UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Utilização de Regras de Associação para apoio ao entendimento de problemas de aplicações usando metadados de consultas em SGBDs relacionais

Luiz Augusto Biazotto Filho

Monografia - MBA em Inteligência Artificial e Big Data

ICMC USP SÃO CARLOS

Luiz Augusto Biazotto Filho

Utilização de Regras de Associação para apoio ao entendimento de problemas de aplicações usando metadados de consultas em SGBDs relacionais

Monografia apresentada ao Departamento de Ciências de Computação do Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo - ICMC/USP, como parte dos requisitos para obtenção do título de Especialista em Inteligência Artificial e Big Data.

Área de concentração: Inteligência Artificial

Orientador: Profa. Dra. Cristina Dutra de Aguiar

Versão original

São Carlos

2023

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi, ICMC/USP, com os dados fornecidos pelo(a) autor(a)

B579u

Biazotto Filho, Luiz Augusto
Utilização de Regras de Associação para apoio ao entendimento de problemas de aplicações usando metadados de consultas em SGBDs relacionais / Luiz Augusto Biazotto Filho ; orientadora Cristina Dutra de Aguiar. – São Carlos, 2023.
92 p.

Trabalho de conclusão de curso (MBA em Inteligência Artificial e Big Data) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2023.

1. Inteligência Artificial. 2. Aprendizado Não-Supervisionado. 3. Regras de Associação. 4. APRIORI. 5. Descoberta de conhecimento 6. Metadados de SGBDs. I. Dutra de Aguiar, Cristina, orient. II. Título.

Luiz Augusto Biazotto Filho

Association Rules to support a better troubleshooting of application problems using relational DBMS metadata

Monograph presented to the Departamento de Ciências de Computação do Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo - ICMC/USP, as part of the requirements for obtaining the title of Specialist in Artificial Intelligence and Big Data.

Concentration area: Artificial Intelligence

Advisor: Profa. Dra. Cristina Dutra de Aguiar

Original version

São Carlos

2023

Este trabalho é dedicado aos alunos da USP, como uma contribuição das Bibliotecas do Campus USP de São Carlos para o desenvolvimento e disseminação da pesquisa científica da Universidade.

AGRADECIMENTOS

A Deus e a toda minha família por compreenderem que os vários finais de semana de estudo “longe” deles valeram a pena!

À minha orientadora, Profa. Dra. Cristina Dutra de Aguiar, por toda orientação durante o trabalho.

A todos os professores do MBA em IA e Big Data, em especial à Profa. Dra. Solange Oliveira Rezende e ao Prof. Dr. Ricardo Marcondes Marcacini, que me "socorreram"realmente em todas as situações onde precisei de apoio técnico relacionado ao Aprendizado Não-Supervisionado.

A todos os tutores deste MBA que, de alguma forma, também me apoiaram sempre quando possível, em especial ao tutores Leonardo Mauro P. Moraes e Marcos Paulo Silva Gôlo.

A todos meus amigos que nunca me deixaram desistir do problema, por mais difícil que ele parecia ser, em especial à Katia Abes, à Luísa Anastácio e também à minha professora de Italiano, Fabiana Sciamanna.

Ao Pietro Tercitano e à Jucelle Biagioli, que me apoiaram a buscar autorização para utilização dos dados do cliente. Apesar de não termos conseguido, foram eles que me deram apoio total nesse pedido.

E, por fim, ao tutor e membro da banca na defesa deste trabalho, Nícolas Roque dos Santos, que apontou sugestões, permitindo que este trabalho tivesse maior qualidade em sua versão final.

"No meio da dificuldade encontra-se a oportunidade."
Albert Einstein

RESUMO

Biazotto Filho, L. A. **Utilização de Regras de Associação para apoio ao entendimento de problemas de aplicações usando metadados de consultas em SGBDs relacionais**. 2023. 92p. Trabalho de conclusão de curso (MBA em Inteligência Artificial e Big Data) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2023.

Aplicações comerciais, independentemente do tipo ou domínio, tendem a gerar cada vez mais dados e armazená-los, seja para fins da própria aplicação ou exploração futura dos mesmos. Este crescimento de dados em sistemas gerenciadores de banco de dados (SGBDs) relacionais pode degradar o desempenho de tais aplicações, principalmente se não forem realizadas análises e intervenções constantes nas consultas realizadas. Metadados são dados sobre os dados. Eles incluem dados muito interessantes sobre cada consulta, como o tempo de execução e a quantidade de acessos à memória primária ou secundária, separados em intervalos pré-estabelecidos. Por meio de uma análise dos metadados, é possível auxiliar o entendimento do que estava acontecendo com o SGBD no momento no qual uma determinada aplicação apresentou instabilidades. Este trabalho de conclusão de curso se propõe a utilizar os metadados gerados automaticamente pelo SGBD para apoiar a investigação de causa de problemas em ambientes produtivos. A investigação é realizada por meio do uso do algoritmo de aprendizado de máquina não supervisionado APRIORI. A ideia consiste em utilizar regras de associação para encontrar os atributos dos metadados que mais ocorreram no momento em que a aplicação passava por algum problema, e posteriormente encontrar possíveis relações entre estes atributos. Para tanto, são gerados quatro conjuntos de dados diferentes contendo metadados oriundos do SGBD Oracle, é aplicada a estratégia de discretização aos dados desses conjuntos de dados e são realizados cinco diferentes experimentos. Esses experimentos são executados sobre os diferentes conjuntos de dados e são configurados em termos de remoção de atributos, valores de suporte e confiança. Adicionalmente, no trabalho também são realizadas a análise e discussão da configuração dos experimentos e dos resultados obtidos. Por meio dos resultados obtidos, é possível concluir que, quando as regras de associação são utilizadas em conjunto com outras explorações nos dados e quando se tem o conhecimento do domínio da aplicação, é possível estabelecer correlações entre as regras geradas e as consultas que tiveram mais utilização de um recurso em um intervalo de tempo.

Palavras-chave: Regras de Associação. Aprendizado Não-Supervisionado. SGBDs Relacionais. Metadados.

ABSTRACT

Biazotto Filho, L. A. **Association Rules to support a better troubleshooting of application problems using relational DBMS metadata**. 2023. 92p. Monograph (MBA in Artificial Intelligence and Big Data) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2023.

Any comercial application, no matter their type or domain, tend to generate more and more data and store them, whether for the purposes of the application itself or future knowledge exploitation. This data growth over database management systems (DBMS) may degrade the performance of such applications, especially if not frequently having their query performance evaluated. Metadata means having data about the data. They may include interesting information about each query executed, such as its execution time, amount of primary or secondary memory usage, all of this split into pre-established snapshots. When analyzing metadata, it is possible to support understanding what was going on into the DBMS at the moment the application was performing bad or not working prorperly. This study proposes to use this metadata generated automatically by the DBMS to support the root cause investigation of problems on production environments. That investigation is possible to be done using the APRIORI machine learning unsupervised algorithm. The idea is to use association rules to find the most frequent attributes in the metadata by the moment there was an application problem, and then to find possible relationship between these attributes. To make this possible, four datasets are generated containing that metadata from the Oracle DBMS, then discretization process is applied on all of them and five experiments are performed. These experiments are executed under the different datasets and are setup by using attribute reduction strategy and different parameters for support and confidence. Additionally, on this study, analysis and discussion about those parameters setup and their results are found. Analyzing the experiments results, it is possible to conclude that when the association rules are used together with another data exploitation and when the application domain of the data present on the datasets is well known, it is possible to establish correlation between the association rules generated and the queries which utilized most of the resources in a certain period of time.

Keywords: Association Rules. Unsupervised Learning. Relational Databases. Oracle query metadata history. Metadatata. SQL_ID. Execution time. DBA_HIST_SQLSTAT. Snapshots. Problem resolution. Productive application problems. Maintenance. Performance. Data growth. Graphs to support Association Rules visualization.

LISTA DE FIGURAS

Figura 1 – Exemplo retirado dos autores Kotu and Deshpande (2019) para demonstrar o princípio de *itemsets* frequentes no APRIORI 36
Figura 2 – Exemplo retirado dos autores Kotu and Deshpande (2019) para demonstrar o princípio de *itemsets* infrequentes no APRIORI 37
Figura 3 – Exemplo de funcionamento do APRIORI com filtragem dos *itemsets* de acordo com suporte mínimo escolhido 39
Figura 4 – Matriz de correlação entre atributos no *dataset* 1 (à esquerda) e *dataset* 2 (à direita) 47
Figura 5 – Matriz de correlação entre atributos no *dataset* 3 (à esquerda) e *dataset* 4 (à direita) 48
Figura 6 – Metodologia apresentando as principais fases do ciclo deste trabalho . . 49
Figura 7 – *Dataset* antes de passar por processo de discretização 50
Figura 8 – *Dataset* após passar por processo de discretização 52
Figura 9 – Algumas colunas do *dataset* após passar por processo de redução de atributos representando faixas pequenas e médias e remoção de linhas com transações zeradas. Na figura não são exibidas todas as colunas. Porém, não existem linhas sem que ao menos uma coluna esteja preenchida com o valor 1. 56
Figura 10 – Mapa de calor das regras com melhores valores da medida *lift* 60
Figura 11 – Mapa de calor das melhores regras com melhores valores da medida *lift* 60
Figura 12 – Algumas estatísticas coletadas no dia 26 de maio durante o período da tarde, as quais mostram mudança em planos de execução de consultas e seu aumento de consulta em memória quando comparada com o plano anterior 61
Figura 13 – *Dataset* do dia 26 de maio, reduzido e discretizado 62
Figura 14 – Mapa de calor das regras com melhores valores da medida *lift* para o experimento 3 65
Figura 15 – Mapa de calor das melhores regras com melhores valores da medida *lift* para o experimento 4 69
Figura 16 – Mapa de calor das melhores regras com melhores valores da medida *lift* para o experimento 5 73
Figura 17 – Grafo utilizando a espessura das ligações para representar a medida *lift* e tamanho do nó representando quantidade de regras para o experimento 2 76
Figura 18 – Mapa de calor com suporte mínimo 0.001 e confiança 0.6 gerou novas regras fortes para o experimento 2 77

Figura 19 – Grafo utilizando a espessura das ligações para representar a medida *lift* e tamanho do nó representando quantidade de regras para o experimento 3, com suporte 0.003 e confiança 0.6 78
Figura 20 – Mapa de calor do experimento 3 com suporte mínimo 0.003 e confiança 0.6 79
Figura 21 – Grafo utilizando a espessura das ligações para representar a medida *lift* e tamanho do nó representando quantidade de regras para o experimento 4, com suporte 0.01 e confiança 0.7 80
Figura 22 – Mapa de calor com suporte mínimo 0.01 e confiança 0.7 gerou novas regras fortes para o experimento 4 81
Figura 23 – Grafo utilizando a espessura das ligações para representar a medida *lift* e tamanho do nó representando quantidade de regras para o experimento 5, com suporte 0.001 e confiança 0.7 82
Figura 24 – Mapa de calor com suporte mínimo 0.01 e confiança 0.7 para o experimento 5 83

LISTA DE TABELAS

Tabela 1 – Exemplo de transações de um fluxo de cliques em uma página Web . . 31
Tabela 2 – *Dataset* das transações mostrada na Tabela 1 31
Tabela 3 – *Dataset* condensado após remoção do item Arts 38
Tabela 4 – Cálculo do Suporte para os itens frequentes no exemplo 39
Tabela 5 – Dataset 1: Porcentagem dos atributos discretizados nos bins baixo, médio e alto utilizando as estratégias uniform e kmeans no KBinsDiscretizer 53
Tabela 6 – Dataset 2: Porcentagem dos atributos discretizados nos bins baixo, médio e alto utilizando as estratégias uniform e kmeans no KBinsDiscretizer 53
Tabela 7 – Dataset 3: Porcentagem dos atributos discretizados nos bins baixo, médio e alto utilizando as estratégias uniform e kmeans no KBinsDiscretizer 54
Tabela 8 – Dataset 4: Porcentagem dos atributos discretizados nos bins baixo, médio e alto utilizando as estratégias uniform e kmeans no KBinsDiscretizer 54
Tabela 9 – Valor máximo do atributo no intervalo/consulta 57
Tabela 10 – Alguns dos Itemsets mais frequentes gerados pelo experimento 2 58
Tabela 11 – Algumas regras geradas pelo experimento 2, ordenadas pela medida *lift* em ordem decrescente, sendo que ant support = suporte do antecedente; cons support = suporte do consequente; supp = suporte; conf = confiança; lev = levarage; conv = convicção 59
Tabela 12 – Valor máximo do atributo no intervalo/consulta do experimento 3 . . . 63
Tabela 13 – Itemsets mais frequentes gerados no experimento 3 64
Tabela 14 – Regras geradas pelo experimento 3, ordenadas pela medida *lift* em parâmetros decrescente 65
Tabela 15 – Valor máximo do atributo no intervalo/consulta para o experimento 4 . 66
Tabela 16 – Vinte Itemsets mais frequentes gerados para o experimento 4 67
Tabela 17 – Vinte melhores regras geradas no experimento 4, ordenadas pela medida *lift* em ordem decrescente, sendo que ant support = suporte do antecedente; cons support = suporte do consequente; supp = suporte; conf = confiança; lev = levarage; conv = convicção 68
Tabela 18 – Valor máximo do atributo no intervalo/consulta para o experimento 5 . 70
Tabela 19 – Itemsets mais frequentes gerados para o experimento 5, com suporte igual ou superior a 0.03 71
Tabela 20 – Regras geradas para o experimento 5, ordenadas pela medida *lift* em ordem decrescente (embora os valores das medidas estudadas sejam idênticos) 72

LISTA DE ABREVIATURAS E SIGLAS

SGBD(s)	Sistema(s) Gerenciador(es) de Banco de Dados
DBMS(s)	Database Management System(s)
Dataset	Conjunto de dados estudado
Snapshot	Janela ou intervalo de tempo
View	Visualização Oracle com metadados
SQL_ID	Identificação única do Oracle para uma consulta (baseado em MD5)
Churn	Medida do número de indivíduos ou items movendo de uma comunidade em um período de tempo
Metadados	Dados sobre os dados
Itemset(s)	Item ou Conjunto de itens
SGA	System Global Area (SGBD Oracle)

SUMÁRIO

1 INTRODUÇÃO 25
1.1 Motivação 25
1.2 Objetivo e Contribuição 27
1.3 Estruturação da monografia 27

2 FUNDAMENTAÇÃO TEÓRICA 29
2.1 Mineração de Dados 29
2.2 Aprendizado de Máquina 30
2.2.1 Regras de Associação 30
2.2.2 Medidas de frequência 33
2.2.3 Algoritmo APRIORI 36
2.2.3.1 Funcionamento do APRIORI através de exemplo 38
2.3 Banco de Dados 40
2.4 Trabalhos Relacionados 41

3 COLETA E TRATAMENTO DOS DADOS 43
3.1 Extração dos dados das bases relacionais 43
3.1.1 Consulta 43
3.1.2 Atributos extraídos e seus significados 45
3.1.3 Análise Inicial dos Atributos 46
3.2 Metodologia 48

4 PROPOSTAS DE EXECUÇÃO 51
4.1 Discretização dos Dados 51
4.2 Aplicação do Algoritmo APRIORI 55
4.2.1 Experimento 1 55
4.2.2 Experimento 2 55
4.2.3 Experimento 3 61
4.2.4 Experimento 4 66
4.2.5 Experimento 5 69

5 AVALIAÇÃO EXPERIMENTAL 75
5.1 Configuração dos Experimentos 75
5.1.1 Experimento 1 75
5.1.2 Experimento 2 75
5.1.3 Experimento 3 78
5.1.4 Experimento 4 80

5.1.5 Experimento 5 82
5.2 Resultados e Discussões 84
5.2.1 Experimentos 2 e 3 84
5.2.2 Experimentos 4 e 5 84

6 CONCLUSÕES 87
6.1 Revisão dos Resultados 87
6.2 Dificuldades 89
6.3 Trabalhos Futuros 89

Referências 91

1 INTRODUÇÃO

A aplicação da Tecnologia da Informação está presente em todas as áreas de conhecimento. Muitas tarefas que antes eram feitas de forma manual hoje podem ser realizadas online, via internet. Os sistemas são projetados, implementados e devem ser mantidos de maneira que apresentem bom desempenho durante todo seu ciclo de vida. Fazer a manutenção destes sistemas, de maneira que este ofereça suporte para toda a demanda de novos clientes realizando transações, pode ser um desafio.

Dados destes sistemas geralmente são armazenados em sistemas gerenciadores de banco de dados (SGBD) relacionais, para diversas aplicações diferentes, como cadastro de clientes, geração de ordens de compra ou contabilização de eventos para futura cobrança de fatura. Neste sentido, o crescimento cada vez maior do volume de dados armazenados nestes sistemas em produção pode exigir manutenção constante para evitar problemas de desempenho em aplicações.

SGBDs relacionais geralmente registram em seus metadados dados muito interessantes sobre vários aspectos, como tempo de execução das consultas e número de acessos à memória memória primária e memória secundária (disco). Esses dados são coletados e armazenados em intervalos pré-estabelecidos pelo sistema, chamados geralmente de *snapshots*. Por meio de uma análise dos metadados armazenados, é possível auxiliar o entendimento dos problemas sendo enfrentados pelas aplicações no momento que estas apresentam alguma instabilidade.

1.1 Motivação

Baseado em problemas que são enfrentados no dia-a-dia do administrador de bancos de dados, como problemas de performance em consultas, verificação de sessões que a aplicação possa ter esquecido de fazer o *commit*, o administrador de bancos de dados também precisa investigar problemas em sistema produtivos. Neste trabalho propõe-se explorar os metadados de SGBDs relacionais. Os metadados são coletados automaticamente pelos SGBDs. Neste trabalho, utiliza-se a coleta realizada pelo SGBD Oracle, que disponibiliza um histórico de tais estatísticas de consultas na visualização interna "DBA_HIST_SQLSTAT". Este volume histórico de estatísticas é humanamente difícil de ser analisado, devido à sua complexidade e tamanho. O tamanho é dependente da atividade no SGBD e, neste trabalho, foi identificada uma média de aproximadamente 12 mil linhas geradas a cada 24 horas. Cada linha do *Dataset* representa a quantidade de cada recurso consumido do SGBD num intervalo de tempo, por uma determinada consulta. Portanto, a aplicação de algoritmos de aprendizado de máquina pode facilitar a extração de conhecimento destes *Datasets*.

Existem outras ferramentas que possibilitam analisar estado do SGBD em um determinado intervalo de tempo, muitas vezes disponibilizadas pela própria fabricante do mesmo. Uma destas é o *AWR Report*, um tipo de relatório com muita informação sobre estes metadados. Por conta de ser bem detalhado, pode ser um pouco mais complicado encontrar o conhecimento que se deseja, e também ele requer privilégios específicos de visualização que nem sempre estão disponíveis. Durante o dia a dia de trabalho do administrador de banco de dados, algumas vezes foram solicitadas a extração destas estatísticas ou metadados, e sua posterior análise em ferramentas do tipo Excel, com uso de tabelas *pivot* para tentar identificar tendências e consultas que podem ter causado esgotamento de recursos no SGBD. Foi, então, pensado em como o Aprendizado de Máquina poderia auxiliar nesta extração de conhecimento de maneira mais objetiva e automática.

A linha de pesquisa do aprendizado de máquina se preocupa com questões sobre como construir programas que automaticamente melhorem com a experiência, desde programas que fazem mineração de dados e aprendem a detectar transações fraudulentas até sistemas de recomendação que aprendem de acordo com a experiência do usuário, ou até mesmo a veículos autônomos que aprendem como dirigir em rodovias públicas (MITCHELL, 1997). Por meio de seu uso, é possível explorar algoritmos que possam encontrar anomalias ou mesmo encontrar associações entre as estatísticas, de forma a tentar entender se algum tipo de evento aconteceu junto com outro e apoiar na investigação de causa de problemas em ambientes produtivos.

Existem dois paradigmas fundamentais dentro do aprendizado de máquina: supervisionado e não supervisionado (MITCHELL, 1997). No aprendizado supervisionado, os dados são rotulados, de forma que a resposta da classificação (ou seja, o rótulo) é usado no treinamento do algoritmo. Por outro lado, no aprendizado não-supervisionado, não se tem exemplos de resposta de classificação, ou seja, os dados não são rotulados. Como resultado, é necessário analisar os exemplos de treinamento e tentar determinar se alguns dados podem ser agrupados ou *clusterizados*. Segundo Bramer (2013), problemas de classificação são os problemas mais frequentes em mineração de dados.

Em conjuntos de dados existem exemplos (chamados também de instâncias), onde cada exemplo contem valores de um número de variáveis, chamados de atributos em mineração de dados (BRAMER, 2013). Quando o objetivo é utilizar os dados para predizer o valor de atributos que ainda não se conhece, tem-se que o conjunto de dados é rotulado, e por isso trata-se de um aprendizado supervisionado (BRAMER, 2013). Dados não rotulados requerem um tipo de aprendizado não-supervisionado, principalmente quando deseja-se utilizar um conjunto de treinamento para encontrar relacionamentos entre os valores das variáveis, geralmente na forma de regras de associação (BRAMER, 2013).

Neste trabalho, os metadados disponibilizados pelos SGBDs relacionais não possuem informações de rótulo ou classificação. Portanto, é empregado o aprendizado não

supervisionado. Em especial é utilizada a técnica de regras de associação por meio do algoritmo APRIORI (YAMAMOTO, 2009).

1.2 Objetivo e Contribuição

Como objetivo, o trabalho pretende estudar os metadados dos SGBDs relacionais para auxiliar as empresas na resolução de problemas em sistemas produtivos por meio da aplicação de regras de associação. Ao identificar possíveis causas para esses problemas, ações podem ser tomadas para evitá-los novamente, gerando menos tempo de instabilidade ou indisponibilidade de sistemas. As regras de associação geralmente são utilizadas em *datasets* transacionais, geralmente representando itens de um carrinho de compras, para se tentar identificar padrões de comportamento entre os produtos do carrinho. Neste trabalho, foi escolhida as regras de associação de forma análoga, ao se pensar que recursos consumidos no SGBD poderiam representar os produtos de uma carrinho de compras.

Para tanto, no trabalho primeiramente é realizada a extração de metadados de uma aplicação usando o SGBD Oracle, que foi escolhido por ser o mais utilizado no ambiente de trabalho no momento desta pesquisa. Então, temos maior diponibilidade para geração dos *datasets* a cada nova situação onde houve problema na produção. São, então, gerados quatro diferentes conjuntos de dados, cada qual com um volume de dados diferente e contextualizado por um problema específico. Os dados desses conjuntos de dados são tratados em uma etapa de pré-processamento, a qual inclui a discretização desses dados. Na sequência, são desenvolvidos cinco experimentos aplicando-se o algoritmo APRIORI para extração dos atributos mais frequentes e posterior geração das regras de associação. Por fim, os resultados obtidos são analisados e investigados quanto a qual consulta, em qual momento e qual o recurso (atributo) mais utilizado.

