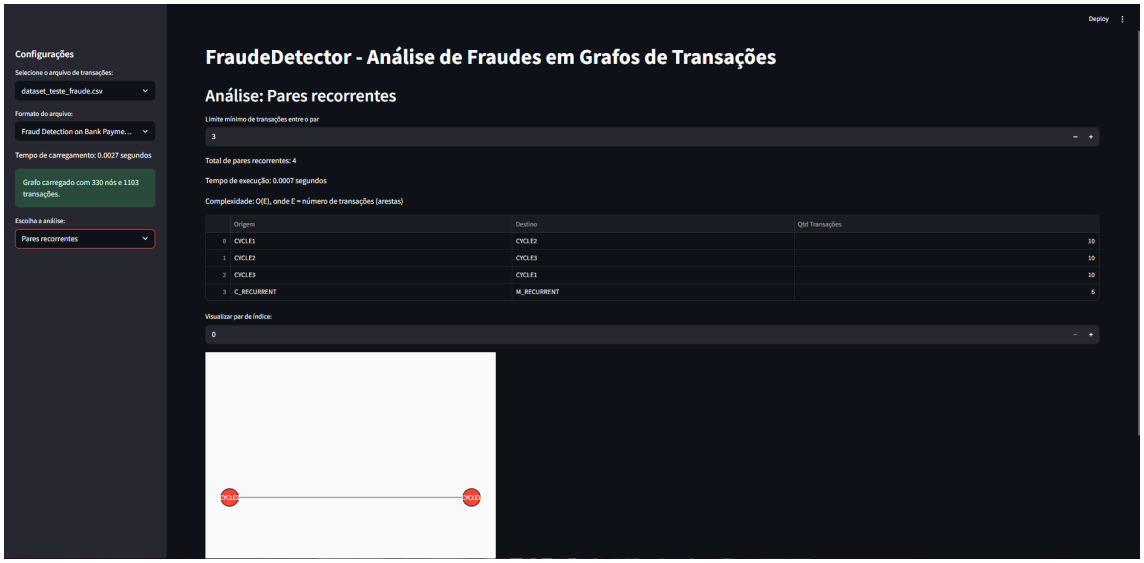

Figure 8. Visualização de um par de nós com transações recorrentes.

5.3.3. Detecção de Bursts Temporais

Esta funcionalidade (Figura 9) identifica nós que realizaram um grande número de transações em uma janela de tempo muito curta. Esse padrão é um forte indicativo de ataques automatizados, uso de scripts ou fracionamento de valores.

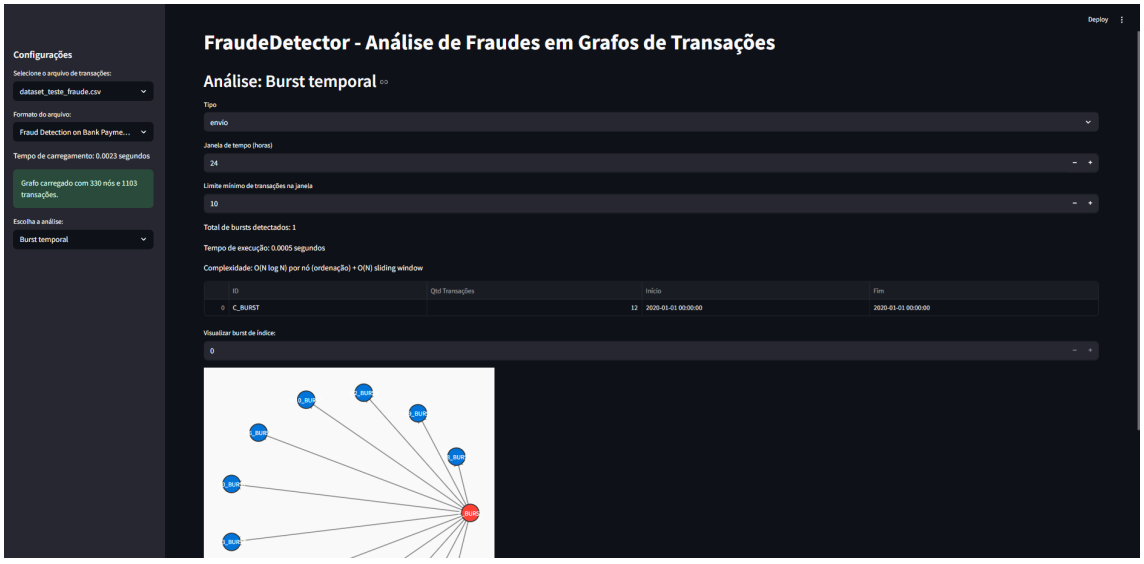

Figure 9. Exibição de um nó que apresentou um burst de transações.