Com o desenvolvimento do presente trabalho, espera-se que novos problemas semelhantes possam ser evitados em ambientes produtivos. Ou seja, espera-se que os resultados do trabalho possam contribuir para um baixo tempo de indisponibilidade ou instabilidade das aplicações, gerando melhor experiência ao usuário.

1.3 Estruturação da monografia

Além deste capítulo introdutório, esta monografia está estruturada em mais cinco capítulos, conforme descrito a seguir.

- Capítulo 2: Descreve a fundamentação teórica necessária para o desenvolvimento do trabalho, incluindo conceitos de mineração de dados, aspectos relacionados ao aprendizado de máquina, com destaque para as regras de associação e ao algoritmo APRIORI, e características dos metadados dos SGBDs relacionais. Também resume trabalhos relacionados.

- Capítulo 3: Descreve como os dados são obtidos e utilizados. Acrescenta, também, uma descrição para cada atributo pertencente ao conjunto de dados. Uma análise inicial dos atributos é realizada utilizando Mapas de Calor que representam a correlação entre os atributos do dataset. Há também uma introdução da metologia de todo o desenvolvimento do trabalho.

- Capítulo 4: Descreve como o processo de discretização dos dados é realizado. Compara estratégias diferentes de discretização dos dados. Também descreve todos os cinco experimentos realizados, com informação de seus datasets e parâmetros utilizados em cada experimento.

- Capítulo 5: Avalia os experimentos realizados e acrescenta figuras representando as regras de associação geradas em formato de grafos, para tentar facilitar a compreensão das mesmas.

- Capítulo 6: Faz a revisão dos resultados e conclui, comparando as regras geradas com tabelas de consultas que mais consumiram recursos. Sugere trabalhos futuros, citando também os desafios encontrados ao utilizar as regras de associação.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são descritos os conceitos necessários para o desenvolvimento do trabalho. Na seção 2.1 são descritos conceitos de mineração de dados. Na seção 2.2 são detalhados aspectos relacionados ao aprendizado de máquina, com destaque para as regras de associação. A proposta deste trabalho consiste em analisar a associação entre diversas métricas disponibilizadas nos metadados dos SGBDs relacionais. Portanto, na seção 2.3 são destacados aspectos do SGBD Oracle que são utilizados para a disponibilização das métricas analisadas. Finalmente, na seção 2.4, são resumidos trabalhos relacionados.

2.1 Mineração de Dados

Minerar dados consiste em coletar, limpar, processar, analisar e adquirir conhecimentos úteis a partir destes dados. Segundo Aggarwal (2015), o termo "mineração dos dados" vem do conceito das minas de minérios, no sentido de comparação com as várias fases que precisam ser realizadas para se chegar ao produto final.

Vários domínios de aplicação podem fazer coleta de dados automaticamente (AGGARWAL, 2015). Um destes domínios pode ser entendido como o de tecnologia da informação, tanto nos ambientes de tecnologia dentro do ambiente da empresa quanto atualmente com a movimentação dos serviços para a nuvem. Especialmente na nuvem, entender melhor sobre a carga dos sistemas é importante para tomar decisões de manutenção e expansão.

O crescimento na geração de dados é resultado dos avanços da tecnologia na vida moderna. O problema é que estes dados são coletados em diferentes locais, o que faz com que estes dados não tenham um padrão ou estrutura. É por esta razão que uma das fases mais complexas da mineração de dados é o pré-processamento dos dados. Fazer um bom pré-processamento pode ajudar a ter melhores resultados quando da aplicação de algoritmos e técnicas relacionadas.

O pré-processamento de dados, uma das fases mais importantes da mineração, começa logo após a coleta dos dados. Ele pode ser dividido em três fases, conforme descrito a seguir. Na primeira fase, de extração dos atributos, é necessário um conhecimento do domínio da aplicação. A segunda fase, de limpeza, consiste em verificar se algo precisa ser removido, em caso de dados faltantes ou incorretos. Na terceira fase, de seleção dos atributos e transformação, verifica-se a necessidade de alterações de dimensionalidade ou transformação dos tipos de dados dos atributos.

Com relação à última fase, ela é um pouco controversa de ser classificada como uma fase de pré-processamento, porque a escolha dos atributos pode depender de quais

algoritmos são utilizados no processo de classificação. Este pode ser um problema maior para o aprendizado não-supervisionado, pois não se tem os rótulos para futura avaliação do modelo e, neste caso, busca-se atributos que maximizem a tendência ao agrupamento (AGGARWAL, 2015).

2.2 Aprendizado de Máquina

Ainda não foi possível fazer os computadores aprenderem como os humanos aprendem. Entretanto, vários algoritmos e técnicas computacionais têm sido propostas com o objetivo de prover diferentes tipos de aprendizado. Neste sentido, uma definição ampla de aprendizado de máquina engloba qualquer tipo de programa que melhore o desempenho de alguma tarefa de acordo com sua experiência. Este aprendizado é realizado treinando o programa com conhecimento anterior, já conhecido (MITCHELL, 1997).

Para criar um sistema de aprendizado, é necessário primeiro ter conhecimento de qual é o tipo de aprendizado que se deseja obter, qual a função de representação do conhecimento do atributo conhecido e qual o mecanismo de aprendizado (MITCHELL, 1997).

Um exemplo amplamente utilizado em problemas de classificação consiste em analisar cestas de mercado com objetivos como facilitar a disposição dos itens dentro do mercado e melhorar estratégias de *marketing*. Neste trabalho, em um contexto um pouco diferente, pode-se imaginar que a cesta de de compras do mercado é, na verdade, recursos sendo consumidos de um SGBD. Então, a proposta desde trabalho consiste em analisar a associação entre diversas métricas disponibilizadas nos metadados dos SGBDs relacionais.

Dentre as técnicas de aprendizado de máquina não supervisionado existentes, neste trabalho empregam-se as regras de associação, sendo seus conceitos descritos na seção 2.2.1. As medidas de frequência associadas a essas regras são detalhadas na seção 2.2.2. A aplicação das regras de associação usualmente requer encontrar os conjuntos de itens mais frequentes, sendo o algoritmo APRIORI bastante utilizado para este fim. Este algoritmo é detalhado na seção 2.2.3.

2.2.1 Regras de Associação

As regras de associação pertencem a uma subdivisão do aprendizado não supervisionado capaz de descobrir diferentes padrões ocultos nos dados, na forma de regras mais fáceis de serem reconhecidas. Segundo Kotu and Deshpande (2019), analisar associação mede a força de co-ocorrências entre um item e outro, com o objetivo de identificar padrões interessantes nestas co-ocorrências. Os mesmos autores exemplificam utilizando a Tabela 1 com transações representando cliques de usuários em uma página da Internet, separados por sessões.

Identificação Sessão	Lista da categoria de mídias acessadas
1	{News, Finance}
2	{News, Finance}
3	{Sports, Finance, News}
4	{Arts}
5	{Sports, News, Finance}
6	{News, Arts, Entertainment}

Tabela 1 – Exemplo de transações de um fluxo de cliques em uma página Web

Então, o respectivo *dataset* para este exemplo é mostrado na Tabela 2.

Identificador da Sessão	News	Finance	Entertainment	Sports	Arts
1	1	1	0	0	0
2	1	1	0	0	0
3	1	1	0	1	0
4	0	0	0	0	1
5	1	1	0	1	0
6	1	0	1	0	1

Tabela 2 – *Dataset* das transações mostrada na Tabela 1

Os autores Kotu and Deshpande (2019) também lembram que o formato binário mostrado na Tabela 2 ignora qualidades como tempo gasto na categoria ou a sequência de acesso, que poderiam ser importantes em alguns casos de análise de sequência.

As regras são dispostas na forma de um item implicar em outro item. Por exemplo, pode-se citar a seguinte regra de associação: “clientes que compraram este item, possivelmente também compraram aquele outro”. No modelo A implica em B, tem-se A como antecedente e B como consequente ou conclusão da regra. Os antecedentes e consequentes podem conter mais de um item. Então, como exemplo de regra:

$$\{News, Finance\} \implies \{Sports\} \tag{2.1}$$

Esta regra implica que, se usuários acessaram *News* e *Finance* na mesma sessão, exitem altas chances deles também acessarem *Sports*, baseado no histórico de transações. A combinação dos itens *News* e *Finance* é denominada *itemset*, ou conjunto de itens. Os *itemsets* podem ocorrer tanto na parte antecedente quanto consequente da regra, mas não devem existir itens comuns aos dois lados da regra.

Para extrair as regras de associação, os três passos descritos a seguir devem ser realizados:

- Preparar o conjunto de dados para um formato transacional, no qual cada linha representa uma transação contendo vários itens;
- Encontrar os itens ou conjunto de itens mais frequentes. Os algoritmos podem limitar a análise aos itens mais frequentes apenas, de maneira a encontrar regras melhores. Segundo Kotu and Deshpande (2019), para uma análise de associação de n itens é possível encontrar 2^n -1 conjuntos de itens excluindo o nulo. Então, conforme o número de itens aumenta, o número de conjunto de itens aumenta exponencialmente. Por esta razão é difícil encontrar um valor de suporte mínimo que descarte os itens menos frequentes. É comum a exclusão de itens de maneira a investigar subconjuntos usando apenas atributos relevantes. Isso pode ser observado nas Figuras 1 e 2, onde o item *Arts* foi eliminado por falta de interesse em analisar regras que o continham.
- Gerar as regras de associação relevantes a partir do conjunto de dados e depois filtrar as regras de acordo com a medida de interesse escolhida.

Segundo Kotu and Deshpande (2019), mesmo para conjuntos de dados pequenos com dezenas de itens, podem ser geradas muitas regras de associação. Essa característica é corroborada por YAMAMOTO (2009), que destaca que existem vários problemas citados na literatura relacionados à extração regras de associação, conforme destacado a seguir:

- "Analistas não sabem como deve ser o ajuste ideal de limiares mínimos" (IVKOVIC; YEARWOOD; STRANIERI, 2003)
- "(...) obtém-se bem mais regras do que [a quantidade] manipulável confortavelmente pelos humanos" (HOFMANN; SIEBES; WILHELM, 2000)
- "(...) é bastante entediante para o usuário (...) achar conhecimento interessante (...) em conjuntos que podem conter centenas de milhares de regras (...)" (BLANCHARD *et al.*, 2003)
- "Regras são difíceis de entender" (HOFMANN; SIEBES; WILHELM, 2000)
- "(...) mesmo correlações fortes entre atributos das regras obtidas não são sempre óbvias" (HOFMANN; SIEBES; WILHELM, 2000)

Portanto, parametrizar corretamente o suporte mínimo que descarte de forma apropriada os itens menos frequentes é fundamental. Segundo Agrawal and Srikant (1994),

todos os algoritmos para extração de regras de associação devem encontrar conjuntos de itens frequentes em transações de maneira eficiente.

2.2.2 Medidas de frequência

Segundo Kotu and Deshpande (2019), a força de uma regra de associação é geralmente quantificada pelos valores de suporte e confiança de cada regra. Em alguns casos especiais, também podem ser utilizadas outras medidas como *lift* e convicção. Todas as medidas são baseadas na frequência relativa de ocorrências em um conjunto de itens utilizados para treinamento. Portanto, é importante que o conjunto de dados não esteja enviesado e represente realmente o universo de transações.

A medida suporte pode ser descrita de diferentes formas, a saber:

- Suporte do item: A medida de suporte de um item é a frequência que aquele item ocorre nas transações do conjunto de dados, pelo menos uma vez. No *dataset* exibido na Tabela 2, o suporte do item *News* é cinco, das seis transações, ou 5/6, ou 0.83. Então, tem-se:

```
Support({News}) = 5/6 = 0.83
Support({Sports}) = 2/6 = 0.33
```

- Suporte do conjunto de itens: A frequência de um conjunto de itens é a co-ocorrência de ambos em uma transação com relação ao total de transações do conjunto de dados. Ainda tendo o *dataset* exibido na Tabela 2 como referência, o suporte do *itemset {News, Finance}* ocorre quando há co-ocorrência de ambos em uma transação, em relação a todas as transações:

```
Support({News, Finance}) = 4/6 = 0.67
```

- Suporte da regra: Medida de quantas vezes todos os itens da regra aparecem em relação a todas as transações. Como exemplo referenciado no *dataset* exibido na Tabela 2, na regra:

```
{News} → {Sports}
```

News e *Sports* ocorrem em duas das seis transações e, assim, o suporte para esta regra é 0.33. A medida de suporte da regra indica o quanto é viável considerá-la.

A medida de suporte favorece os itens mais frequentes. No exemplo de cesta de mercado, pode ser bem interessante estudá-los e melhorar as vendas. Já as regras com baixo suporte são itens que não ocorrem com frequência e podem levar a regras que não fazem sentido, no caso deste mesmo exemplo. O valor da medida de suporte escolhido para o algoritmo filtra os itens não frequentes.

Ainda segundo Kotu and Deshpande (2019), com relação às demais medidas, elas são descritas conforme segue:

- Confiança: A medida de confiança mede a probabilidade da ocorrência do consequente da regra baseado no total de transações que contém o antecedente da regra. Como exemplo, em uma regra que contém os itens A e B implicando em um item C, ela representa a probabilidade de uma transação que contém A e B levar ao item C. Apesar da medida de confiança ser muito utilizada, ela não considera a frequência de ocorrência do consequente, e no caso de consequentes não frequentes, pode gerar regras que talvez não fossem interessantes. Isso significa que ela não mede a dependência entre os itens. Pode ser calculada com a seguinte fórmula:

$$Confidence(X \longrightarrow Y) = \frac{Support(X \cup Y)}{Support(X)} \tag{2.2}$$

Então, no caso da regra: {News, Finance} → {Sports} Se um transação tem ambos *News* e *Finance*, qual a probabilidade de ver *Sports* na mesma transação? A medida de confiança consegue responder esta pergunta:

```
Confidence({News, Finance} -> {Sports})
= Support({News, Finance, Sports}) / Support({News, Finance)}
= (2/6) / (4/6)
= 0.5
```

Metade das transações que contém *News* e *Finance* também contém *Sports*. Isso significa que, neste exemplo, 50% dos usuários que visitaram as páginas de *News* e *Finance* também visitaram a página de *Sports*.

- Lift: Para resolver o problema introduzido pela confiança, a medida *lift* adiciona o suporte do consequente como denominador na fórmula de cálculo da confiança. Valores de *lift* próximos a 1 significam que o antecedente e o consequente são independentes e, portanto, a regra talvez não seja interessante. Como é uma medida simétrica, valores maiores que 1 devem ser interpretados como antecedentes e consequentes positivamente dependentes e valores menores que 1 devem ser interpretados como antecedentes e consequentes negativamente dependentes. Em

geral, quanto maior o valor do *lift*, mais interessante a regra é. A medida pode ser calculada da seguinte forma:

$$Lift(X \longrightarrow Y) = \frac{Support(X \cup Y)}{Support(X) \times Support(Y)} \tag{2.3}$$

Então, ainda usando os dados de exemplo da Tabela 2:

```
Lift({News, Finance} -> {Sports})
= Support(X union Y) / Support(X) x Support(Y)
= 0.333 / ( 0.667 x 0.33 )
= 1.5
```

- Convicção: Segundo Brin *et al.* (1997), diferente da confiança, a convicção é normalizada com base em ambos antecedentes e consequentes da regra, como a noção estatística da correlação. É uma medida direcional ou assimétrica e mede a força da dependência entre o conjunto de itens da regra. Valores próximos de 1 identificam independência entre antecedente e consequente, e regras com alta confiança (próximos de 1) possuem valores de convicção tendendo ao infinito.

$$Conviction(X \longrightarrow Y) = \frac{1 - Support(Y)}{1 - Confidence(X \longrightarrow Y)} \tag{2.4}$$

Pelo exemplo da Tabela 2, têm-se o cálculo da convicção:

```
Conviction({News, Finance} -> {Sports})
= (1 - 0.33 ) / (1 - 0.5)
= 1.32
```

Uma convicção de valor 1.32 significa que a regra {News, Finance} → {Sports} seria incorreta 32% mais frequentemente se o relacionamento entre {News, Finance} e {Sports} é puramente aleatório.

Kotu and Deshpande (2019) também destacam que as regras de associação podem produzir regras que não fazem muito sentido, como combinações estranhas de cigarro com adesivos de nicotina. Isto está relacionado ao fato de que a análise é usualmente realizada sobre bilhões de transações com qualquer tipo de possibilidade de relacionamento entre os itens. Logo, é necessário possuir algum conhecimento analítico ou de negócios para aplicar o resultado obtido pelas regras geradas.

2.2.3 Algoritmo APRIORI

O algoritmo APRIORI apresenta uma estrutura simples baseada em entrelaçamento dos conjuntos de itens de maneira a reduzir o número de conjuntos de itens a serem testados para a medida de suporte escolhida. Na matemática discreta ou álgebra linear, estas estruturas são chamadas de reticulados. Isso acontece porque, segundo Tan, Steinbach and Kumar (2005), a base do algoritmo é que "se um conjunto de itens é frequente, então todos os seus subconjuntos também serão". Um conjunto de itens é considerado frequente se sua medida de suporte é maior ou igual à medida de suporte escolhida para execução do algoritmo.

Ainda usando o *dataset* da Tabela 2, se o conjunto de itens ou *itemset* {News, Finance, Sports} é frequente, ou seja, a sua medida de suporte (0.33) é maior que o suporte definido para execução do algoritmo (por exemplo, 0.25), então todos os seus subconjuntos também são considerados frequentes. Isso pode ser visualizado na Figura 1.

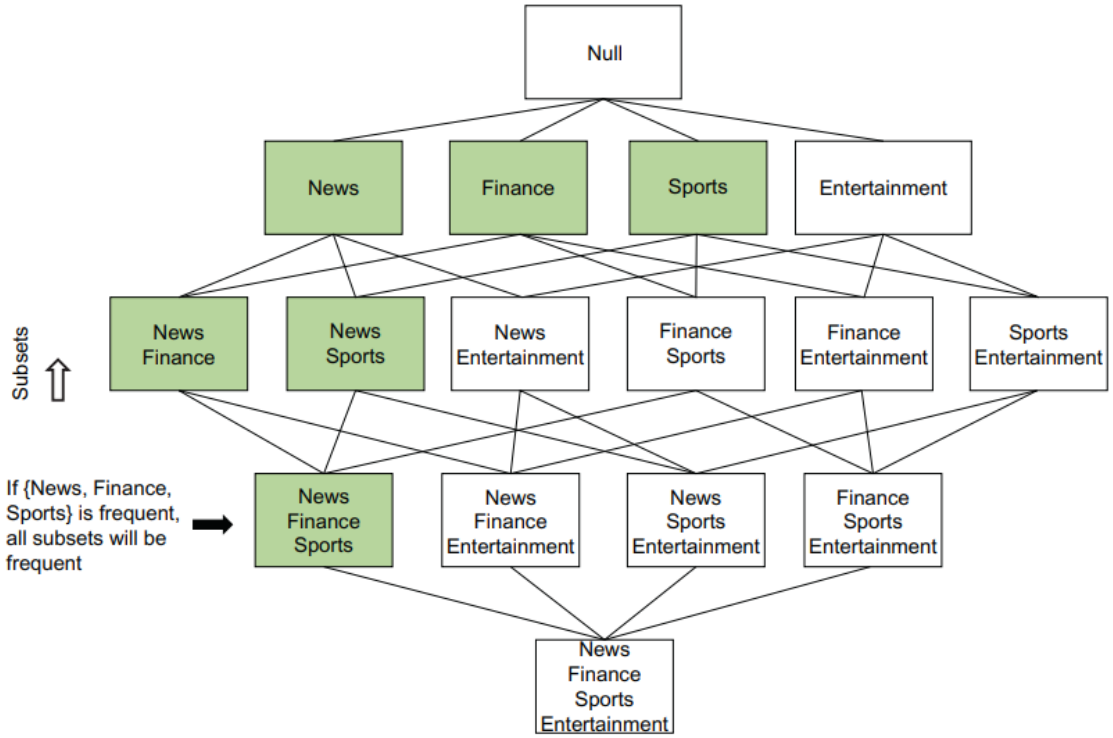

Figura 1 – Exemplo retirado dos autores Kotu and Deshpande (2019) para demonstrar o princípio de *itemsets* frequentes no APRIORI

As medidas de suporte dos subconjuntos de *itemsets*, neste exemplo, são:

```
Support {News, Finance, Sports} = 0.33 (acima do suporte escolhido)
Support {News, Finance} = 0.66
Support {News, Sports} = 0.33
```

```
Support {News} = 0.83
Support {Sports} = 0.33
Support {Finance} = 0.66
```

De maneira inversa, Kotu and Deshpande (2019) mencionam que se o *itemset* é infrequente, então todos os seus superconjuntos serão infrequentes também. No exemplo ainda utilizando o *dataset* da Tabela 2, o suporte de *Entertainment* é 0.16, e o suporte de todos os superconjuntos que contém *Entertainment* como um de seus itens serão iguais ou menor que 0.16. Eles são considerados infrequentes quando se escolhe 0.25 para execução do algoritmo. A exclusão dos superconjuntos pode ser vista conforme a Figura 2. O princípio de funcionamento do APRIORI torna-se interessante porque nem todos os *itemsets* precisarão ser considerados para teste e cálculo do suporte, de maneira que a geração dos *itemsets* frequentes pode ser mais eficiente pela eliminação destes *itemsets* que tem um item ou *itemset* infrequentes (BODON, 2005).

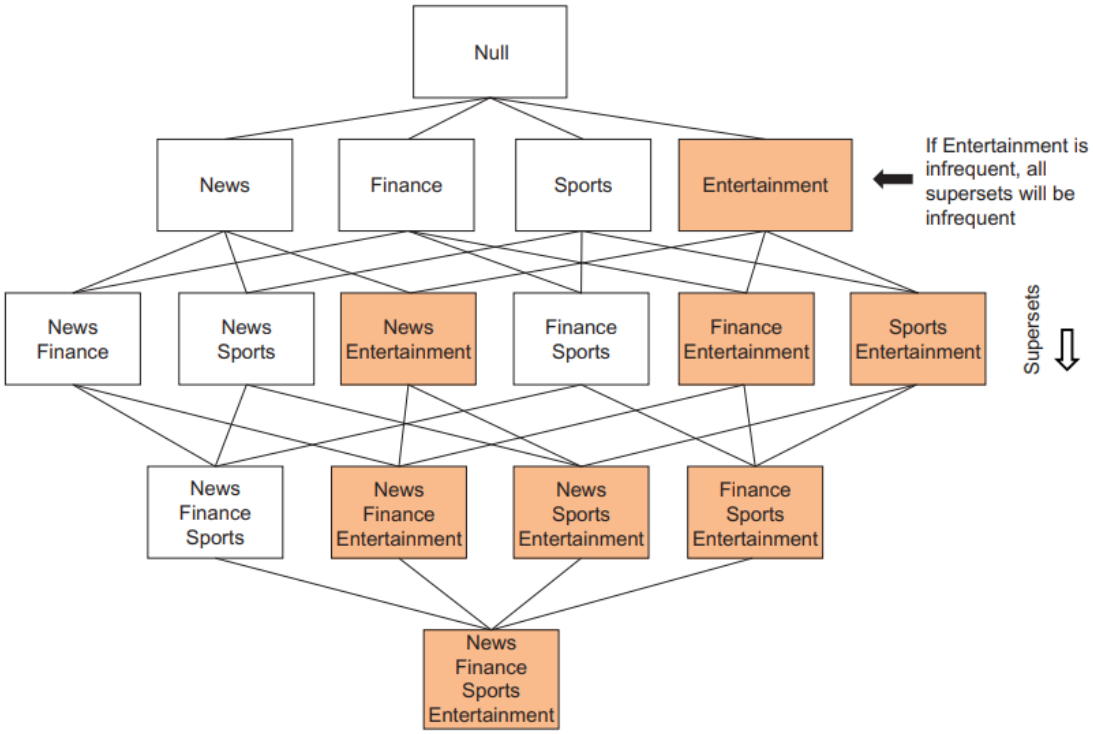

Figura 2 – Exemplo retirado dos autores Kotu and Deshpande (2019) para demonstrar o princípio de *itemsets* infrequentes no APRIORI

Na literatura, existem outros algoritmos que poderiam ser utilizados para o desenvolvimento deste trabalho, como o algoritmo FP-Growth, descrito por Han, Pei and Yin (2000), que também realiza a extração de regras de associação. Entretanto, como algoritmos de extração de regras de associação são determinísticos, não foi considerado pois produziria o mesmo resultado do APRIORI.

2.2.3.1 Funcionamento do APRIORI através de exemplo

Considere o exemplo da Tabela 3, que é uma versão condensada do primeiro mostrado no *dataset* da Tabela 2. Continuam as mesmas seis transações. Se o suporte escolhido para execução do algoritmo for de 0.25, então espera-se que todos os itens apareçam em pelo menos duas, das seis transações (KOTU; DESHPANDE, 2019).

Identificador da Sessão	News	Finance	Entertainment	Sports
1	1	1	0	0
2	1	1	0	0
3	1	1	0	1
4	0	0	0	0
5	1	1	0	1
6	1	0	1	0

Tabela 3 – *Dataset* condensado após remoção do item Arts

No exemplo, o *Support count*, ou quantidade do suporte é a contagem absoluta das transações que contém aquele *itemset*, e *Support* é taxa da quantidade do suporte em relação a quantidade total de transações no *dataset*. Todos os *itemsets* abaixo do suporte mínimo definido, que no caso deste exemplo é 2, podem ser eliminados dos processamentos futuros. A Tabela 4 mostra a quantidade do suporte (absoluto) e o suporte relativo ao *dataset*. Uma vez que o item *Entertainment* tem quantidade do suporte inferior ao limite escolhido no exemplo, ele pode também ser removido da próxima iteração de geração de *itemsets*. O próximo passo é então gerar os possíveis três *itemsets* com dois itens, para os itens {News}, {Finance} e {Sports}. Se o item *Entertainment* não tivesse sido removido, seriam seis *itemsets* com dois itens cada. A figura 3 mostra de maneira visual a representação dos *itemsets* após a eliminação do item *Entertainment*, por não atingir o suporte mínimo escolhido (KOTU; DESHPANDE, 2019).

Este processo continua até que todos os *n-itemsets* sejam considerados de conjuntos anteriores. No final, existem sete *itemsets* frequentes que estão conformes com o suporte mínimo escolhido, como também pode ser visto na Figura 3. O número total de *itemsets* possíveis é 15 (2^4 -1). Após eliminar *Entertainment* no primeiro passo, sete *itemsets* adicionais não precisam ser gerados por não estarem conformes com o suporte mínimo definido (WITTEN; FRANK, 2005).

Uma vez que *itemsets* frequentes são gerados, o próximo passo é gerar as regras de associação, no formato antecedente (premissa) -> consequente (conclusão). A importância de uma regra pode ser medida pela escolha de uma das medidas vistas aqui neste mesmo capítulo. O cálculo da confiança é descrito na Equação 2.2. Cada *itemset* frequente de n itens pode gerar 2^n -2 regras. Por exemplo, {News, Sports, Finance} pode gerar regras com os seguintes valores de confiança:

Item	Support Count	Support
{News}	5	0.83
{Finance}	4	0.67
{Entertainment}	1	0.17
{Sports}	2	0.33
Two-itemsets	**Support Count**	**Support**
{News, Finance}	4	0.67
{News, Sports}	2	0.33
{Finance, Sports}	2	0.33
Three-itemsets	**Support Count**	**Support**
{News, Finance, Sports}	2	0.33

Tabela 4 – Cálculo do Suporte para os itens frequentes no exemplo

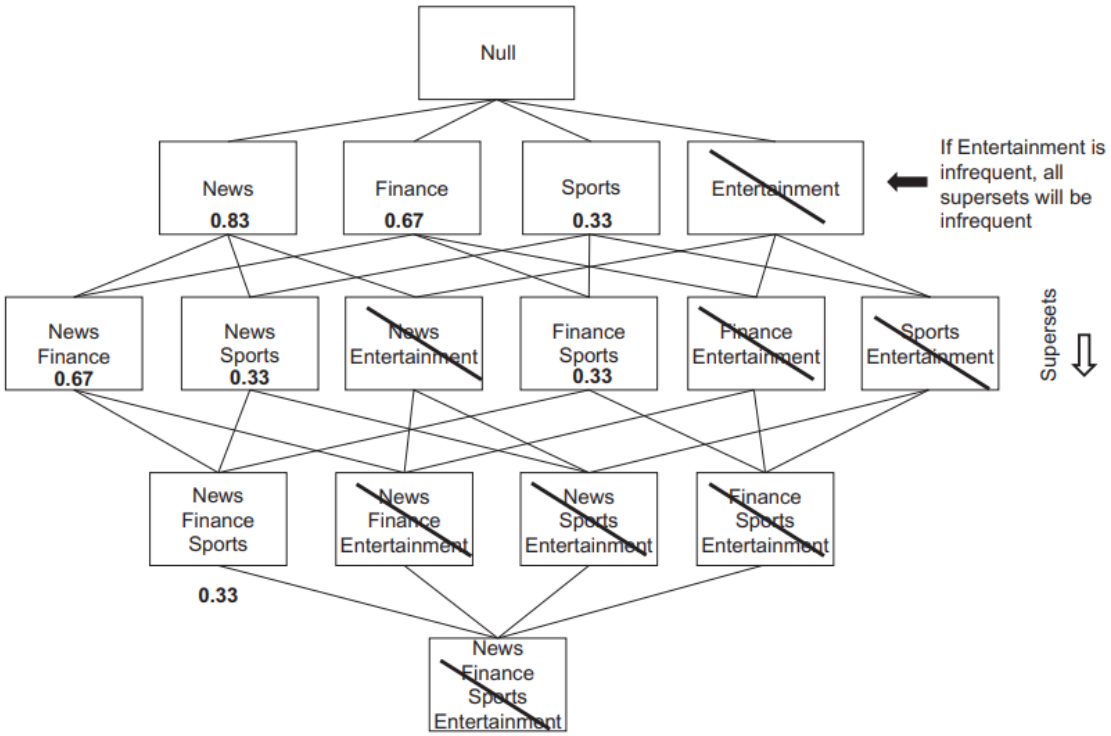

Figura 3 – Exemplo de funcionamento do APRIORI com filtragem dos *itemsets* de acordo com suporte mínimo escolhido

```
{News, Sports} -> {Finance} - 0.33/0.33 = 1.0
{News, Finance} -> {Sports} - 0.33/0.67 = 0.5
{Sports, Finance} -> {News} - 0.33/0.33 = 1.0
{News} -> {Sports, Finance} - 0.33/0.83 = 0.4
{Sports} -> {News, Finance} - 0.33/0.33 = 1.0
{Finance} -> {News, Sports} - 0.33/0.67 = 0.5
```

É possível eliminar/podar regras que potencialmente tenham baixa confiança também usando o APRIORI. Para o *itemset* frequente {News, Finance, Sports}, se a regra {News, Finance} -> {Sports} tem confiança baixa, então qualquer regra dentro do mesmo subconjunto do antecedente terão baixa confiança. Desta forma, todas as regras como {News} -> {Sports, Finance} e {Finance} -> {News, Sports} podem ser descartadas, uma vez que são subconjuntos do antecedente da regra. A razão é que as três regras tem o mesmo numerador no cálculo da confiança, segundo Equação 2.2, que é 0.33, neste caso. O cálculo do denominador depende do suporte do antecedente. Uma vez que o suporte de um subconjunto é sempre maior ou igual ao conjunto, pode-se concluir que todas as seguintes regras dentro do subconjunto de um *itemset* local terão baixa confiança e, portanto, podem ser ignoradas (KOTU; DESHPANDE, 2019).

Todas as regras que cumprem com um valor mínimo de confiança escolhido são consideradas, além de suas medidas de suporte e confiança. Estas regras devem ser avaliadas para determinar se um relacionamento importante deixou de ser considerado, se houve uma ocorrência ocasional ou se a regra confirma relacionamentos intuitivos já conhecidos (KOTU; DESHPANDE, 2019).

2.3 Banco de Dados

Um banco de dados consiste em um lugar de armazenamento de dados. Em SGBDs relacionais, os dados representam uma colação de relações. Já o SGBD é um software que encapsula o banco de dados e controla o armazenamento, organização e busca dos dados.

Dentre os SGBDs disponíveis, neste trabalho é investigado o SGBD Oracle, que possui alguns componentes de interesse, conforme descrito a seguir:

- Código Kernel: Gerencia memória e armazenamento para o sistema.
- Repositório de metadados (geralmente chamado de dicionário de dados): Armazena não apenas os dados da aplicação ou domínio, mas também dados sobre como este armazenamento está acontecendo, de forma a gerenciar melhor o acesso e a disponibilidade das informações. Estes dados geralmente são chamadas de metadados, ou seja, dados sobre os dados.
- Linguagem de consulta: Permite que aplicações acessem os dados armazenadas no banco de dados.

Além de fazer todo o gerenciamento descrito acima, o Oracle (2023a) também disponibiliza visualizações para acesso ao seus metadados, como visualizações estáticas static_data_dictionary. Ela é definida como uma visualização estática, porque apenas se altera quando uma mudança é realizada no dicionário de dados, como quando uma nova

tabela é criada ou quando um usuário recebe novas permissões de acesso. Adicionalmente, esta visualização captura as consultas SQL mais utilizadas baseado em alguns critérios e as informações de estatísticas da visualização V$SQL. Os valores totais representam estatísticas desde o momento de inicialização da instância, enquanto que os valores delta representam as estatísticas entre o BEGIN_INTERVAL_TIME e o END_INTERVAL_TIME, que se encontra na visualização DBA_HIST_SNAPSHOT. Esta visualização é utilizada neste trabalho para a extração dos *datasets*.

Segundo a Oracle (2023b), "Buffer Cache" é uma sub-área em memória dentro da SGA (System Global Area), que armazena cópias de blocos de dados que são lidos dos arquivos de dados. É uma área principal com um gerenciamento para armazenar blocos utilizados recentemente e seu principal objetivo é otimizar leitura e escrita. A primeira vez que uma consulta requer um pedaço de dados a ser trazido, ele é procurado nesta área de *buffer*. Se encontrado, é lido diretamente da memória. Caso contrário, o bloco de dados é copiado do disco antes de ser acessado. O acesso ao dado quando ele está armazenado no *buffer* é mais rápido que se ler o dado do bloco no disco. O atributo "BUFFER_GETS" da visualização mede o quanto de acesso a esta área é realizado e é descrito na seção 3.1.2.

2.4 Trabalhos Relacionados

Para a identificação de trabalhos relacionados, foi realizada uma pesquisa no Portal de Busca Integrada da USP por algumas palavras chave como *troubleshooting*, *outliers* e *isolation forest*. Foram selecionados dois trabalhos relacionados, os quais são descritos a seguir.

Em (NAVARRO; HUET; ROSSI, 2022) descreve-se o HURRA, um sistema criado para reduzir o tempo que os operadores levam para fazer investigação em problemas de rede. É um sistema modular, que compreende de subsistemas para detecção de anomalias (AD), ranqueamento de atributos (FS) e um sistema de entrada de conhecimento prévio (EK). Alguns destes subsistemas utilizam técnicas de aprendizado-não supervisionado, outros utilizam técnicas semi-supervisionadas. Como resultados, os autores argumentam utilizar apenas a detecção de anomalias (não-supervisionado) com hiper-parametrização fixa pode não ser muito preciso. Então, a utilização de um subsistema adicional, como o FS pode melhorar a falta de performance da detecção de anomalias sozinha. Adicionalmente, o subsistema de entrada de conhecimento prévio pode compensar totalmente a falta de acurácia da detecção de anomalias, alcançando na prática o mesmo desempenho de conjuntos AD ideais (por exemplo, usando múltiplos algoritmos e hiper-parametrização).

Em (DUNDJERSKI; TOMAšEVIć, 2022), é criada a ferramenta Automatic database troubleshooting system (ADTS) para monitorar o SGBD SQL Server, principalmente com a motivação dos serviços em nuvem atuais. Cita-se que não se pode confiar apenas em modelos de aprendizado de máquina para detecção de anomalias. Neste sentido, os

autores primeiro utilizam modelos estatísticos de ciência de dados genéricos para investigar aspectos como tempo de execução total da consulta, por exemplo. Então, com base nos modelos genéricos, os autores criam categorias para melhorar a identificação do problema, investigando aspectos como alto consumo de recursos de CPU e disco, alteração de plano de execução, *failover* de instância para outro servidor, aumento de carga de trabalho por meio do aumento de requisições, alto consumo de memória, *lock* quando consultas sobre mesmo objeto, sincronização quando usado paralelismo, contenção de memória temporária, falta de índice, e cliente lento por problema de rede ou outro. Na sequência, um sistema baseado em regras é usado para distinguir as saídas obtidas dos modelos genéricos e categorizados. Cada regra consiste de uma parte antecedente e outra consequente. A parte antecedente consiste de um ou mais fatos e representa as categorias detectadas pelos modelos como fatos. A parte consequente é o resultado que deve ser atingido se a regra é obedecida. Então, a causa raiz final é apresentada após análise do sistema de regras. Os resultados obtidos mostraram que houve melhoria do tempo de processamento. O estudo também menciona que não foram encontradas soluções na indústria e academia que consigam trazer melhor benefício.

Portanto, pode-se perceber que detectar anomalias é uma tarefa complicada e é necessário utilizar de abordagens diferentes em conjunto para melhorar a acurácia do resultado. Existe semelhança no que é encontrado na literatura sobre as dificuldades do aprendizado não-supervisionado, seja para detecção de anomalias ou para se tentar encontrar agrupamentos ou regras de associação com os resultados obtidos neste trabalho, onde as regras extraídas, se vistas de maneira individual, dificultam muito a interpretação do que estava acontecendo no SGBD. Ao se utilizar abordagens adicionais, pode-se encontrar correlações e então melhorar a acurácia na escolha das regras de associação.

Apesar destes dois trabalhos terem algum tipo de relação com a pesquisa a ser realizada, não foram encontrados na literatura trabalhos que utilizam Regras de Associação especificamente para busca de relacionamentos entre consumo de recursos em um SGBD Oracle, principalmente quando o objetivo é apoiar a busca de causa raiz de problemas em aplicações.

3 COLETA DOS DADOS

Neste capítulo é descrito como os dados foram obtidos e utilizados. O objetivo é tentar melhorar a compreensão das causas de problemas e suas relações com os SGBDs analisando estatísticas de consultas do SGBD relacional Oracle, e aplicar algum algoritmo de aprendizado de máquina. O termo *dataset* é utilizado como representação do conjunto de dados, comum na língua inglesa. Na seção 3.1 os dados utilizados para o desenvolvimento do trabalho são descritos e analisados. Na seção 3.2 é detalhada a metodologia empregada no trabalho.

3.1 Extração dos dados das bases relacionais

A extração dos dados das bases relacionais é feita em termos da consulta realizada (seção 3.1.1), da descrição dos atributos extraídos (seção 3.1.2) e da análise dos dados extraídos para verificar suas principais características (seção 3.1.3).

3.1.1 Consulta

Foi feita uma junção das visualizações DBA_HIST_SQLSTAT e DBA_HIST_-SNAPSHOT utilizando como colunas de junção os atributos BEGIN_INTERVAL_TIME e o END_INTERVAL_TIME. A consulta SQL descrita a seguir detalha a junção realizada. É importante destacar que esta consulta é utilizada pelos membros da equipe de suporte ao SGBD Oracle, na existência de uma situação na qual se deseja extrair conhecimento. Atualmente, este processo é realizado utilizando a ferramenta Microsoft Excel e tabelas *pivot*.

```
col SNAP_TIME for a12
col SQL_ID for a13
col EXECUTIONS_DELTA for 999,999,999,999 heading EXECUTIONS
col DISK_READS_DELTA for 999,999,999,999,999 heading DISK_READS
col BUFFER_GETS_DELTA for 999,999,999,999,999,999 heading BUFFER_GETS
col ROWS_PROCESSED_DELTA for 999,999,999,999 heading ROWS_PROCESSED
col CPU_TIME_DELTA for 999,999 heading CPU_TIME
col IOWAIT_DELTA for 999,999,999,999 heading IO_WAIT
col APWAIT_DELTA for 999,999,999,999 heading AP_WAIT
col CCWAIT_DELTA for 999,999,999,999 heading CONCURRENCY_WAIT
col ELAPSED_TIME_DELTA for 999,999,999,999 heading ELAPSED_TIME
col AVG_GET for 999,999,999,999,999.9
col AVG_RD for 999,999,999,999,999.9
col AVG_TM for 9,999,999,999.9 heading AVG_TM(MS)
```

```
set colsep |
set pages 22222
set line 450

column start_snap new_value start_snap noprint;
column end_snap new_value end_snap noprint;
set heading on

SELECT
    to_char(end_interval_time, 'MM-DD hh24:mi')  snap_time,
    s.instance_number,
    sql_id,
    executions_delta,
    disk_reads_delta,
    disk_reads_delta / executions_delta          avg_rd,
    buffer_gets_delta,
    buffer_gets_delta / executions_delta         avg_get,
    rows_processed_delta,
    cpu_time_delta / 1000 / 1000                 cpu_time_delta,
    iowait_delta,
    apwait_delta,
    ccwait_delta,
    elapsed_time_delta / 1000 / 1000             elapsed_time_delta,
    elapsed_time_delta / 1000 / executions_delta avg_tm,
    plan_hash_value                              plan,
    physical_write_requests_delta,
    physical_read_requests_delta,
    physical_write_bytes_delta,
    physical_read_bytes_delta,
    direct_writes_delta,
    sorts_delta
FROM
    dba_hist_sqlstat  s,
    dba_hist_snapshot h
WHERE
    executions_delta > 0
    AND s.snap_id = h.snap_id
    AND s.instance_number = h.instance_number
    AND h.begin_interval_time >= TO_DATE('2023-05-24 00:00', 'YYYY-MM-DD hh24:mi')
    AND h.end_interval_time <= TO_DATE('2023-05-31 00:00', 'YYYY-MM-DD hh24:mi')
ORDER BY 1;
```

Após a extração dos dados, realizada através de um *spool* para um arquivo texto, o conjunto de dados foi convertido para um arquivo no formato *CSV*, separado por vírgula, de maneira que pudesse ser utilizado em um *dataframe* do Pandas.

3.1.2 Atributos extraídos e seus significados

A seguir são listados os atributos extraídos pela consulta descrita na seção 3.1.1 e suas respectivas descrições, segundo a documentação da visualização DBA_HIST_SQLSTAT pela Oracle (2023c). É importante mencionar que algumas unidades de medida estão diferentes (microssegundos, segundos) e também que alguns atributos são calculados a partir de outros. Estes aspectos são destacados juntamente com a descrição dos atributos extraídos.

- END_INTERVAL_TIME (as SNAP_TIME): Horário ao final do intervalo, ou seja, horário real no qual o intervalo foi registrado;
- INSTANCE_NUMBER: Número da instância para o intervalo;
- SQL_ID: Identificador SQL do cursor pai na área de cache de biblioteca (consulta já analisada pelo otimizador);
- EXECUTIONS_DELTA: Diferença (delta) do número de execuções para o objeto mencionado desde que ele foi colocado no cache de biblioteca, considerando um intervalo de tempo estabelecido;
- DISK_READS_DELTA: Diferença (delta) do número de leituras em disco para este cursor filho;
- DISK_READS_DELTA/EXECUTIONS_DELTA as AVG_RD: Divisão entre dois atributos descritos anteriormente;
- BUFFER_GETS_DELTA: Diferença (delta) do número de acessos ao *buffer* para este cursor filho;
- BUFFER_GETS_DELTA/EXECUTIONS_DELTA as AVG_GET: Divisão entre dois atributos já descritos anteriormente;
- ROWS_PROCESSED_DELTA: Diferença (delta) do número de linhas que as consultas já analisadas pelo otimizador retornam;
- CPU_TIME_DELTA/1000/1000 as CPU_TIME_DELTA: Diferença (delta) do tempo de CPU (convertido de microssegundos para segundos) usado por este cursor para analisar/executar/coletar;
- IOWAIT_DELTA: Diferença (delta) do tempo de uso de entrada/saída por usuário (em microssegundos);

- APWAIT_DELTA: Diferença (delta) do tempo de espera de aplicação (em microssegundos);

- CCWAIT_DELTA: Diferença (delta) do tempo de espera de concorrência (em microssegundos);

- ELAPSED_TIME_DELTA/1000/1000 as ELAPSED_TIME_DELTA: Diferença (delta) do tempo total decorrido (convertido de microssegundos para segundos) usado por este cursor para analisar/executar/coletar;

- ELAPSED_TIME_DELTA/1000 / EXECUTIONS_DELTA as AVG_TM: Divisão entre dois atributos já descritos anteriormente;

- PLAN_HASH_VALUE as PLAN: Representação numérica do plano de consulta para o cursor. Comparar este valor auxilia na identificação de mudança de plano sem precisar analisar o plano linha a linha;

- PHYSICAL_WRITE_REQUESTS_DELTA: Diferença (delta) do número de requisições de escrita (entrada e saída), pela consulta monitorada;

- PHYSICAL_READ_REQUESTS_DELTA: Diferença (delta) do número requisições de leitura (entrada e saída), pela consulta monitorada;

- PHYSICAL_WRITE_BYTES_DELTA: Diferença (delta) do número de bytes escritos em disco pela consulta monitorada;

- PHYSICAL_READ_BYTES_DELTA: Diferença (delta) do número de bytes lidos do disco pela consulta monitorada;

- DIRECT_WRITES_DELTA: Diferença (delta) do número de escritas diretas (não utilizam o cache do sistema de arquivos, dependendo do sistema operacional e configurações);

- SORTS_DELTA: Diferença (delta) do número de operações de ordenação que foram realizadas para o cursor filho.