6. Resultados Experimentais

Para validar a robustez e a eficácia do `FraudeDetector`, a ferramenta foi aplicada a quatro conjuntos de dados distintos, com características variadas de tamanho e distribuição de fraudes. Esta abordagem permite avaliar a performance dos algoritmos em cenários diversos.

6.1. Conjuntos de Dados Utilizados

Os datasets selecionados para a análise representam diferentes escalas de redes de transações. Suas principais características estão sumarizadas na Tabela 1.

Table 1. Características dos conjuntos de dados utilizados nos experimentos.

Dataset	Transações	Fraudes	% Fraude
`paysim`	6.362.620	8.213	0,13%
`bs140513_032310`	594.643	7.200	1,21%
`dataset_teste_fraude`	1.103	5	0,45%
`fraud_detection`	1.000	50	5,00%

O dataset `paysim` é o maior e mais esparso em termos de fraude, servindo como um benchmark de estresse e escalabilidade. Em contraste, o `fraud_detection` possui a maior densidade de fraudes, sendo ideal para testar a sensibilidade dos algoritmos a padrões fraudulentos mais evidentes.

6.2. Análise Comparativa dos Padrões Detectados

A aplicação das metodologias de detecção nos quatro datasets revelou resultados marcadamente distintos, o que evidencia que cada conjunto de dados possui uma ”impressão digital” de fraude e comportamento única. A Tabela 2 apresenta um resumo dos principais achados. A análise a seguir interpreta esses resultados.

Table 2. Resultados comparativos da detecção de padrões nos datasets.

Metodologia de Detecção	paysim	bs140513	teste_fraude	fraud_detection
Ciclos	0	0	3	0
Hubs de Envio ($>$10 trans.)	0	3.028	20	0
Hubs de Recebimento ($>$10 trans.)	146.629	49	56	0
Pares Recorrentes	0	17.181	4	15
Burst Envio ($\geq$10 em 24h)	0	2	1	0
Burst Recebimento ($\geq$10 em 24h)	1.592	1.970	0	0
Transações Outliers ($\mu + 3\sigma$)	44.945	3.188	5	18
Clusters (SCCs $\geq$2 nós)	0	0	1	0

A análise dos resultados da Tabela 2 permite extrair as seguintes observações:

- **Padrões Estruturais Fortes (Ciclos e Clusters):** Estes padrões, que representam fortes indícios de fraude coordenada, foram detectados exclusivamente no

dataset `dataset_teste_fraude`. A identificação de 3 ciclos e 1 cluster neste pequeno conjunto de dados valida a corretude dos algoritmos e sugere que este dataset foi especificamente construído para conter essas topologias complexas. A ausência total de ciclos nos demais datasets indica que a fraude presente neles (ou a estrutura geral da rede) não se baseia em movimentação circular de fundos.

- **Hubs de Envio e Recebimento:** A análise de hubs revela a natureza distinta das redes. O dataset `bs140513` apresenta um número significativo de Hubs de Envio (3.028), um padrão clássico de contas "laranja" ou "mulas", que recebem fundos e os pulverizam para diversos destinos. Em contraste, o `paysim` exibe um número massivo de Hubs de Recebimento (146.629), mas nenhum de envio. Este comportamento assimétrico provavelmente não indica fraude, mas sim uma característica do simulador, onde um vasto número de clientes ('C') paga um conjunto menor de comerciantes ('M'), que se tornam, naturalmente, hubs de recebimento.
- **Comportamento Transacional (Pares e Bursts):** A detecção de Pares Recorrentes foi mais proeminente no `bs140513` (17.181), o que é esperado em um cenário real com transações legítimas e repetitivas (como pagamentos de salários ou assinaturas). O resultado mais surpreendente é a ausência total de pares recorrentes no `paysim`, o que é um forte artefato de sua natureza sintética, indicando que cada par cliente-comerciante pode ter transacionado apenas uma vez. A detecção de "Bursts de Recebimento" em `paysim` e `bs140513` pode indicar momentos de pico de vendas para comerciantes, enquanto um "Burst de Envio" (detectado em `bs140513` e `teste_fraude`) é um sinal de alerta mais forte, sugerindo atividade automatizada ou fracionamento de saques.
- **A Universalidade das Transações Outliers:** A detecção de outliers de valor foi a única metodologia que identificou um volume significativo de anomalias em todos os quatro datasets. O `paysim` lidera com 44.945 casos, seguido pelo `bs140513` com 3.188. Isso demonstra que a análise estatística de valores é uma primeira linha de defesa fundamental e universalmente aplicável, pois consegue sinalizar transações suspeitas independentemente da existência de padrões de conexão complexos na rede.