3.1.3 Análise Inicial dos Atributos

Foram gerados dois *datasets* coletados em momentos diferentes, representando situações de problemas diferentes, conforme descrito a seguir. Ambos os *datasets* contêm dados estatísticos de aproximadamente sete dias. Geralmente, considera-se um intervalo de três dias antes da ocorrência do problema e três dias após.

- *Dataset* 1: Compreendendo o momento em que houve uma janela de manutenção noturna. A coleta foi feita considerando o período de 25/04/2023 00:15 a 30/05/2023 23:45. Foram geradas 93546 linhas e 23 colunas. Uma observação importante a respeito deste *dataset* é que as estatísticas que ele contém foram

coletadas em tabelas de aplicação e, durante o dia, a aplicação apresentou muita lentidão em consultas. Qualquer alteração de mili-segundos de diferença no tempo de resposta pode impactar as aplicações, principalmente se estas consultas são utilizadas com muita frequência. Supõe-se que essa lentidão seja devido às alterações nos planos de execução realizadas pela coleta;

- *Dataset* 2: Compreendendo o momento em que houve lentidão da aplicação, mas sem aparente intervenção antecedente no SGBD. A coleta foi feita considerando o período de 30/07/2023 00:15 a 06/08/2023 15:00. Foram geradas 122429 linhas e 23 colunas. Supõe-se que a lentidão tenha sido causada por um problema da aplicação não estar finalizando as transações no SGBD, gerando *locks* em tabelas usadas durante o mesmo momento. Aparentemente, o problema foi gerado por um *script* que realiza limpeza das tabelas.

Na Figura 4 é ilustrada a matriz de correlação entre atributos nos dois *datasets*. Analisando-se as matrizes geradas, nota-se que selecionar atributos é um desafio, uma vez que para cada momento de coleta, o problema a ser analisado pode ser diferente. Outro fato interessante que pode ser observado pelos resultados obtidos é que os *datasets* contêm dados referentes a um intervalo de tempo grande, ou seja, vários dias. Como resultado, espera-se que o atributo AP_WAIT, que representa o tempo em espera de aplicação, tenha uma correlação maior com o atributo ELAPSED_TIME no *dataset* 2.

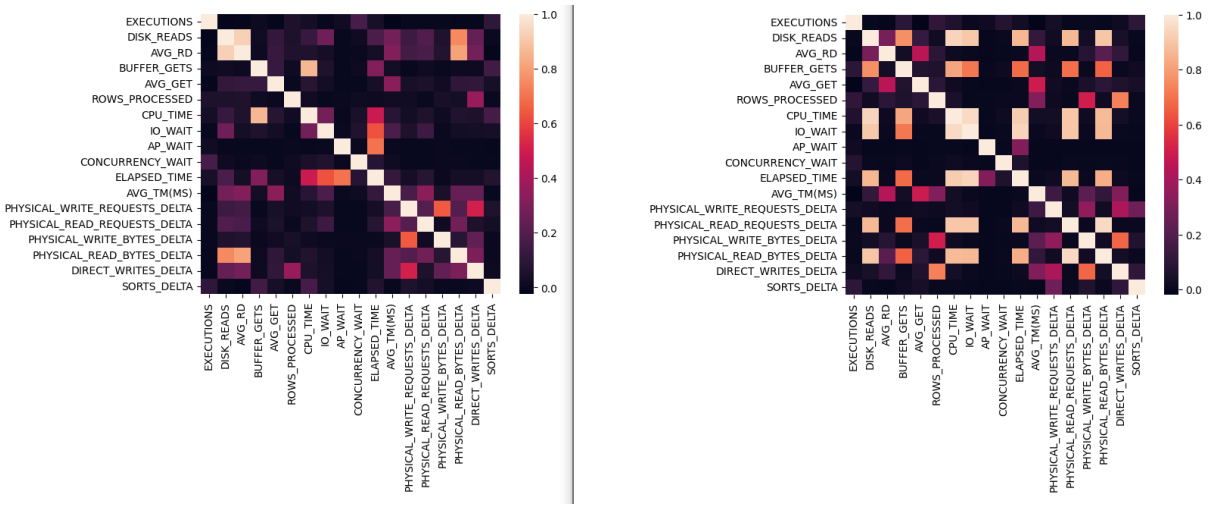

Figura 4 – Matriz de correlação entre atributos no *dataset* 1 (à esquerda) e *dataset* 2 (à direita)

Para solucionar o problema destacado anteriormente e possibilitar a obtenção de um melhor resultado decorrente do desenvolvimento do presente trabalho, foram gerados novos *datasets*. Esses novos *datasets* contêm dados apenas do dia em que o problema ocorreu, conforme descrito a seguir.

- *Dataset* 3: Corresponde à redução do *dataset* 1 de 7 dias para 1 dia. Contém dados considerando o período de 26/05/2023 09:00 a 26/05/2023 18:00. Armazena 4363 linhas e 23 colunas;

- *Dataset* 4: Corresponde à redução do *dataset* 2 de 7 dias para 1 dia. Contém dados considerando o período de 02/02/2023 14:00 a 02/08/2023 18:15. Armazena 3025 linhas e 23 colunas.

Conforme ilustrado na Figura 5, verifica-se que, com apenas um dia de coleta, a investigação a ser feita neste trabalho torna-se mais bem representada. A correlação entre os atributos ELAPSED_TIME e AP_WAIT se torna maior. Portanto, em alguns experimentos são utilizados estes *datasets* reduzidos para melhor precisão na investigação do problema.

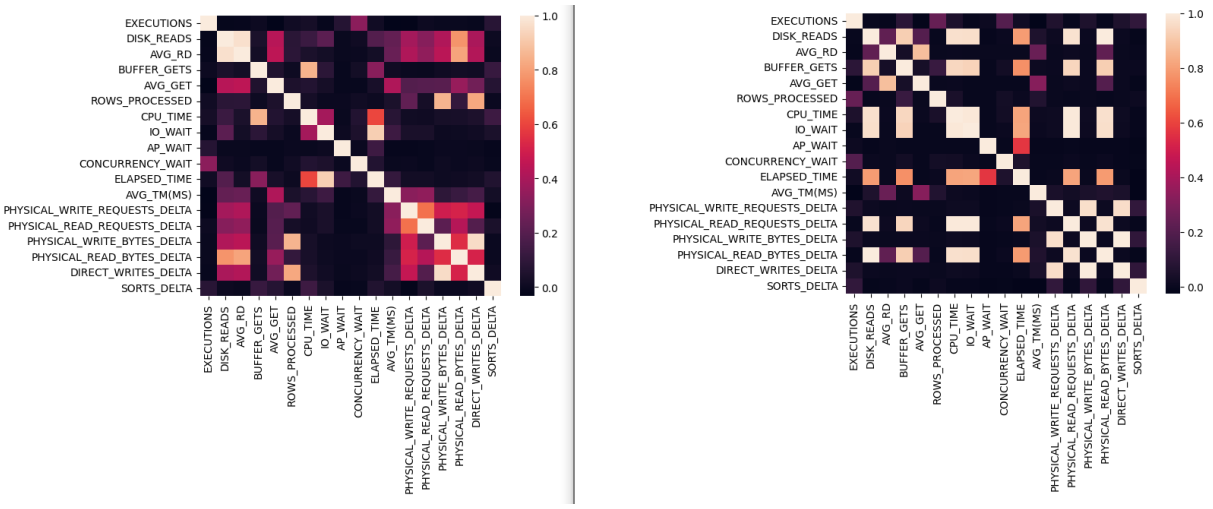

Figura 5 – Matriz de correlação entre atributos no *dataset* 3 (à esquerda) e *dataset* 4 (à direita)

3.2 Metodologia

A metodologia proposta para o desenvolvimento do trabalho é composta pelas fases sumarizadas na Figura 6. A discussão das duas primeiras fases dentro do contexto do trabalho é feita a seguir. As demais fases são discutidas nos próximos capítulos.

Na primeira fase, *identificação do problema*, deseja-se minerar metadados de um SGBD relacional para fins de exploração e melhor compreensão dos problemas que podem acontecer no dia-a-dia de uma aplicação. A coleta dos dados foi realizada manualmente. Para a extração das regras de associação, devem ser utilizados o *dataset* 3 e o *dataset* 4, que se tratam de *datasets* resumidos contendo apenas estatísticas do dia em que o problema foi reportado.

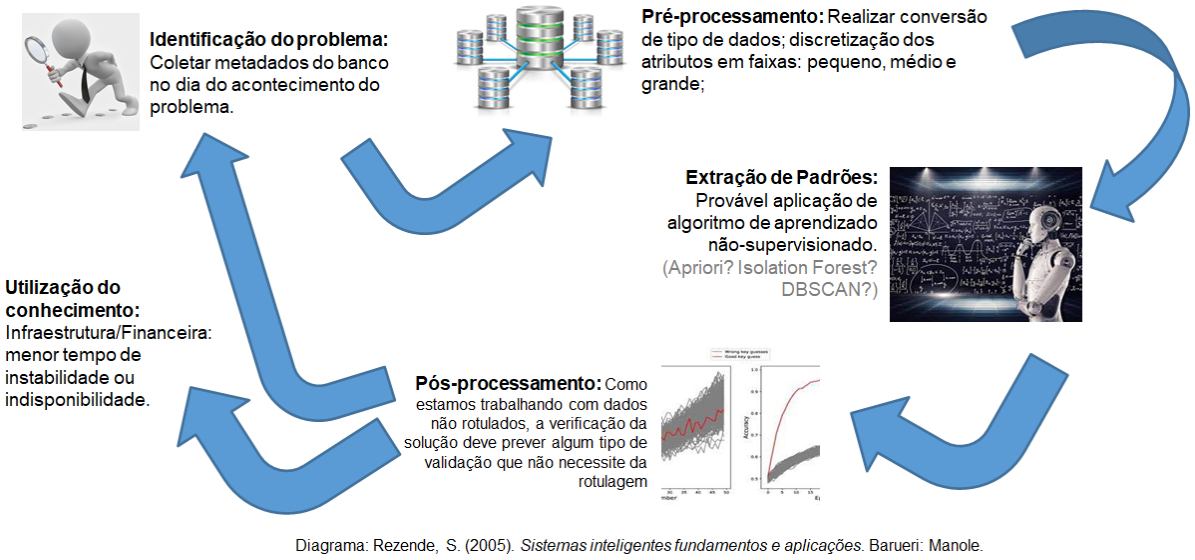

Figura 6 – Metodologia apresentando as principais fases do ciclo deste trabalho

Na etapa de *pré-processamento dos dados*, foi utilizado o método *replace* do Pandas para remover as vírgulas e retornar os dados com o tipo de dado numérico, que é o tipo de dado adequado para as análises a serem realizadas. Essa conversão de tipo de dados foi realizada porque, ao serem carregados no *dataframe* Pandas, os dados foram definidos como tipo de dados object, inviabilizando operações numéricas sobre os mesmos.

Adicionalmente, segundo Ramírez-Gallego *et al.* (2016), dados podem ser apresentados em diversos formatos, como categóricos, numéricos, contínuos. Dados categóricos não são ordenados, mas dados numéricos ou contínuos podem ser ordenados ou representar dados ordinais. Os dados extraídos para serem utilizados neste trabalho contêm valores quantitativos para a maioria dos atributos interessantes de serem estudados, conforme ilustrado na Figura 7.

Os algoritmos de aprendizado de máquina dependem do domínio da aplicação e do tipo de dados. O processo de discretização se mostra como uma importante fase de pré-processamento para mineração de dados. Basicamente, pode-se resumir esse processo em traduzir dados quantitativos para qualitativos, em forma de intervalos de valores que não se sobreponham. Pode-se pensar também no processo de discretização como um processo de redução dos dados, pelo fato de conseguir traduzir grandes domínios de valores em faixas categóricas. Na seção 4.1 é detalhado o processo de discretização realizado nos dados.

Na fase de *extração de padrões*, aplica-se um algoritmo de aprendizado não supervisionado para a extração de padrões. Neste trabalho, emprega-se o algoritmo APRIORI. Os resultados referentes a esta etapa são detalhados na seção 4.2.

	SNAP_TIME	INSTANCE_NUMBER	SQL_ID	EXECUTIONS	DISK_READS	AVG_RD	BUFFER_GETS	AVG_GET	ROWS_PROCESSED	CPU_TIME	IO_
0	8/2/2023 14:00	2	a3jpjzh7mtwwp	103	0	0	672	6.5	140	0	
1	8/2/2023 14:00	2	g9btkf9d2m9q8	1	0	0	75	75	0	2	
2	8/2/2023 14:00	2	d2sx5c3539zqp	4	68	17	121	30.3	36	0	3
3	8/2/2023 14:00	2	8swypbbr0m372	297	0	0	2,677	9	982	0	
4	8/2/2023 14:00	2	9g485acn2n30m	79	0	0	276	3.5	0	0	
...	...	...	...	...	...	...	...	...	...	...	
3020	8/2/2023 18:00	2	baf6hfqvpwvfb	3	0	0	76,143	25,381.00	5,009	13	
3021	8/2/2023 18:00	2	6vqh7bwu1hvqq	14	14	1	249	17.8	27	3	1
3022	8/2/2023 18:00	2	5rs0w65bbqtqm	1	0	0	142	142	0	5	
3023	8/2/2023 18:00	2	9tgj4g8y4rwy8	915	0	0	3,660	4	915	0	
3024	8/2/2023 18:15	2	gd28w82ct6rva	146	0	0	463	3.2	146	0	

3025 rows × 23 columns

Figura 7 – *Dataset* antes de passar por processo de discretização

Com relação ao *pós-processamento*, é necessário se realizar uma investigação relacionada aos resultados obtidos após a aplicação do algoritmo APRIORI. Os resultados referentes a esta etapa são detalhados no Capítulo 5.

4 DISCRETIZAÇÃO DOS DADOS E APLICAÇÃO DO ALGORITMO APRIORI

Neste capítulo são descritos aspectos de desenvolvimento do trabalho quanto à discretização dos dados (seção 4.1) da etapa de pré-processamento e à aplicação do algoritmo de mineração de dados APRIORI para a extração de padrões (seção 4.2).

4.1 Discretização dos Dados

A discretização foi realizada por meio da interface de pré-processamento do pacote KBinsDiscretizer pelo scikit-learn (2023). Cada atributo foi dividido em três novos atributos dependendo da faixa de valores (pequenos, médios e grandes). A lista de atributos discretizados é exibida a seguir:

- IO_WAIT
- AP_WAIT
- CONCURRENCY_WAIT
- EXECUTIONS
- DISK_READS
- BUFFER_GETS
- CPU_TIME
- ELAPSED_TIME
- ROWS_PROCESSED
- PHYSICAL_WRITE_REQUESTS_DELTA
- PHYSICAL_WRITE_BYTES_DELTA
- PHYSICAL_READ_REQUESTS_DELTA
- PHYSICAL_READ_BYTES_DELTA
- DIRECT_WRITES_DELTA
- SORTS_DELTA

Os atributos AVG_RD, AVG_GET e AVG_TM(MS) foram removidos porque na consulta original são calculados a partir de outros atributos.

	index	SNAP_TIME	INSTANCE_NUMBER	SQL_ID	IO_WAIT_low	IO_WAIT_medium	IO_WAIT_high	AP_WAIT_low	AP_WAIT_medium	AP_WAIT_high
0	0	8/2/2023 14:00	2	a3jpjzh7mtwwp	0	1	0	0	1	0
1	1	8/2/2023 14:00	2	g9btkf9d2m9q8	0	1	0	0	1	0
2	2	8/2/2023 14:00	2	d2sx5c3539zqp	0	1	0	0	1	0
3	3	8/2/2023 14:00	2	8swypbbr0m372	0	1	0	0	1	0
4	4	8/2/2023 14:00	2	9g485acn2n30m	0	1	0	0	1	0
...	...	...	...	...	...	...	...	...	...	...
3020	3020	8/2/2023 18:00	2	baf6hfqvpwvfb	0	1	0	0	1	0
3021	3021	8/2/2023 18:00	2	6vqh7bwu1hvqq	0	1	0	0	1	0
3022	3022	8/2/2023 18:00	2	5rs0w65bbqtqm	0	1	0	0	1	0
3023	3023	8/2/2023 18:00	2	9tgj4g8y4rwy8	0	1	0	0	1	0
3024	3024	8/2/2023 18:15	2	gd28w82ct6rva	0	1	0	0	1	0

3025 rows × 58 columns

Figura 8 – *Dataset* após passar por processo de discretização

A saída do KBinsDiscretizar é em formato de ponto flutuante (1.0 ou 0.0). Neste contexto, ela foi convertida para o formato inteiro. Na Figura 8 é ilustrado um exemplo contento algumas tuplas do *dataset* discretizado.

Nas Tabelas 5 a 8 são ilustradas a porcentagem de dados em cada faixa de valores (baixo, médio e alto) ao utilizar o KBinsDiscretizer nos quatro *datasets* estudados. Os parâmetros e estratégias utilizados são:

- Uniform (Todas as faixas em cada atributo com tamanhos idênticos): com parametrizações n_bins=3, encode='onehot-dense' e strategy='uniform';

- K-Means (Valores em cada faixa têm o mesmo centro próximo de um cluster K-Means de uma dimensão): com parametrizações n_bins=3, encode='onehot-dense' e strategy='kmeans'.

Atributo	uniform (%)			k-means (%)		
	baixo	médio	alto	baixo	médio	alto
IO_WAIT	99,98	0,02	0	99,57	0,41	0,02
AP_WAIT	99,99	0,01	0	99,99	0,01	0
CONCURRENCY_WAIT	99,99	0,01	0	99,99	0,01	0
EXECUTIONS	99,96	0,03	0,01	97,37	2,40	0,23
DISK_READS	99,96	0,03	0,01	99,88	0,09	0,03
BUFFER_GETS	99,36	0,18	0,46	98,19	1,20	0,61
CPU_TIME	99,25	0,73	0,02	97,48	1,78	0,73
ELAPSED_TIME	99,99	0,01	0	98,47	1,52	0,01
ROWS_PROCESSED	99,99	0	0,01	99,98	0,01	0,01
PHYSICAL_WRITE_REQUESTS_DELTA	100	0	0	100	0	0
PHYSICAL_WRITE_BYTES_DELTA	100	0	0	99,99	0,01	0
PHYSICAL_READ_REQUESTS_DELTA	100	0	0	100	0	0
PHYSICAL_READ_BYTES_DELTA	99,97	0,03	0,01	99,93	0,06	0,02
DIRECT_WRITES_DELTA	99,99	0,01	0	99,98	0,02	0
SORTS_DELTA	98,91	0,76	0,33	97,51	1,86	0,63

Tabela 5 – Dataset 1: Porcentagem dos atributos discretizados nos bins baixo, médio e alto utilizando as estratégias uniform e kmeans no KBinsDiscretizer

Atributo	uniform (%)			k-means (%)		
	baixo	médio	alto	baixo	médio	alto
IO_WAIT	99,97	0,03	0	99,91	0,09	0
AP_WAIT	99,98	0,01	0,01	99,98	0,01	0,01
CONCURRENCY_WAIT	100	0	0	99,99	0,01	0
EXECUTIONS	99,80	0,18	0,01	96,61	3,16	0,24
DISK_READS	99,88	0,08	0,04	99,65	0,25	0,10
BUFFER_GETS	99,83	0,16	0,01	99,66	0,20	0,14
CPU_TIME	99,89	0,07	0,04	99,73	0,19	0,08
ELAPSED_TIME	99,94	0,05	0	99,82	0,12	0,05
ROWS_PROCESSED	100	0	0	100	0	0
PHYSICAL_WRITE_REQUESTS_DELTA	99,71	0,23	0,06	98,72	0,99	0,29
PHYSICAL_WRITE_BYTES_DELTA	100	0	0	100	0	0
PHYSICAL_READ_REQUESTS_DELTA	99,92	0,04	0,03	99,82	0,13	0,06
PHYSICAL_READ_BYTES_DELTA	99,91	0,05	0,04	99,73	0,20	0,07
DIRECT_WRITES_DELTA	100	0	0	99,57	0,43	0
SORTS_DELTA	98,30	0,92	0,78	97,79	1,28	0,93

Tabela 6 – Dataset 2: Porcentagem dos atributos discretizados nos bins baixo, médio e alto utilizando as estratégias uniform e kmeans no KBinsDiscretizer

Atributo	uniform (%)			k-means (%)		
	baixo	médio	alto	baixo	médio	alto
IO_WAIT	99,91	0,07	0,02	99,08	0,89	0,02
AP_WAIT	99,54	0,23	0,23	99,17	0,60	0,23
CONCURRENCY_WAIT	99,98	0	0,02	99,34	0,64	0,02
EXECUTIONS	99,45	0,34	0,21	96,54	3,14	0,32
DISK_READS	99,91	0,07	0,02	99,91	0,07	0,02
BUFFER_GETS	99,59	0,16	0,25	98,33	1,28	0,39
CPU_TIME	99,17	0,34	0,48	97,48	1,99	0,53
ELAPSED_TIME	99,95	0,02	0,02	98,03	1,95	0,02
ROWS_PROCESSED	99,95	0	0,05	99,95	0,02	0,02
PHYSICAL_WRITE_REQUESTS_DELTA	99,98	0	0,02	97,94	2,04	0,02
PHYSICAL_WRITE_BYTES_DELTA	99,93	0	0,07	99,93	0,02	0,05
PHYSICAL_READ_REQUESTS_DELTA	99,56	0,34	0,09	97,46	2,41	0,14
PHYSICAL_READ_BYTES_DELTA	99,95	0,02	0,02	99,95	0,02	0,02
DIRECT_WRITES_DELTA	99,93	0	0,07	99,65	1,28	0,07
SORTS_DELTA	98,62	1,19	0,18	98,26	1,56	0,18

Tabela 7 – Dataset 3: Porcentagem dos atributos discretizados nos bins baixo, médio e alto utilizando as estratégias uniform e kmeans no KBinsDiscretizer

Atributo	uniform (%)			k-means (%)		
	baixo	médio	alto	baixo	médio	alto
IO_WAIT	99,54	0,26	0,20	99,44	0,30	0,26
AP_WAIT	99,57	0,23	0,20	99,40	0,33	0,26
CONCURRENCY_WAIT	99,83	0,13	0,03	99,70	0,26	0,03
EXECUTIONS	99,44	0,50	0,07	94,35	5,09	0,56
DISK_READS	99,40	0,43	0,17	99,40	0,43	0,17
BUFFER_GETS	99,54	0,30	0,17	99,50	0,36	0,13
CPU_TIME	99,47	0,36	0,17	99,47	0,40	0,13
ELAPSED_TIME	99,21	0,60	0,20	99,01	0,69	0,30
ROWS_PROCESSED	99,40	0,26	0,33	99,40	0,36	0,23
PHYSICAL_WRITE_REQUESTS_DELTA	99,27	0,46	0,26	99,27	0,50	0,23
PHYSICAL_WRITE_BYTES_DELTA	99,21	0,53	0,26	99,21	0,56	0,23
PHYSICAL_READ_REQUESTS_DELTA	99,47	0,36	0,17	99,47	0,36	0,17
PHYSICAL_READ_BYTES_DELTA	99,40	0,43	0,17	99,40	0,43	0,17
DIRECT_WRITES_DELTA	99,21	0,53	0,26	99,21	0,56	0,23
SORTS_DELTA	97,02	1,65	1,32	96,73	1,79	1,49

Tabela 8 – Dataset 4: Porcentagem dos atributos discretizados nos bins baixo, médio e alto utilizando as estratégias uniform e kmeans no KBinsDiscretizer

4.2 Aplicação do Algoritmo APRIORI

Foram realizados diversos experimentos para investigar a aplicação do algoritmo APRIORI. Esses experimentos são sumarizados a seguir.

- Experimento 1: Realizado sobre o *dataset* 1, sendo que não foram removidos atributos após a discretização. Esse experimento é descrito na seção 4.2.1.
- Experimento 2: Realizado sobre o *dataset* 1, com a remoção de atributos nas faixas baixas (_low). Esse experimento é descrito na seção 4.2.2.
- Experimento 3: Realizado sobre o *dataset* 3, com a remoção de atributos nas faixas baixas (_low). Esse experimento é descrito na seção 4.2.3.
- Experimento 4: Realizado sobre o *dataset* 2, com a remoção de atributos nas faixas baixas (_low). Esse experimento é descrito na seção 4.2.4.
- Experimento 5: Realizado sobre o *dataset* 4, com a remoção de atributos nas faixas baixas (_low). Esse experimento é descrito na seção 4.2.5.

Os experimentos são descritos na ordem na qual eles foram realizados, visando detalhar a evolução do trabalho desenvolvido.

4.2.1 Experimento 1

O primeiro experimento foi realizado com o *dataset* 1. Não foram removidos atributos após a discretização, ou seja, todos os atributos foram mantidos. Em detalhes, nenhuma linha (transação) possuía a soma de todos os atributos igual a zero (transação vazia), então nenhuma linha foi removida do *dataset* após o processo de discretização.

Foram realizadas diferentes execuções desse experimento. Inicialmente foi aplicado o suporte mínimo com valor 0.03 e confiança mínima com valor 0.75 e max_length=5. O experimento não pode ser executado por falta de memória. Depois foram realizadas mais duas execuções. Uma primeira aumentando o suporte mínimo para valor igual a 0.5 e uma segunda aumentando o suporte mínimo para o valor igual a 0.8. Nessas duas execuções, manteve-se a confiança mínima com valor 0.75. Estes hiper-parâmetros foram definidos inicialmente com base ao que geralmente se utilizou nos exemplos vistos em aula do curso, mas nenhuma execução foi possível de ser realizada devido ao tamanho do *dataset*. Isso significa que deve haver remoção de atributos.

4.2.2 Experimento 2

O segundo experimento foi realizado com o *dataset* 1, porém com remoção de atributos nas faixas baixas (_low). Os atributos representando valores pequenos foram

removidos porque o interesse consiste em investigar a geração de regras para os atributos nos quais existem picos. Esses atributos representam aqueles na faixa de valores médios e altos.

Após a remoção dos atributos nas faixas baixas, observou-se que eles eram a maioria. Adicionalmente, observou-se que o *dataset* passou a conter muitas transações vazias, ou seja, linhas com a soma das colunas igual a zero. Desde que o algoritmo APRIORI não pode trabalhar com transações vazias, foi realizada a remoção destas linhas. O *dataset* reduziu bastante, como mostrado na Figura 9.

	IO_WAIT_medium	IO_WAIT_high	AP_WAIT_medium	AP_WAIT_high	CONCURRENCY_WAIT_medium	CONCURRENCY_WAIT_high
10	0	0	0	0	0	0
14	0	0	0	0	0	0
44	0	0	0	0	0	0
168	0	0	0	0	0	0
210	0	0	0	0	0	0
...	...	...	...	...	...	...
93478	0	0	0	0	0	0
93488	0	0	0	0	0	0
93489	0	0	0	0	0	0
93507	1	0	0	0	0	0
93539	0	0	0	0	0	0

6552 rows × 30 columns

Figura 9 – Algumas colunas do *dataset* após passar por processo de redução de atributos representando faixas pequenas e médias e remoção de linhas com transações zeradas. Na figura não são exibidas todas as colunas. Porém, não existem linhas sem que ao menos uma coluna esteja preenchida com o valor 1.

Sumário do *dataset* para este experimento:

- Período SNAP_TIME *Dataset* 1: 24/05/23 0:15 às 30/05/23 23:45
- Tamanho original: 93546 linhas x 23 colunas
- Remoção de atributos calculados a partir de outros / irrelevantes: Sim
- Remoção de atributos _low (faixa baixa): Sim
- Remoção de linhas com soma igual a zero (transações vazias): Sim
- Tamanho *dataset* após pré-processamento acima: 6552 linhas × 30 colunas
- Suporte mínimo utilizado no APRIORI: 0.01 (1%)

O *dataset* gerado é agrupado por SNAP_TIME e SQL_ID como resultado da extração dos dados do SGBD. Apesar disso, foram definidos os atributos SNAP_TIME e

SQL_ID no dataframe Pandas como formação de seu índice, conforme detalhado no código a seguir. O objetivo foi encontrar o SQL_ID e em qual SNAP_TIME o valor máximo foi atingido para cada atributo.

```
#configurando indexes para verificar em quais momentos e
#qual a consulta atinge picos (por consulta)
dfindex = df2.set_index(['SNAP_TIME', 'SQL_ID'])
atributos_max_ind = dfindex.idxmax()
print(atributos_max_ind)
valores_atributos_max_ind = dfindex.max().astype('int')
print(valores_atributos_max_ind)

#colocando resultado em um outro dataframe
df_max = pd.DataFrame([atributos_max_ind,valores_atributos_max_ind])
df_max.T.columns=['SQL_ID_SNAP_TIME','count']
df_max.T
```

Na Tabela 9 são exibidos valores que mostram em qual momento e qual foi a consulta que mais exigiu de um determinado atributo.

Atributo do dataset (antes discretização)	**SQL_ID e SNAP_TIME**	**Valor máximo do atributo no intervalo**
EXECUTIONS	(b9nbhsbx8tqz5, 5/30/2023 0:15)	9739065
DISK_READS	(25by9zbr3gcpg, 5/26/2023 17:30)	71682909
BUFFER_GETS	(37apnmb6vfuar, 5/27/2023 23:15)	1247400827
ROWS_PROCESSED	(g4wbndxmkk7ga, 5/30/2023 23:45)	290891940
CPU_TIME	(gm2brdhxrqsw6, 5/24/2023 4:15)	3428
IO_WAIT	(5jwhq4w2fy5by, 5/26/2023 15:15)	17363748984
AP_WAIT	(bdqgsd32fxwan, 5/30/2023 12:00)	54543065557
CONCURRENCY_WAIT	(5q6mm4m9fmxq9, 5/24/2023 4:30)	1630671893
ELAPSED_TIME	(bdqgsd32fxwan, 5/30/2023 12:00)	54599
PHYSICAL_WRITE_REQUESTS_DELTA	(gf6qv5uyu6c5a, 5/30/2023 16:30)	229643
PHYSICAL_READ_REQUESTS_DELTA	(18rkx81bggaub, 5/27/2023 8:45)	13576418
PHYSICAL_WRITE_BYTES_DELTA	(gf6qv5uyu6c5a, 5/30/2023 16:30)	241000000000
PHYSICAL_READ_BYTES_DELTA	(25by9zbr3gcpg, 5/26/2023 17:30)	587000000000
DIRECT_WRITES_DELTA	(4nkba3rx4tkdr, 5/26/2023 18:30)	4698546
SORTS_DELTA	(dd5kafytrvh36, 5/24/2023 17:15)	65457

Tabela 9 – Valor máximo do atributo no intervalo/consulta

Na Tabela 10 são listados alguns dos *itemsets* mais frequentes encontrados no presente experimento.

support	**itemsets**
0.343254	(EXECUTIONS_medium)
0.264957	(SORTS_DELTA_medium)
0.254731	(CPU_TIME_medium)
0.216728	(ELAPSED_TIME_medium)
0.171398	(BUFFER_GETS_medium)
0.161630	(BUFFER_GETS_medium, CPU_TIME_medium)
0.104396	(CPU_TIME_high)
0.103480	(CPU_TIME_high, ELAPSED_TIME_medium)
0.094475	(ELAPSED_TIME_medium, SORTS_DELTA_medium)
0.091880	(ELAPSED_TIME_medium, CPU_TIME_high, SORTS_DELTA_medium)
0.091880	(CPU_TIME_high, SORTS_DELTA_medium)
0.090354	(SORTS_DELTA_high)
0.086691	(BUFFER_GETS_high)
0.078755	(CPU_TIME_high, BUFFER_GETS_high)
0.078755	(ELAPSED_TIME_medium, BUFFER_GETS_high)
0.078297	(CPU_TIME_high, ELAPSED_TIME_medium, BUFFER_GETS_high)
0.073871	(SORTS_DELTA_medium, BUFFER_GETS_high)
0.073107	(ELAPSED_TIME_medium, SORTS_DELTA_medium, BUFFER_GETS_high)
0.072955	(CPU_TIME_high, SORTS_DELTA_medium, BUFFER_GETS_high)
0.072955	(ELAPSED_TIME_medium, CPU_TIME_high, SORTS_DELTA_medium, BUFFER_GETS_high)
0.064103	(ELAPSED_TIME_medium, CPU_TIME_medium)
0.058303	(IO_WAIT_medium)
0.058303	(ELAPSED_TIME_medium, IO_WAIT_medium)
0.048993	(ELAPSED_TIME_medium, IO_WAIT_medium, CPU_TIME_medium)
0.048993	(IO_WAIT_medium, CPU_TIME_medium)

Tabela 10 – Alguns dos Itemsets mais frequentes gerados pelo experimento 2

Adicionalmente, na Tabela 11 são descritas as regras com suas medidas, ordenadas pela medida lift.

antecedents	consequents	ant supp	cons supp	supp	conf	lift	lev	conv	zhangs
(IO_WAIT_medium)	(ELAPSED_TIME_medium, CPU_TIME_medium)	0.06	0.06	0.05	0.84	13.11	0.05	5.86	0.98
(ELAPSED_TIME_medium, BUFFER_GETS_high)	(CPU_TIME_high, SORTS_DELTA_medium)	0.08	0.09	0.07	0.93	10.08	0.07	12.33	0.98
(CPU_TIME_high, BUFFER_GETS_high)	(SORTS_DELTA_medium, ELAPSED_TIME_medium)	0.08	0.09	0.07	0.93	9.81	0.07	12.30	0.97
(SORTS_DELTA_medium, ELAPSED_TIME_medium, BUFFER_GETS_high)	(CPU_TIME_high)	0.07	0.10	0.07	1.00	9.56	0.07	428.99	0.97
(SORTS_DELTA_medium, BUFFER_GETS_high)	(CPU_TIME_high, ELAPSED_TIME_medium)	0.07	0.10	0.07	0.99	9.54	0.07	72.32	0.97
(ELAPSED_TIME_medium, BUFFER_GETS_high)	(CPU_TIME_high)	0.08	0.10	0.08	0.99	9.52	0.07	154.04	0.97
(SORTS_DELTA_medium, BUFFER_GETS_high)	(CPU_TIME_high)	0.07	0.10	0.07	0.99	9.46	0.07	72.25	0.97
(CPU_TIME_high)	(SORTS_DELTA_medium, ELAPSED_TIME_medium)	0.10	0.09	0.09	0.88	9.32	0.08	7.55	1.00
(SORTS_DELTA_medium, ELAPSED_TIME_medium)	(CPU_TIME_high)	0.09	0.10	0.09	0.97	9.32	0.08	32.61	0.99
(BUFFER_GETS_high)	(CPU_TIME_high, SORTS_DELTA_medium)	0.09	0.09	0.07	0.84	9.16	0.06	5.73	0.98
(BUFFER_GETS_high)	(SORTS_DELTA_medium, CPU_TIME_high, ELAPSED_TIME_medium)	0.09	0.09	0.07	0.84	9.16	0.06	5.73	0.98
(BUFFER_GETS_high)	(SORTS_DELTA_medium, ELAPSED_TIME_medium)	0.09	0.09	0.07	0.84	8.93	0.06	5.78	0.97
(BUFFER_GETS_high)	(CPU_TIME_high, ELAPSED_TIME_medium)	0.09	0.10	0.08	0.90	8.73	0.07	9.26	0.97
(BUFFER_GETS_high)	(CPU_TIME_high)	0.09	0.10	0.08	0.91	8.70	0.07	9.78	0.97
(IO_WAIT_medium)	(ELAPSED_TIME_medium)	0.06	0.22	0.06	1.00	4.61	0.05	inf	0.83
(IO_WAIT_medium, CPU_TIME_medium)	(ELAPSED_TIME_medium)	0.05	0.22	0.05	1.00	4.61	0.04	inf	0.82

Tabela 11 – Algumas regras geradas pelo experimento 2, ordenadas pela medida *lift* em ordem decrescente, sendo que ant support = suporte do antecedente; cons support = suporte do consequente; supp = suporte; conf = confiança; lev = levarage; conv = convicção

Utilizando o Pyplot, foi feita a representação gráfica das regras por meio do mapa de calor das regras com melhores valores da medida *lift*. O resultado é ilustrado na Figura 10. Já na Figura 11 também é representado um mapa de calor, porém destacando apenas os melhores atributos e métricas de acordo com a medida *lift*.

Figura 10 – Mapa de calor das regras com melhores valores da medida *lift*

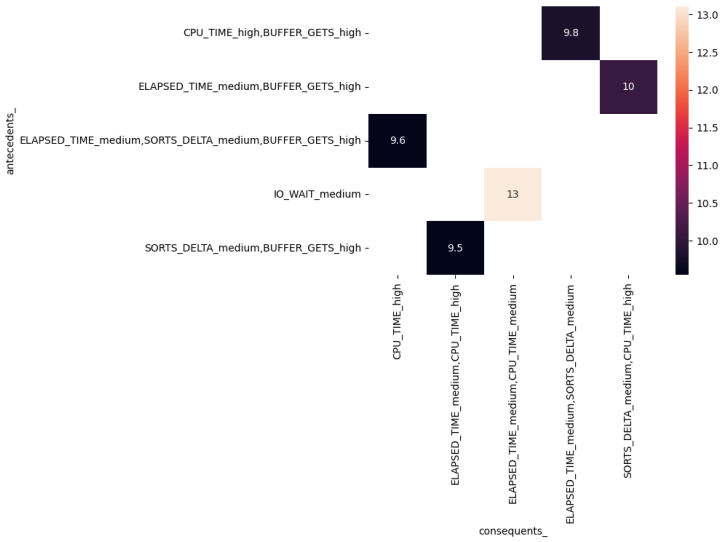

Figura 11 – Mapa de calor das melhores regras com melhores valores da medida *lift*

4.2.3 Experimento 3

O terceiro experimento foi realizado com o *dataset* 3 e com a remoção de atributos nas faixas baixas (_low). Conforme discutido na seção 3.1.3, o *dataset* 3 representa um subconjunto do *dataset* 1. Em detalhes, o *dataset* 1 contém dados referentes a um período muito grande quando comparado ao dia 26 de maio de 2023. Neste dia, houve acionamento da equipe para apoio na resolução de um problema, pois uma das aplicações apresentava lentidão generalizada. Durante conferência do problema com o cliente, foram identificadas algumas consultas com mudança de plano de execução. Essas consultas realizavam mais leituras de *buffer* (memória) por execução, conforme identificado por "Avg LIO Per Exec" na Figura 12.

Summary Execution Statistics Over Time

Snapshot Time	Execs	Avg LIO Per Exec	Avg PIO Per Exec	Avg CPU (secs) Per Exec	Avg Elapsed (secs) Per Exec
24-MAY 14:45	1	1,112.00	5.00	0.54	0.55
24-MAY 17:00	1	1,210.00	0.00	0.36	0.36
26-MAY 10:45	1	59,968,083.00	0.00	119.09	119.76
26-MAY 15:15	1	69,900,824.00	0.00	115.78	115.82
26-MAY 16:00	1	45,134,606.00	0.00	71.11	71.13
26-MAY 16:00	1	69,905,751.00	0.00	107.77	107.79
26-MAY 16:15	1	69,907,924.00	0.00	106.70	106.91
26-MAY 16:15	1	69,907,972.00	0.00	148.00	149.52
avg	1	48,090,935.25	0.63	83.67	83.98
sum	8				

Per-Plan Execution Statistics Over Time

Plan Hash Value	Snapshot Time	Execs	Avg LIO Per Exec	Avg PIO Per Exec	Avg CPU (secs) Per Exec	Avg Elapsed (secs) Per Exec
862931711	26-MAY 10:45	1	59,968,083.00	0.00	119.09	119.76
	26-MAY 15:15	1	69,900,824.00	0.00	115.78	115.82
	26-MAY 16:00	1	69,905,751.00	0.00	107.77	107.79
	26-MAY 16:00	1	45,134,606.00	0.00	71.11	71.13
	26-MAY 16:15	1	69,907,924.00	0.00	106.70	106.91
	26-MAY 16:15	1	69,907,972.00	0.00	148.00	149.52
**********						
avg			64,120,860.00	0.00	111.41	111.82
sum		6				
4018483521	24-MAY 14:45	1	1,112.00	5.00	0.54	0.55
	24-MAY 17:00	1	1,210.00	0.00	0.36	0.36
**********						
avg			1,161.00	2.50	0.45	0.45
sum		2				

Elapsed: 00:00:00.00

Figura 12 – Algumas estatísticas coletadas no dia 26 de maio durante o período da tarde, as quais mostram mudança em planos de execução de consultas e seu aumento de consulta em memória quando comparada com o plano anterior

Como era conhecido que no dia 26 de maio de 2023 houve conferência para tentativa de encontrar o problema e suas soluções, o *dataset* 3 contém um subconjunto dos dados do *dataset* 1. Entretanto, ocorre uma redução para apenas algumas horas daquele dia, como mostrado na Figura 13.

	IO_WAIT_medium	IO_WAIT_high	AP_WAIT_medium	AP_WAIT_high	CONCURRENCY_WAIT_medium	CONCURRENCY_WAIT_high	EXECUTIONS_medium	EXECUTIONS_high	●●●
8	0	0	0	0	0	0	0	0	
17	0	0	0	0	0	0	1	0	
20	0	0	0	0	0	0	0	0	
38	0	0	0	0	0	0	0	0	
44	0	0	0	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	
4293	0	0	0	0	0	0	0	0	
4295	0	0	0	0	0	0	1	0	
4296	0	0	0	0	0	0	0	0	
4306	0	0	0	0	0	0	1	0	
4309	0	0	0	0	0	0	0	0	

543 rows × 30 columns

Figura 13 – *Dataset* do dia 26 de maio, reduzido e discretizado

Sumário do *dataset* para este experimento:

- Período SNAP_TIME *Dataset* 3: 26/05/23 9:00 às 26/05/23 18:00
- Tamanho original: 4363 linhas x 23 colunas
- Remoção de atributos calculados a partir de outros / irrelevantes: Sim
- Remoção de atributos _low (faixa baixa): Sim
- Remoção de linhas com soma igual a zero (transações vazias): Sim
- Tamanho *dataset* após pré-processamento acima: 543 linhas × 30 colunas
- Suporte mínimo utilizado no APRIORI: 0.01 (1%)

Na Tabela 12 são detalhados os valores máximos de cada atributo, seus respectivos SQL_ID e SNAP_TIME, este último indicando o momento no qual os valores ocorreram.