Esses resultados mostram que não existe uma única "bala de prata". A detecção eficaz de fraudes requer um conjunto de ferramentas que possam ser aplicadas de forma complementar. A ausência de um padrão não significa ausência de fraude, mas sim que a fraude pode estar se manifestando de outra forma. A força do `FraudeDetector` reside em sua capacidade de aplicar múltiplas lentes de análise sobre a mesma rede de transações.

7. Avaliação de Desempenho

A performance computacional da ferramenta foi avaliada em um ambiente com as especificações: Processador Ryzen 5 5600, 32 GB RAM e Windows 11. O objetivo foi medir o tempo de execução das principais operações nos diferentes datasets para avaliar a escalabilidade da solução.

O dataset `paysim`, por ser o maior, serviu como principal benchmark de estresse. Os tempos de execução para este dataset estão detalhados na Tabela 3.

Table 3. Tempos de execução das operações no dataset `paysim`.

Operação em `paysim`	Tempo (s)
Importação e Construção do Grafo	112,78
Detecção de Ciclos	239,77
Hubs de Envio ($>$10 trans.)	6,41
Hubs de Recebimento ($>$10 trans.)	6,41
Pares Recorrentes	378,62
Burst Envio ($\geq$10 em 24h)	6,41
Burst Recebimento ($\geq$10 em 24h)	6,41
Detecção de Outliers	8,50
Detecção de Clusters ($\geq$2 nós)	378,62

Conforme esperado, a construção do grafo e as análises que requerem a travessia de grande parte da rede (Ciclos, Pares Recorrentes, Clusters) são as mais custosas computacionalmente. Ainda assim, os tempos se mantêm em uma ordem de magnitude viável para análises offline em um dataset com mais de 6 milhões de transações.

Para fins de comparação, a Tabela 4 mostra o tempo de construção do grafo para todos os datasets.

Table 4. Tempo comparativo de importação e construção do grafo.

Dataset	Nº de Arestas	Tempo de Carga (s)
`paysim`	6.362.620	112,78
`bs140513_032310`	594.643	3.6508
`dataset_teste_fraude`	1.103	0.0019
`fraud_detection`	1.000	0.0067

A análise de desempenho demonstra que a solução é capaz de lidar com volumes de dados consideráveis, com tempos de resposta que variam conforme a complexidade do algoritmo e o tamanho do grafo, o que está alinhado com as expectativas teóricas da complexidade de algoritmos em grafos.

8. Validação e Testes

A corretude das funcionalidades foi assegurada por um conjunto de testes unitários e de integração, localizados no diretório `tests/`. Foram criados dois módulos principais de teste:

- `test_importacao.py`: Valida o processo de leitura do CSV e a correta construção do grafo, verificando se o número de nós e arestas corresponde ao esperado e se os atributos das arestas (como o valor) são carregados corretamente.
- `test_algoritmos.py`: Testa cada função de detecção de fraude contra pequenos grafos de exemplo com resultados conhecidos. Isso garante que os algoritmos identificam corretamente ciclos, hubs e outliers em cenários controlados.

A execução da suíte de testes pode ser feita via linha de comando, garantindo que novas modificações no código não quebrem funcionalidades existentes.

9. Conclusão e Trabalhos Futuros

O `FraudeDetector` demonstra o potencial da análise de grafos como uma abordagem eficaz para a detecção de fraudes financeiras. Ao modelar transações como uma rede, a ferramenta viabiliza a aplicação de um conjunto diversificado de algoritmos capazes de revelar padrões complexos e sistêmicos, que seriam invisíveis a métodos de análise tradicionais.

Como próximos passos, vislumbram-se as seguintes evoluções:

- **Interface Gráfica Interativa:** Desenvolver o front-end com bibliotecas de visualização como D3.js ou Vis.js para permitir que o usuário explore o grafo de forma interativa (zoom, pan, clique em nós para ver detalhes), melhorando a usabilidade.
- **Escalabilidade com Banco de Dados de Grafo:** Para analisar datasets na escala de bilhões de transações.
- **Análise em Tempo Real:** Integrar a ferramenta a um sistema de streaming (ex: Apache Kafka) para analisar transações em tempo real e gerar alertas de fraude com baixa latência.
- **Modelos de Machine Learning:** Incorporar algoritmos de aprendizado de máquina em grafos (Graph Neural Networks - GNNs) para classificar nós e arestas como fraudulentos ou não, aprendendo os padrões a partir de dados históricos.

References

Lopez-Rojas, Edgar A. (2018). Synthetic Financial Datasets For Fraud Detection. `https://www.kaggle.com/datasets/ealaxi/paysim1`. Acessado em: 08-Jun-2025.

Young Lambert Officia (2024). fraud detection system for transactions using ML. `https://www.kaggle.com/code/younglambertofficia/fraud-detection-system-for-transactions-using-ml`. Acessado em: 08-Jun-2025.