Atributo do dataset (antes discretização)	**SQL_ID e SNAP_TIME**	**Valor máximo do atributo no intervalo**
EXECUTIONS	(5xv7qs43u94j9, 5/26/2023 10:15)	2052516
DISK_READS	(25by9zbr3gcpg, 5/26/2023 17:30)	71682909
BUFFER_GETS	(37apnmb6vfuar, 5/26/2023 15:15)	1090417811
ROWS_PROCESSED	(abrt03r5nsw5t, 5/26/2023 10:00)	286946572
CPU_TIME	(37apnmb6vfuar, 5/26/2023 15:15)	2363
IO_WAIT	(5jwhq4w2fy5by, 5/26/2023 15:15)	17363748984
AP_WAIT	(fas80nffgdp6q, 5/26/2023 11:45)	2067796162
CONCURRENCY_WAIT	(5q6mm4m9fmxq9, 5/26/2023 13:00)	447382397
ELAPSED_TIME	(5jwhq4w2fy5by, 5/26/2023 15:15)	31351
PHYSICAL_WRITE_REQUESTS_DELTA	(25by9zbr3gcpg, 5/26/2023 17:30)	26877
PHYSICAL_READ_REQUESTS_DELTA	(25by9zbr3gcpg, 5/26/2023 17:30)	793699
PHYSICAL_WRITE_BYTES_DELTA	(g4wbndxmkk7ga, 5/26/2023 9:45)	7911489536
PHYSICAL_READ_BYTES_DELTA	(25by9zbr3gcpg, 5/26/2023 17:30)	587000000000
DIRECT_WRITES_DELTA	(g4wbndxmkk7ga, 5/26/2023 9:45)	975998
SORTS_DELTA	(5c6hk3aw50d9m, 5/26/2023 17:30)	63937

Tabela 12 – Valor máximo do atributo no intervalo/consulta do experimento 3

Na Tabela 13 são listados alguns dos *itemsets* mais frequentes encontrados no presente experimento.

support	**itemsets**
0.252302	(EXECUTIONS_medium)
0.193370	(PHYSICAL_READ_REQUESTS_DELTA_medium)
0.163904	(PHYSICAL_WRITE_REQUESTS_DELTA_medium)
0.160221	(CPU_TIME_medium)
0.156538	(ELAPSED_TIME_medium)
0.125230	(PHYSICAL_READ_REQUESTS_DELTA_medium, PHYSICAL_WRITE_REQUESTS_DELTA_medium)
0.125230	(SORTS_DELTA_medium)
0.103131	(BUFFER_GETS_medium)
0.103131	(DIRECT_WRITES_DELTA_medium)
0.099448	(BUFFER_GETS_medium, CPU_TIME_medium)
0.095764	(ELAPSED_TIME_medium, CPU_TIME_medium)
0.071823	(ELAPSED_TIME_medium, IO_WAIT_medium)
0.071823	(IO_WAIT_medium)
0.055249	(IO_WAIT_medium, CPU_TIME_medium)
0.055249	(ELAPSED_TIME_medium, IO_WAIT_medium, CPU_TIME_medium)
0.051565	(CONCURRENCY_WAIT_medium)
0.047882	(AP_WAIT_medium)
0.042357	(CPU_TIME_high)
0.042357	(BUFFER_GETS_medium, ELAPSED_TIME_medium, CPU_TIME_medium)
0.042357	(BUFFER_GETS_medium, ELAPSED_TIME_medium)
0.034991	(CPU_TIME_high, ELAPSED_TIME_medium)
0.031308	(CPU_TIME_high, BUFFER_GETS_high)
0.031308	(BUFFER_GETS_high)
0.031308	(CONCURRENCY_WAIT_medium, EXECUTIONS_medium)
0.027624	(ELAPSED_TIME_medium, BUFFER_GETS_high)
0.027624	(CPU_TIME_high, ELAPSED_TIME_medium, BUFFER_GETS_high)
0.025783	(EXECUTIONS_high)
0.020258	(CONCURRENCY_WAIT_medium, EXECUTIONS_high)
0.018416	(AP_WAIT_medium, EXECUTIONS_medium)
0.018416	(AP_WAIT_high)
0.014733	(SORTS_DELTA_medium, EXECUTIONS_medium)
0.014733	(SORTS_DELTA_high)
0.011050	(PHYSICAL_READ_REQUESTS_DELTA_high)

Tabela 13 – Itemsets mais frequentes gerados no experimento 3

Adicionalmente, na Tabela 14 são descritas as regras com suas medidas, ordenadas pela medida *lift*.

antecedents	consequents	ant supp	cons supp	supp	conf	lift	lev	conv	zhangs
(BUFFER_GETS_high)	(CPU_TIME_high, ELAPSED_TIME_medium)	0.03	0.03	0.03	0.88	25.22	0.03	8.20	0.99
(BUFFER_GETS_high)	(CPU_TIME_high)	0.03	0.04	0.03	1.00	23.61	0.03	inf	0.99
(BUFFER_GETS_high, ELAPSED_TIME_medium)	(CPU_TIME_high)	0.03	0.04	0.03	1.00	23.61	0.03	inf	0.98
(IO_WAIT_medium)	(ELAPSED_TIME_medium)	0.07	0.16	0.07	1.00	6.39	0.06	inf	0.91
(IO_WAIT_medium, CPU_TIME_medium)	(ELAPSED_TIME_medium)	0.06	0.16	0.06	1.00	6.39	0.05	inf	0.89
(BUFFER_GETS_medium, ELAPSED_TIME_medium)	(CPU_TIME_medium)	0.04	0.16	0.04	1.00	6.24	0.04	inf	0.88
(BUFFER_GETS_medium)	(CPU_TIME_medium)	0.10	0.16	0.10	0.96	6.02	0.08	23.51	0.93
(BUFFER_GETS_high)	(ELAPSED_TIME_medium)	0.03	0.16	0.03	0.88	5.64	0.02	7.17	0.85
(BUFFER_GETS_high, CPU_TIME_high)	(ELAPSED_TIME_medium)	0.03	0.16	0.03	0.88	5.64	0.02	7.17	0.85
(CPU_TIME_high)	(ELAPSED_TIME_medium)	0.04	0.16	0.03	0.83	5.28	0.03	4.85	0.85

Tabela 14 – Regras geradas pelo experimento 3, ordenadas pela medida *lift* em parâmetros decrescente

Utilizando o Pyplot, foi feita a representação gráfica das regras por meio do mapa de calor das regras com melhores valores da medida *lift*. O resultado é ilustrado na Figura 14.

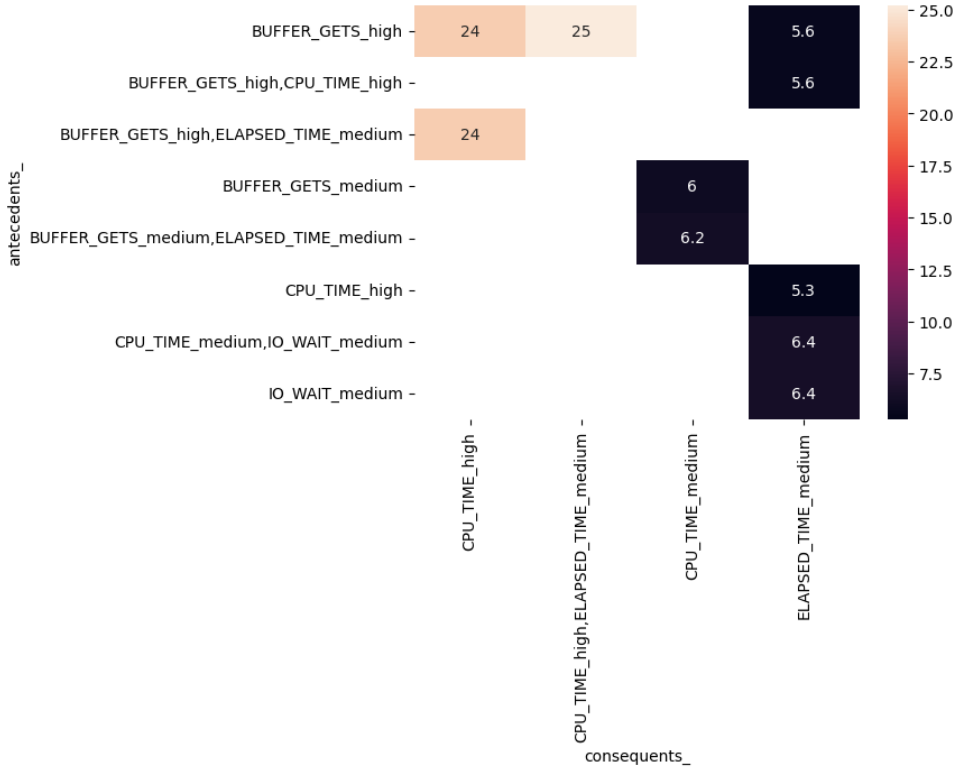

Figura 14 – Mapa de calor das regras com melhores valores da medida *lift* para o experimento 3

4.2.4 Experimento 4

O quarto experimento foi realizado com o *dataset* 2 e com a remoção de atributos nas faixas baixas (_low). Conforme discutido na seção 3.1.3, o *dataset* 2 compreende outra situação na qual houve lentidão em aplicação. A seguir é descrito como foi realizado o experimento.

Sumário do *dataset* para este experimento:

- Período SNAP_TIME *Dataset* 3: 30/07/23 0:15 às 06/08/23 15:00
- Tamanho original: 122429 linhas x 23 colunas
- Remoção de atributos calculados a partir de outros / irrelevantes: Sim
- Remoção de atributos _low (faixa baixa): Sim
- Remoção de linhas com soma igual a zero (transações vazias): Sim
- Tamanho *dataset* após pré-processamento acima: 8119 linhas × 30 colunas
- Suporte mínimo utilizado no APRIORI: 0.01 (1%)

Na Tabela 15 são detalhados os valores máximos de cada atributo, seus respectivos SQL_ID e SNAP_TIME, este último indicando o momento no qual os valores ocorreram.

Atributo do dataset (antes discretização	**SQL_ID e SNAP_TIME**	**Valor máximo do atributo no intervalo**
EXECUTIONS	(8/5/2023 20:00, 7uxkdpxfj8shv)	6676965
DISK_READS	(8/2/2023 19:00, gm2brdhxrqsw6)	127828305
BUFFER_GETS	(8/5/2023 0:15, 3xjw1ncw5vh27)	1100304611
ROWS_PROCESSED	(8/4/2023 23:30, 3k4hsqxwsunn7)	719484992
CPU_TIME	(8/2/2023 19:00, gm2brdhxrqsw6)	31113
IO_WAIT	(8/2/2023 19:00, gm2brdhxrqsw6)	216431534173
AP_WAIT	(8/2/2023 18:15, 30vsxd7ch608a)	151865407497
CONCURRENCY_WAIT	(8/6/2023 3:30, axw75jd3htabr)	49557955040
ELAPSED_TIME	(8/2/2023 19:00, gm2brdhxrqsw6)	292961
PHYSICAL_WRITE_REQUESTS_DELTA	(8/4/2023 23:00, 9cj72br7s0wgq)	89696
PHYSICAL_READ_REQUESTS_DELTA	(8/2/2023 19:00, gm2brdhxrqsw6)	127810200
PHYSICAL_WRITE_BYTES_DELTA	(8/4/2023 23:00, 9cj72br7s0wgq)	23500000000
PHYSICAL_READ_BYTES_DELTA	(8/2/2023 19:00, gm2brdhxrqsw6)	1050000000000
DIRECT_WRITES_DELTA	(8/4/2023 23:30, fqnsy1uz9y4fq)	1260831
SORTS_DELTA	(8/3/2023 6:45, ct7w65zr2n5vx)	65528

Tabela 15 – Valor máximo do atributo no intervalo/consulta para o experimento 4

Na Tabela 16 são listados alguns dos *itemsets* mais frequentes encontrados no presente experimento.

support	**itemsets**
0.475921	(EXECUTIONS_medium)
0.192881	(SORTS_DELTA_medium)
0.149526	(PHYSICAL_WRITE_REQUESTS_DELTA_medium)
0.140781	(SORTS_DELTA_high)
0.064540	(DIRECT_WRITES_DELTA_medium)
0.042986	(PHYSICAL_WRITE_REQUESTS_DELTA_high)
0.042862	(DIRECT_WRITES_DELTA_medium, PHYSICAL_WRITE_REQUESTS_DELTA_high)
0.038182	(DISK_READS_medium)
0.036211	(SORTS_DELTA_medium, EXECUTIONS_medium)
0.035596	(EXECUTIONS_high)
0.033132	(SORTS_DELTA_high, EXECUTIONS_medium)
0.030546	(BUFFER_GETS_medium)
0.030299	(PHYSICAL_READ_BYTES_DELTA_medium)
0.029437	(PHYSICAL_READ_BYTES_DELTA_medium, DISK_READS_medium)
0.028944	(CPU_TIME_medium)
0.026235	(DISK_READS_medium, CPU_TIME_medium)
0.022663	(BUFFER_GETS_medium, DISK_READS_medium)
0.022417	(SORTS_DELTA_high, DIRECT_WRITES_DELTA_medium)
0.021924	(BUFFER_GETS_medium, CPU_TIME_medium)
0.021062	(BUFFER_GETS_medium, DISK_READS_medium, CPU_TIME_medium)

Tabela 16 – Vinte Itemsets mais frequentes gerados para o experimento 4

Adicionalmente, na Tabela 17 são descritas as regras com suas medidas, ordenadas pela medida *lift*.

antecedents	consequents	ant supp	cons supp	supp	conf	lift	lev	conv	zhangs
(CPU_TIME_high)	(IO_WAIT_medium, DISK_READS_high)	0.01	0.01	0.01	0.96	76.28	0.01	23.70	1.00
(IO_WAIT_medium, DISK_READS_high)	(CPU_TIME_high)	0.01	0.01	0.01	0.90	76.28	0.01	10.08	1.00
(CPU_TIME_high)	(IO_WAIT_medium)	0.01	0.01	0.01	0.97	69.60	0.01	31.55	1.00
(IO_WAIT_medium)	(CPU_TIME_high)	0.01	0.01	0.01	0.82	69.60	0.01	5.58	1.00
(CPU_TIME_high, DISK_READS_high)	(IO_WAIT_medium)	0.01	0.01	0.01	0.97	69.58	0.01	31.23	1.00
(IO_WAIT_medium)	(CPU_TIME_high, DISK_READS_high)	0.01	0.01	0.01	0.81	69.58	0.01	5.32	1.00
(BUFFER_GETS_high, DISK_READS_high)	(IO_WAIT_medium)	0.01	0.01	0.01	0.96	68.69	0.01	22.43	1.00
(CPU_TIME_high)	(DISK_READS_high)	0.01	0.01	0.01	0.99	68.09	0.01	94.60	1.00
(DISK_READS_high)	(CPU_TIME_high)	0.01	0.01	0.01	0.81	68.09	0.01	5.07	1.00
(CPU_TIME_high, IO_WAIT_medium)	(DISK_READS_high)	0.01	0.01	0.01	0.99	68.07	0.01	91.65	1.00
(BUFFER_GETS_high, IO_WAIT_medium)	(DISK_READS_high)	0.01	0.01	0.01	0.99	68.02	0.01	86.72	1.00
(IO_WAIT_medium)	(DISK_READS_high)	0.01	0.01	0.01	0.90	62.11	0.01	10.12	1.00
(DISK_READS_high)	(IO_WAIT_medium)	0.01	0.01	0.01	0.86	62.11	0.01	7.27	1.00
(PHYSICAL_READ_REQUESTS_DELTA_medium, DISK_READS_medium)	(PHYSICAL_READ_BYTES_DELTA_medium, BUFFER_GETS_medium, CPU_TIME_medium)	0.02	0.02	0.01	0.82	50.25	0.01	5.37	1.00
(PHYSICAL_READ_BYTES_DELTA_medium, BUFFER_GETS_medium, CPU_TIME_medium)	(PHYSICAL_READ_REQUESTS_DELTA_medium, DISK_READS_medium)	0.02	0.02	0.01	0.88	50.25	0.01	8.11	1.00
(PHYSICAL_READ_REQUESTS_DELTA_medium, BUFFER_GETS_medium)	(PHYSICAL_READ_BYTES_DELTA_medium, DISK_READS_medium, CPU_TIME_medium)	0.01	0.02	0.01	0.97	50.09	0.01	38.89	0.99
(PHYSICAL_READ_REQUESTS_DELTA_medium, DISK_READS_medium)	(PHYSICAL_READ_BYTES_DELTA_medium, CPU_TIME_medium)	0.01	0.02	0.01	1.00	49.81	0.01	inf	0.99
(PHYSICAL_READ_BYTES_DELTA_medium, CPU_TIME_medium)	(PHYSICAL_READ_REQUESTS_DELTA_medium, DISK_READS_medium)	0.02	0.02	0.02	0.87	49.81	0.02	7.63	1.00
(PHYSICAL_READ_REQUESTS_DELTA_medium, DISK_READS_medium, BUFFER_GETS_medium)	(PHYSICAL_READ_BYTES_DELTA_medium, CPU_TIME_medium)	0.02	0.02	0.02	1.00	49.81	0.02	inf	1.00
(PHYSICAL_READ_REQUESTS_DELTA_medium, BUFFER_GETS_medium)	(PHYSICAL_READ_BYTES_DELTA_medium, CPU_TIME_medium)	0.01	0.02	0.01	0.98	48.97	0.01	58.31	0.99

Tabela 17 – Vinte melhores regras geradas no experimento 4, ordenadas pela medida *lift* em ordem decrescente, sendo que ant support = suporte do antecedente; cons support = suporte do consequente; supp = suporte; conf = confiança; lev = levarage; conv = convicção

Utilizando o Pyplot, foi feita a representação gráfica das regras por meio do mapa de calor das regras com melhores valores da medida *lift*. O resultado é ilustrado na Figura 15.

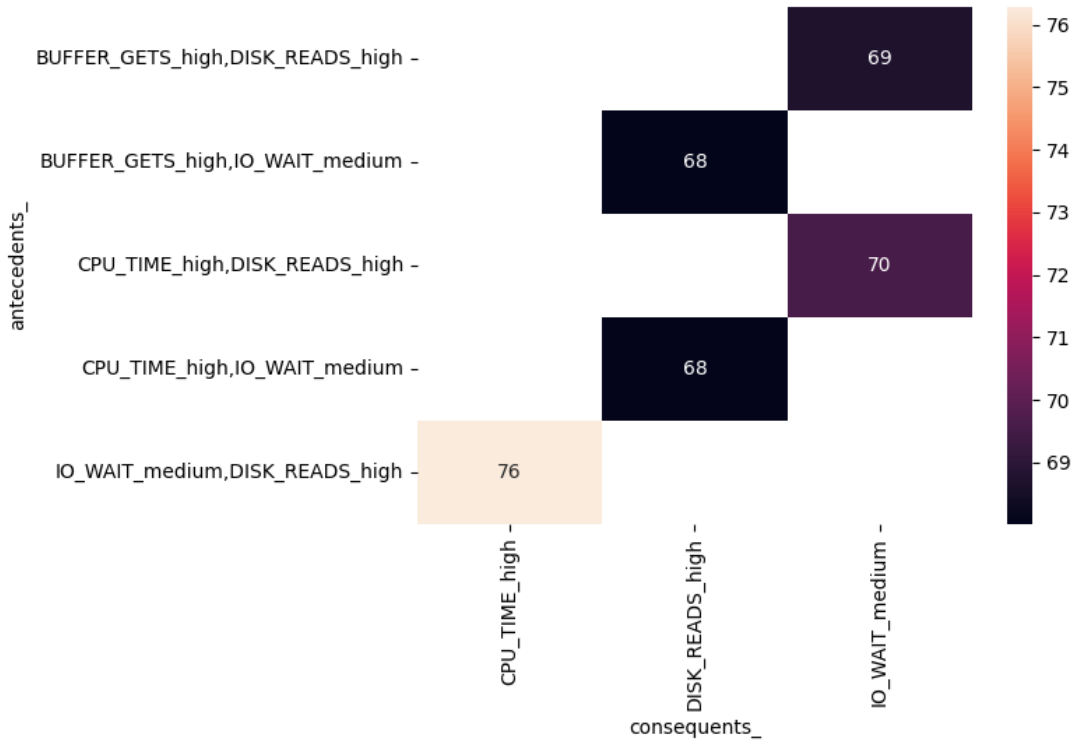

Figura 15 – Mapa de calor das melhores regras com melhores valores da medida *lift* para o experimento 4

4.2.5 Experimento 5

O quinto e último experimento foi realizado com o *dataset* 2 e com a remoção de atributos nas faixas baixas (_low). Conforme discutido na seção 3.1.3, o *dataset* 2 contém um intervalo de tempo que melhor representa o momento no qual foi realizada a conferência junto à equipe de suporte a banco de dados para entender o que poderia ter acontecido. Este intervalo, começa às 14 horas e termina as 18:15 do mesmo dia. Durante este período, foi identificado que a aplicação estava executando um *script* para limpeza de dados em tabelas. Assim que esse *script* foi interrompido, o problema foi resolvido.

O sumário do *dataset* para este experimento é descrito a seguir:

- Período SNAP_TIME *Dataset* 4: 02/08/2023 14:00 às 02/08/23 18:15
- Tamanho original: 3025 linhas x 23 colunas
- Remoção de atributos calculados a partir de outros / irrelevantes: Sim
- Remoção de atributos _low (faixa baixa): Sim
- Remoção de linhas com soma igual a zero (transações vazias): Sim
- Tamanho *dataset* após pré-processamento acima: 304 linhas × 30 colunas
- Suporte mínimo utilizado no APRIORI: 0.01 (1%)

Na Tabela 18 são detalhados os valores máximos de cada atributo e seus respectivos valores de SQL_ID e SNAP_TIME, este último indicando o momento no qual os valores ocorreram.

Atributo do dataset (antes discretização)	**SQL_ID e SNAP_TIME**	**Valor máximo do atributo no intervalo**
EXECUTIONS	(8/2/2023 16:30, 5xv7qs43u94j9)	5392865
DISK_READS	(8/2/2023 16:15, gm2brdhxrqsw6)	116566137
BUFFER_GETS	(8/2/2023 16:15, gm2brdhxrqsw6)	806964623
ROWS_PROCESSED	(8/2/2023 16:30, 5b4k08d6sgqyb)	34511056
CPU_TIME	(8/2/2023 16:15, gm2brdhxrqsw6)	29977
IO_WAIT	(8/2/2023 16:15, gm2brdhxrqsw6)	125027302248
AP_WAIT	(8/2/2023 15:15, 30vsxd7ch608a)	150475738818
CONCURRENCY_WAIT	(8/2/2023 15:45, axw75jd3htabr)	17996987628
ELAPSED_TIME	(8/2/2023 16:15, gm2brdhxrqsw6)	224817
PHYSICAL_WRITE_REQUESTS_DELTA	(8/2/2023 16:15, cscj3rtsqnqkq)	67448
PHYSICAL_READ_REQUESTS_DELTA	(8/2/2023 16:15, gm2brdhxrqsw6)	116551317
PHYSICAL_WRITE_BYTES_DELTA	(8/2/2023 16:15, cscj3rtsqnqkq)	552599552
PHYSICAL_READ_BYTES_DELTA	(8/2/2023 16:15, gm2brdhxrqsw6)	955000000000
DIRECT_WRITES_DELTA	(8/2/2023 16:15, cscj3rtsqnqkq)	67456
SORTS_DELTA	(8/2/2023 16:30, ct7w65zr2n5vx)	63259

Tabela 18 – Valor máximo do atributo no intervalo/consulta para o experimento 5

Na Tabela 19 são listados alguns dos *itemsets* mais frequentes encontrados no presente experimento. Foram gerados 294 *itemsets* frequentes, sendo que na tabela são listados apenas os que tiveram suporte maior ou igual a 0.03.

support	**itemsets**
0.50657895	(EXECUTIONS_medium)
0.17763158	(SORTS_DELTA_medium)
0.14802632	(SORTS_DELTA_high)
0.06907895	(ELAPSED_TIME_medium)
0.05592105	(PHYSICAL_WRITE_BYTES_DELTA_medium)
0.05592105	(EXECUTIONS_high)
0.05592105	(DIRECT_WRITES_DELTA_medium, PHYSICAL_WRITE_BYTES_DELTA_medium)
0.05592105	(DIRECT_WRITES_DELTA_medium)
0.04934211	(DIRECT_WRITES_DELTA_medium, PHYSICAL_WRITE_REQUESTS_DELTA_medium, PHYSICAL_WRITE_BYTES_DELTA_medium)
0.04934211	(PHYSICAL_WRITE_REQUESTS_DELTA_medium, PHYSICAL_WRITE_BYTES_DELTA_medium)
0.04934211	(DIRECT_WRITES_DELTA_medium, PHYSICAL_WRITE_REQUESTS_DELTA_medium)
0.04934211	(PHYSICAL_WRITE_REQUESTS_DELTA_medium)
0.04276316	(SORTS_DELTA_medium, EXECUTIONS_medium)
0.04276316	(PHYSICAL_READ_BYTES_DELTA_medium)
0.04276316	(PHYSICAL_READ_BYTES_DELTA_medium, DISK_READS_medium)
0.04276316	(DISK_READS_medium)
0.03947368	(CPU_TIME_medium)
0.03618421	(PHYSICAL_READ_REQUESTS_DELTA_medium)
0.03618421	(PHYSICAL_READ_BYTES_DELTA_medium, PHYSICAL_READ_REQUESTS_DELTA_medium)
0.03618421	(PHYSICAL_READ_BYTES_DELTA_medium, DISK_READS_medium, PHYSICAL_READ_REQUESTS_DELTA_medium, CPU_TIME_medium)
0.03618421	(PHYSICAL_READ_REQUESTS_DELTA_medium, DISK_READS_medium)
0.03618421	(PHYSICAL_READ_BYTES_DELTA_medium, CPU_TIME_medium)
0.03618421	(PHYSICAL_READ_REQUESTS_DELTA_medium, CPU_TIME_medium)
0.03618421	(PHYSICAL_READ_REQUESTS_DELTA_medium, DISK_READS_medium, CPU_TIME_medium)
0.03618421	(ROWS_PROCESSED_medium)
0.03618421	(PHYSICAL_READ_BYTES_DELTA_medium, DISK_READS_medium, CPU_TIME_medium)
0.03618421	(PHYSICAL_READ_BYTES_DELTA_medium, DISK_READS_medium, PHYSICAL_READ_REQUESTS_DELTA_medium)
0.03618421	(BUFFER_GETS_medium)
0.03618421	(PHYSICAL_READ_BYTES_DELTA_medium, PHYSICAL_READ_REQUESTS_DELTA_medium, CPU_TIME_medium)
0.03618421	(DISK_READS_medium, CPU_TIME_medium)
0.03289474	(BUFFER_GETS_medium, CPU_TIME_medium)
0.03289474	(AP_WAIT_medium)
0.03289474	(SORTS_DELTA_high, EXECUTIONS_medium)

Tabela 19 – Itemsets mais frequentes gerados para o experimento 5, com suporte igual ou superior a 0.03

Adicionalmente, o presente experimento gerou mais de 400 regras com todas as medidas avaliadas idênticas, as quais são descritas a seguir.

- Suporte Antecedente: 0.013157894736842105
- Suporte Consequente: 0.013157894736842105
- Suporte da Regra: 0.013157894736842105
- Confiança: 1
- Lift: 76
- Leverage: 0.012984764542936287
- Convicção: Infinito
- Zhangs: 1

No total, foram produzidas quase 3000 regras de associação entre *itemsets*. Na Tabela 20 constam algumas destas regras. Assim como nos experimentos anteriores, o resultado foi ordenado pela medida *lift*.

antecedents	**consequents**
(BUFFER_GETS_high)	(CPU_TIME_high)
(CPU_TIME_high)	(BUFFER_GETS_high)
(BUFFER_GETS_high, IO_WAIT_high)	(CPU_TIME_high)
(IO_WAIT_high, CPU_TIME_high)	(BUFFER_GETS_high)
(BUFFER_GETS_high)	(IO_WAIT_high, CPU_TIME_high)
(CPU_TIME_high)	(BUFFER_GETS_high, IO_WAIT_high)
(BUFFER_GETS_high, DISK_READS_high)	(CPU_TIME_high)
(CPU_TIME_high, DISK_READS_high)	(BUFFER_GETS_high)
(BUFFER_GETS_high)	(CPU_TIME_high, DISK_READS_high)
(CPU_TIME_high)	(BUFFER_GETS_high, DISK_READS_high)
(BUFFER_GETS_high, ELAPSED_TIME_high)	(CPU_TIME_high)
(CPU_TIME_high, ELAPSED_TIME_high)	(BUFFER_GETS_high)
(BUFFER_GETS_high)	(CPU_TIME_high, ELAPSED_TIME_high)
(CPU_TIME_high)	(BUFFER_GETS_high, ELAPSED_TIME_high)
(BUFFER_GETS_high, PHYSICAL_READ_REQUESTS_DELTA_high)	(CPU_TIME_high)
(CPU_TIME_high, PHYSICAL_READ_REQUESTS_DELTA_high)	(BUFFER_GETS_high)
(BUFFER_GETS_high)	(CPU_TIME_high, PHYSICAL_READ_REQUESTS_DELTA_high)
(CPU_TIME_high)	(BUFFER_GETS_high, PHYSICAL_READ_REQUESTS_DELTA_high)
(BUFFER_GETS_high, PHYSICAL_READ_BYTES_DELTA_high)	(CPU_TIME_high)
(CPU_TIME_high, PHYSICAL_READ_BYTES_DELTA_high)	(BUFFER_GETS_high)
(BUFFER_GETS_high)	(CPU_TIME_high, PHYSICAL_READ_BYTES_DELTA_high)
(CPU_TIME_high)	(BUFFER_GETS_high, PHYSICAL_READ_BYTES_DELTA_high)

Tabela 20 – Regras geradas para o experimento 5, ordenadas pela medida *lift* em ordem decrescente (embora os valores das medidas estudadas sejam idênticos)

Utilizando o Pyplot, foi feita a representação gráfica das regras por meio do mapa de calor das regras com melhores valores da medida *lift*. O resultado é ilustrado na Figura 16. Como grande parte das regras tem a mesma medida *lift*, a ordem das melhores regras exibidas foi a mesma utilizada nos experimentos anteriores:

```
df_toplift[df_toplift['lhs_items']>1].sort_values('lift', ascending=False).head()
```

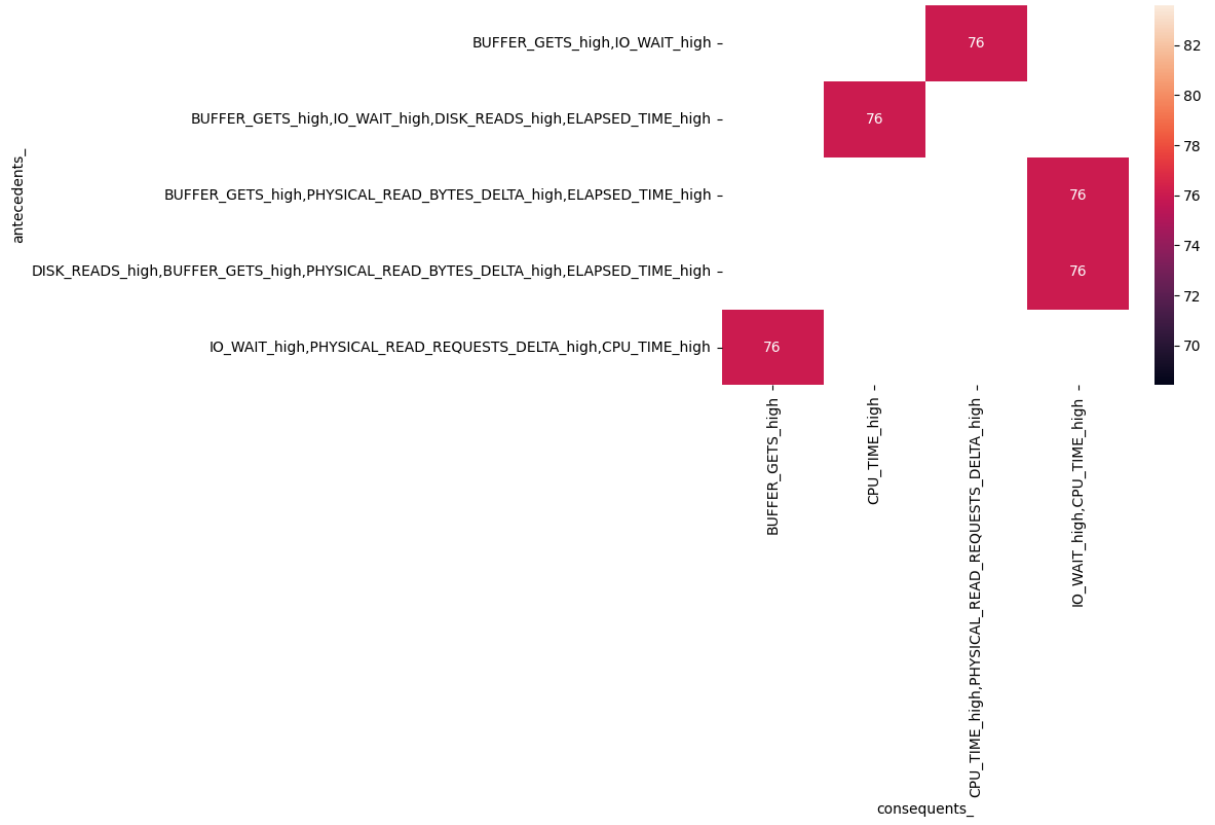

Figura 16 – Mapa de calor das melhores regras com melhores valores da medida *lift* para o experimento 5

5 AVALIAÇÃO EXPERIMENTAL

Neste capítulo são avaliados os cinco experimentos descritos no Capítulo 4, destacando informações comuns e avaliando as mesmas em relação à situação nas quais os *datasets* foram coletados. Antes de iniciar a discussão dos experimentos, é importante lembrar que o algoritmo utilizado, APRIORI, necessitou de uma fase importante de pré-processamento denominada discretização, na qual cada atributo foi separado por faixa de valores (baixo, médio, alto) utilizando o método KBinsDiscretizer. Foi realizada a comparação da estratégia de discretização e optou-se pela estratégia KMeans pelo fato desta discretizar mais dados nas faixas médias e baixas. Na seção 5.1 é feita uma análise da configuração experimental de cada um dos cinco experimentos realizados. Na seção 5.2 são discutidos e analisados os resultados obtidos nos experimentos 2 a 5.

5.1 Configuração dos Experimentos

Os experimentos 1 a 5 são analisados nas seções 5.1.1 a 5.1.5, respectivamente.

5.1.1 Experimento 1

Não foi possível realizar o experimento 1 devido ao intervalo de tempo grande e, consequentemente, ao tamanho do *dataset* gerado, que continha todas as faixas de valores dos atributos (faixas baixa, média e alta). Isso acontece porque mais de 99% das transações (ou linhas, que representam parâmetros de consumo de uma consulta em um determinado intervalo de tempo) estão na faixa baixa.

Adicionalmente, a não execução do experimento 1 também foi devido aos limites de memória RAM do ambiente Google COLAB (12GB) no qual os códigos foram executados e do equipamento pessoal do autor deste trabalho de conclusão de curso. Desde que o trabalho tem como objetivo investigar picos de médio a alto consumo de recursos nestas transações, optou-se pela redução do conjunto de dados nos demais experimentos.

5.1.2 Experimento 2

O experimento 2 foi realizado utilizando o *dataset* 1, o qual contém dados referentes a vários dias de transações. Adicionalmente, este *dataset* inclui um dia do intervalo no qual houve o problema da coleta de estatísticas automáticas no SGBD. Esse problema resultou em um impacto na aplicação e na abertura de conferência para investigação.

Todos os experimentos a partir do experimento 2 mostraram como atributo mais frequente nas transações o atributo/item EXECUTIONS. No caso do experimento 2, esse

atributo obteve o suporte de 0.34. Apesar de ser o atributo mais frequente em todos os *datasets*, ele não aparece muito nas regras geradas.

Na tentativa de melhorar a interpretação das regras de associação obtidas, buscou-se como alternativa uma representação em formato de grafos para tais regras. Na Figura 17 é ilustrada uma representação utilizando a medida *lift* para representar a espessura das ligações e a quantidade de regras. Para tanto, primeiramente foram filtradas as regras que possuem apenas um único antecedente e consequente. Depois, foi utilizada a ferramenta Cosmograph disponível no sítio <https://cosmograph.app/run/> para gerar o gráfico ilustrado na figura. Nela, a força da regra, no caso a medida *lift*, é representada pela espessura das ligações entre os nós. O tamanho dos nós representa a quantidade de regras que possuem aquele atributo como antecedente.

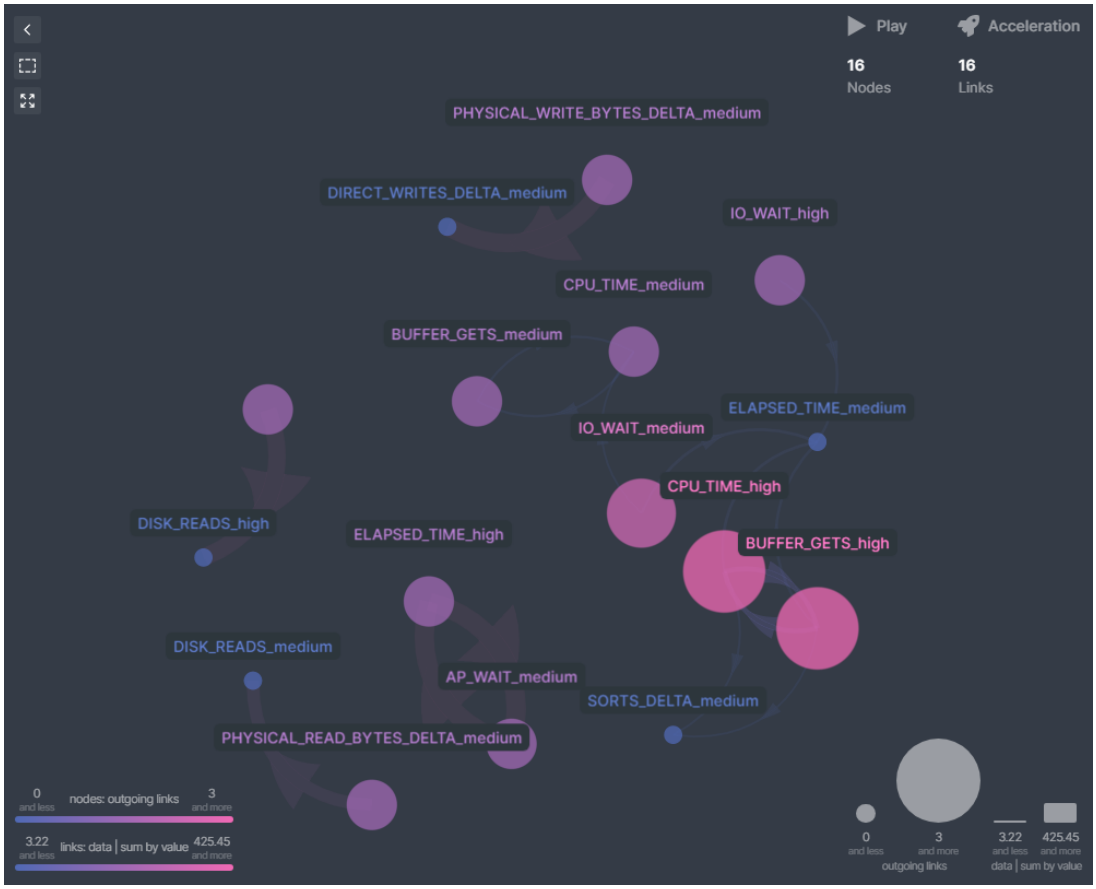

Figura 17 – Grafo utilizando a espessura das ligações para representar a medida *lift* e tamanho do nó representando quantidade de regras para o experimento 2

Na Figura 17, observa-se pela espessura larga da aresta que, durante todos os dias, a regra com maior força infere que um tempo transcorrido (ELAPSED_TIME) alto pode ter sido causado na maior parte das vezes por falta de atividade da própria aplicação (AP_WAIT). Ainda na mesma figura, nota-se que foram geradas mais regras que contêm como antecedente alta os atributos CPU (CPU_TIME), alto número de dados sendo

consumido do *buffer* em memória do SGBD (BUFFER_GETS) e médio tempo de espera em operações de entrada e saída (IO_WAIT).

Pode ser observado que, ao baixar o suporte mínimo para 0.001 e a confiança para 0.6, novas regras foram geradas com medida *lift* bem grande quando comparado ao experimento inicial discutido no Capítulo 4, realizado com outra parametrização. Estes parâmetros foram reduzidos como parte do experimento para verificar se, com mais *itemsets* frequentes, melhores regras poderiam ser geradas. Neste caso, foram geradas regras com *lift* maior ao realizar esta mudança de parametrização. Os valores de *lift* podem ser observados na Figura 18.

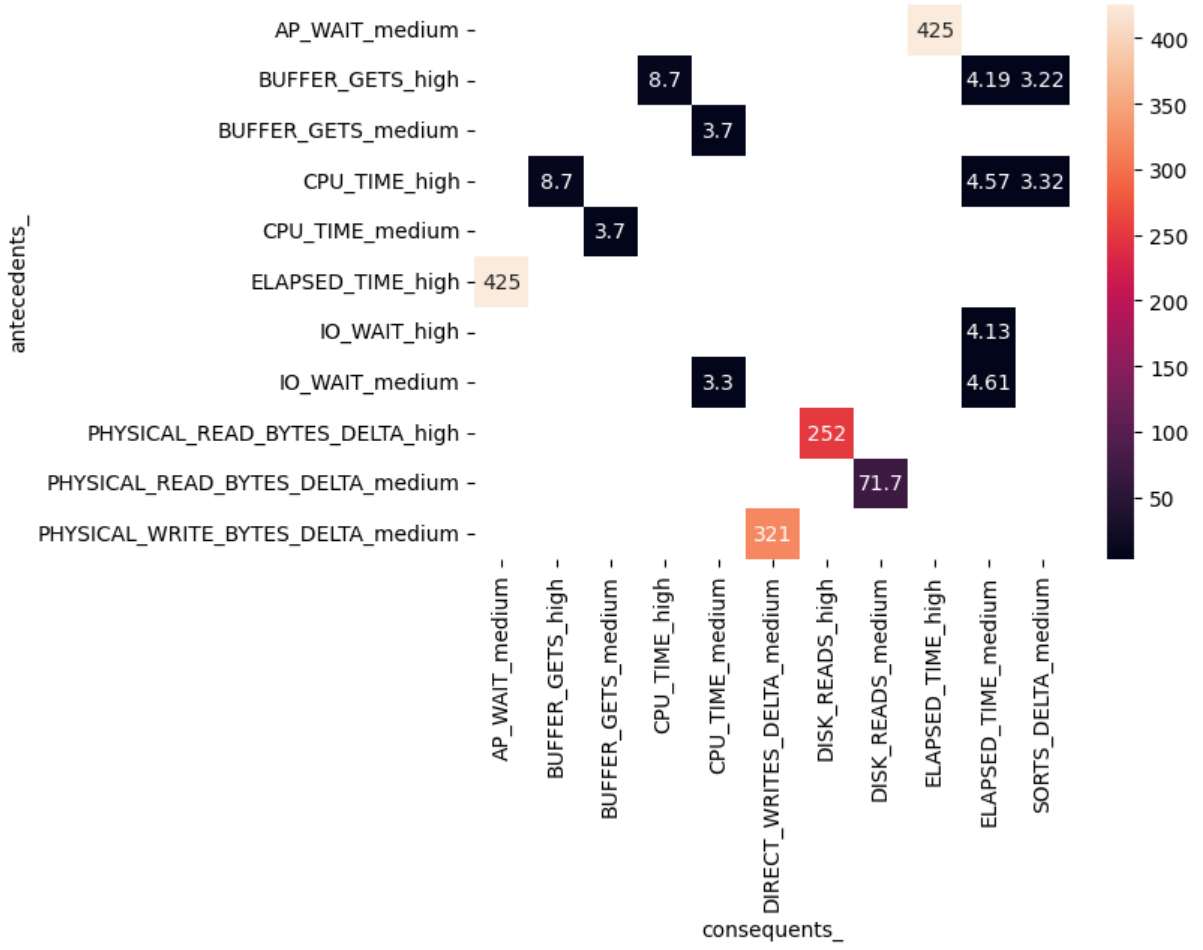

Figura 18 – Mapa de calor com suporte mínimo 0.001 e confiança 0.6 gerou novas regras fortes para o experimento 2

5.1.3 Experimento 3

Neste experimento, com o dataset reduzido para armazenar dados referentes ao período de horas no qual o problema foi reportado, percebeu-se que o alto consumo de CPU (CPU_TIME) aconteceu como consequência da alta quantidade de dados sendo consumido em memória do SGBD (BUFFER_GETS). Também foi observado que o tempo de espera médio em operações de entrada e saída (IO_WAIT) levou a um tempo médio transcorrido no total, como mostrado na Figura 14.

Adicionalmente, também foi realizada diminuição dos valores de suporte e confiança. Ao diminuir um pouco mais o suporte para 0.003 e a confiança para 0.6, regras com medida *lift* maior apareceram, e isso pode ser observado na Figura 19. Essas regras indicam também um alto número de escritas em disco sem intervenção pelo sistema operacional, DIRECT_WRITES.

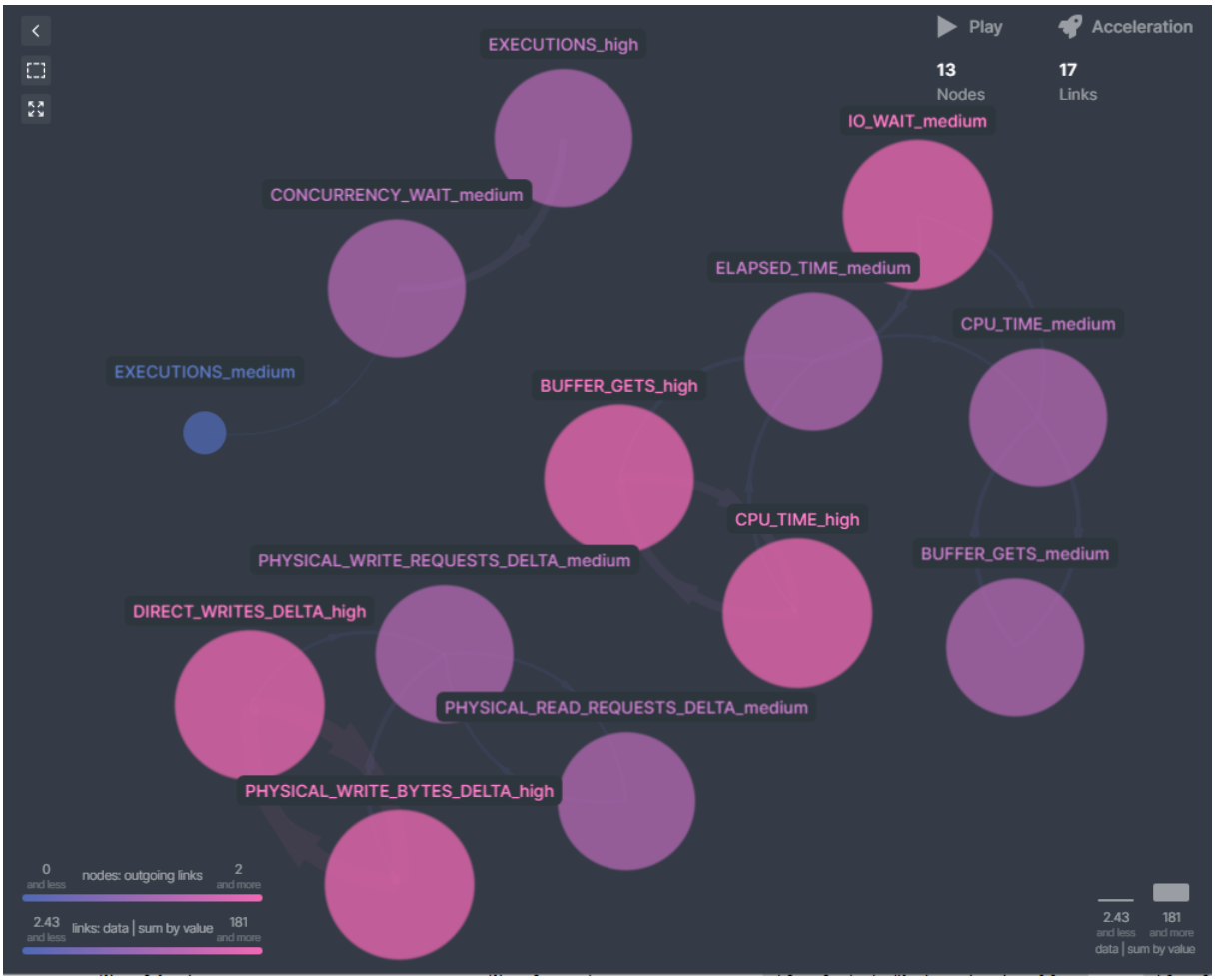

Figura 19 – Grafo utilizando a espessura das ligações para representar a medida *lift* e tamanho do nó representando quantidade de regras para o experimento 3, com suporte 0.003 e confiança 0.6

O valor desta regra forte relacionada à escrita em disco pode ser observada no mapa de calor ilustrado na Figura 20.

Figura 20 – Mapa de calor do experimento 3 com suporte mínimo 0.003 e confiança 0.6

5.1.4 Experimento 4

Este foi o primeiro experimento utilizando o *dataset* 2, o qual contém dados referentes a todos os dias e considera o cenário de um outro problema enfrentado. Primeiramente, foi investigada a geração de regras com parâmetros menores: suporte mínimo 0.001 e confiança 0.6. Como resultado, foram obtidas muitas regras óbvias, como PHYSICAL_READ_REQUESTS_DELTA_high levando a PHYSICAL_READ_BYTES_DELTA_high, com *lift* de 99. Também foram geradas regras que talvez não façam tanto sentido quando analisado o domínio do *dataset*. Por exemplo, ELAPSED_TIME_high levando a PHYSICAL_READ_REQUESTS_DELTA_high. Em geral, pensa-se em tempo transcorrido como sendo uma consequência, mas é difícil pensar neste atributo como sendo antecedente a algo.

Como resultado, foi investigada a geração de regras com outros valores de parâmetros. Foram gerados os *itemsets* frequentes com suporte 0.01 e as regras com confiança de 0.7. Os resultados obtidos podem ser observados na Figura 21, na forma de grafos onde podemos visualizar a força da medida *lift* pela espessura das arestas e a quantidade de regras contendo aquele item pelo tamanho de seu nó.

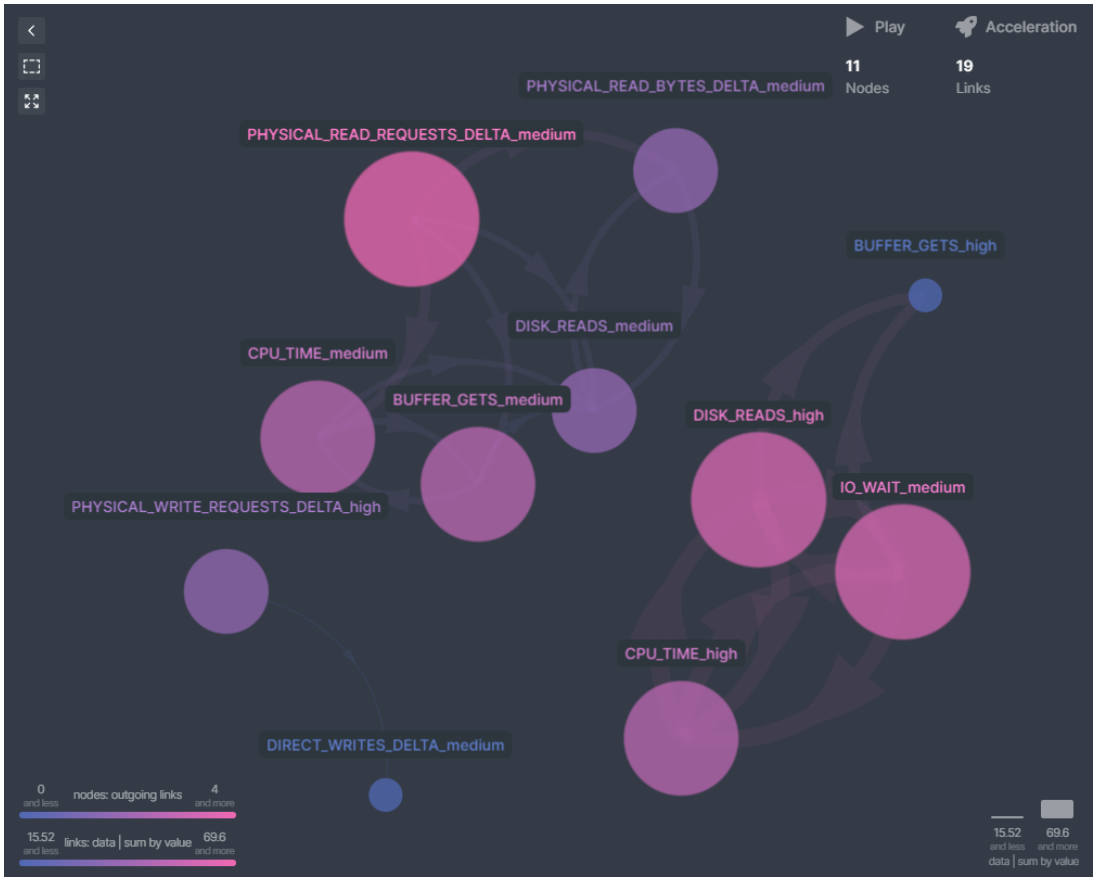

Figura 21 – Grafo utilizando a espessura das ligações para representar a medida *lift* e tamanho do nó representando quantidade de regras para o experimento 4, com suporte 0.01 e confiança 0.7

De maneira alternativa à forma em grafos, a Figura 22 apresenta o mapa de calor para os valores de parâmetros utilizados no experimento, que foram de suporte 0.01 e regras geradas com confiança de 0.7.

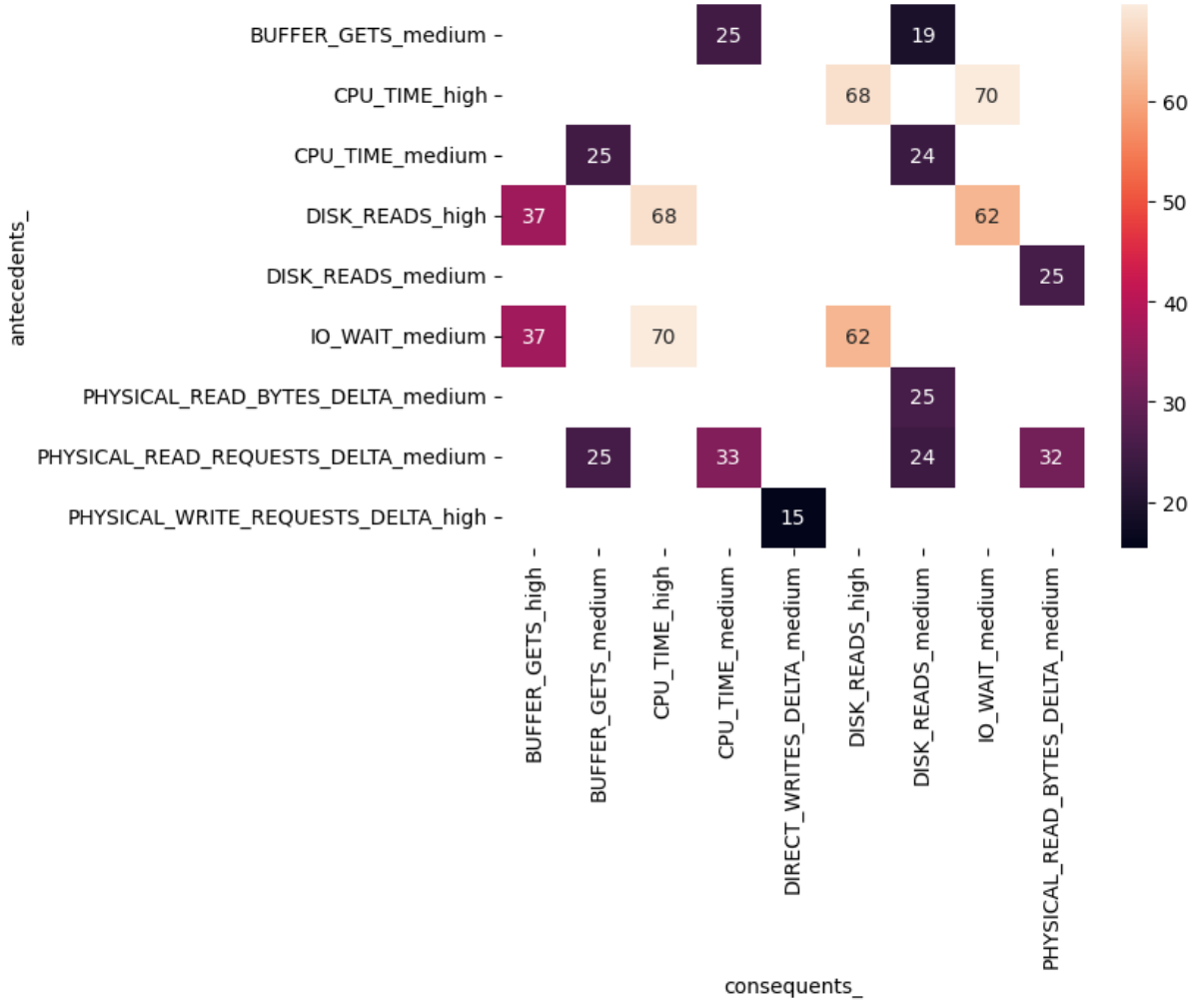

Figura 22 – Mapa de calor com suporte mínimo 0.01 e confiança 0.7 gerou novas regras fortes para o experimento 4

5.1.5 Experimento 5

Este experimento foi realizado com o *dataset* 4. Observando-se os resultados obtidos com o experimento detalhado na seção 4.2.5, observa-se que muitas regras foram geradas exatamente com todas as medidas iguais e com *lift* igual a 76. Adicionalmente, muitas regras foram geradas com mais do que um antecedente e consequente, dificultando ainda mais a extração de conhecimento das mesmas.

Ao se filtrar regras apenas com um antecedente e um consequente, mantém-se uma regra com *lift* igual a 76. Essa regra indica que um alto consumo de BUFFER_CACHE resulta em um alto consumo de CPU. Mesmo diminuindo o suporte mínimo para 0.001 ao gerar *itemsets*, foi mantida a mesma regra com o mesmo valor de *lift*. Regras com *lift* maiores só foram geradas com mais de um antecedente ou consequente.

Adicionalmente, notou-se também no experimento 5 a presença de muitas regras que talvez sejam óbvias para este domínio. Por exemplo, pode-se citar a regra que salienta que um pedido (REQUEST) de leitura ou escrita implica em BYTES lidos ou escritos. Com base nessas observações, foram removidos os atributos de pedido (REQUEST) e foram mantidos os atributos relacionados à escrita ou leitura efetiva, a saber: PHYSICAL_-WRITE_REQUESTS_DELTA médio e alto, PHYSICAL_READ_REQUESTS_DELTA médio e alto. Os resultados obtidos são ilustrados em forma de Grafos na Figura 23.

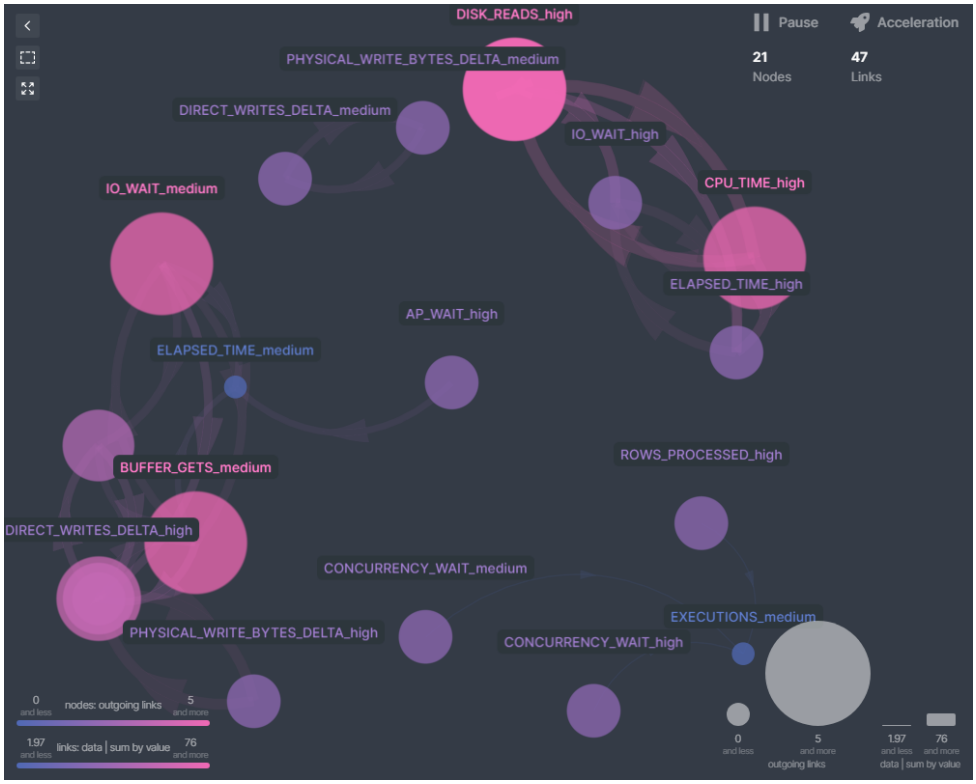

Figura 23 – Grafo utilizando a espessura das ligações para representar a medida *lift* e tamanho do nó representando quantidade de regras para o experimento 5, com suporte 0.001 e confiança 0.7

E, alternativamente, temos os resultados também exibidos na forma de mapa de calor, na Figura 24.

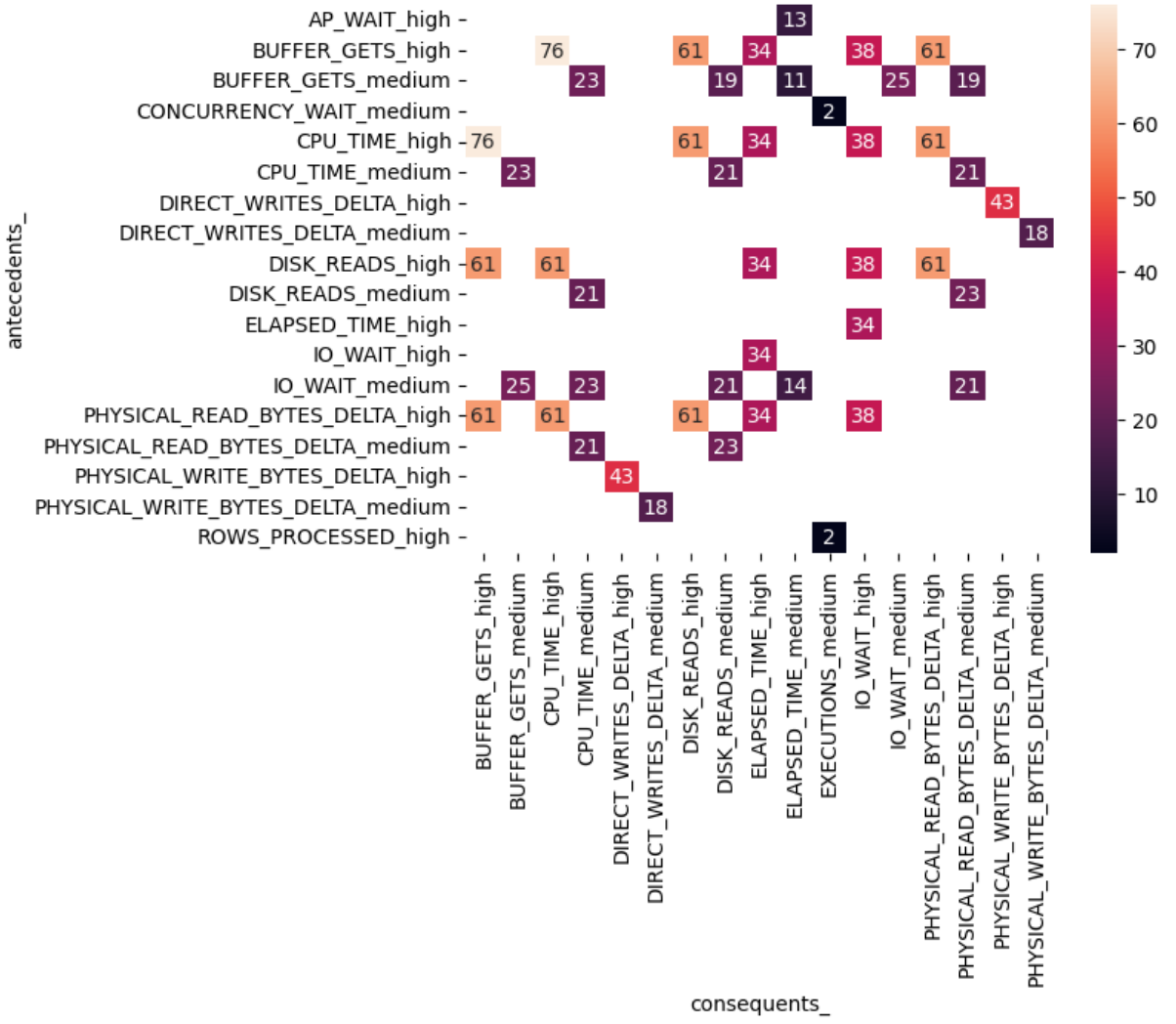

Figura 24 – Mapa de calor com suporte mínimo 0.01 e confiança 0.7 para o experimento 5

5.2 Resultados e Discussões

Análises comparando os resultados obtidos pelos experimentos 2 e 3 são discutidas na seção 5.2.1. Já os resultados obtidos pelos experimentos 4 e 5 são analisados na seção 5.2.2.

5.2.1 Experimentos 2 e 3

Analisando-se os experimentos 2 e 3, é possível observar que o experimento 3, o qual investiga um *dataset* reduzido, é mais capaz de representar a situação na qual o problema aconteceu. Entretanto, apesar dessa observação, destaca-se que ambos os experimentos geram uma regra com *lift* elevado considerando a escrita física em disco alta e também muito acesso ao *buffer cache*.

Como descrito na seção 2.3, supõe-se que existe uma situação na qual a coleta de estatísticas da madrugada possa ter tentando otimizar planos de acesso colocando mais dados em memória e o tamanho desta não foi aumentado. Entende-se portanto, que este poderia ser uma das causas do problema. Adicionalmente, acredita-se que a regra com *lift* mais alto, descoberta ao reduzir o suporte e confiança, apesar de mostrar que houve bastante escrita direta em disco, não tem como consequente alto tempo transcorrido ou alta tempo em CPU.

Escolhe-se, portanto a regra na qual o alto número de acesso ao BUFFER_CACHE levou ao consequente consumo elevado de CPU como provável causa do problema.

5.2.2 Experimentos 4 e 5

Os experimentos 4 e 5 remetem a uma situação na qual houve atividade de limpeza de dados da aplicação. Essa atividade causou impacto na disponibilidade da mesma, requerendo intervenção em conferência para tentativa de investigação e resolução do problema.

O experimento 4 mostrou como regra com maior medida *lift* uma média de espera em eventos de entrada e saída IO_WAIT levando ao alto uso de CPU_TIME (lift 70), alto número de operações de leitura DISK_READS (lift 62) e também alto número de operações de BUFFER_GETS (lift 37).

Ao realizar o experimento 5, inicialmente obteve-se centenas de regras com todas as medidas iguais e *lift* igual a 76 (Figura 16). Após filtradas as regras com apenas um antecedente e um consequente (seção 5.1.5), foram obtidas mais regras. Escolhe-se como melhor regra que o alto número de BUFFER_GETS (*lift* 76) indica a alto consumo de CPU e o alto número de DISK_READS (*lift* 61).

É difícil interpretar as regras obtidas, principalmente neste experimento 5, uma vez que as centenas de regras geradas com bastante itens antecedentes ou consequentes e

valores de todas as medidas idênticas tornava praticamente impossível escolher uma regra. O processo de filtrar por regras que contenham apenas um antecedente e consequente facilitou um pouco a interpretação das mesmas, de forma que fosse possível escolher as duas ou três regras com melhor medida *lift*.

6 CONCLUSÕES

Neste capítulo é apresentada a conclusão do trabalho desenvolvido. Na seção 6.1 é resumido o trabalho realizado. Na seção 6.2 são discutidas as dificuldades encontradas no desenvolvimento do trabalho. Finalmente, na seção 6.3 são listadas sugestões de trabalhos futuros.

6.1 Revisão dos Resultados

Este trabalho teve como objetivo principal auxiliar os administradores de bancos de dados relacionais (DBAs) a extrair conhecimento de tais sistemas em situações de problemas reportados pela empresa. É conhecido que as próprias ferramentas possuem relatórios que também podem auxiliar o DBA, mas às vezes não se tem acesso ou deseja-se fazer algum outro tipo de exploração ou ainda o SGBD não dispõe de tal funcionalidade.

Para tal, fui considerada uma aplicação de uma empresa e foi proposta a utilização de regras de associação para encontrar os atributos que mais ocorreram no momento em que a aplicação passava por algum problema. As regras de associação também tiveram como objetivo servir de base para encontrar possíveis relações entre estes atributos. Na literatura, existem vários cenários de uso das regras de associação em transações do tipo cesta/carrinho de compras. Elas podem auxiliar a loja a sugerir produtos ao cliente de acordo com a extração do conhecimento ou até mesmo facilitar a disposição de produtos em loja física. No domínio de aplicação deste trabalho, o contexto foi diferente. Apesar de ainda se ter transações, os datasets estudados contêm, em cada linha, a utilização de cada atributo pelo SGBD para cada consulta realizada no banco de dados, agrupados em janelas de, geralmente, quinze minutos. É como se a "compra" em transações do tipo cesta/carrinho fosse de "recursos de memória", "disco" e "CPU", por exemplo, no caso do presente trabalho.

Os dados disponibilizados pelo SGBD representam medidas quantitativas. Em contrapartida, os algoritmos de regras de associação trabalham com datasets que contenham transações com dados qualitativos. Então, realizou-se o processo de discretização dos dados em faixas de valores baixos, médios e altos para cada atributo da transação. A discretização foi realizada por meio da interface de pré-processamento do pacote sklearn KBinsDiscretizer. Também foi aplicado o algoritmo APRIORI para identificação de items frequentes e para a descoberta de regras de associação.

Foram gerados 4 datasets, sendo realizados 5 experimentos utilizando esses datasets. O dataset 1 não pode ser analisado devido à falta de memória para execução. Esse dataset armazena uma grande quantidade de atributos e uma grande quantidade de transações,

dificultando as análises a serem realizadas. Uma vez que o problema investigado neste trabalho tem como objetivo encontrar a alta utilização de algum recurso que possa levar a algum problema, os atributos com baixa utilização de recursos obtidos pelo algoritmo de discretização foram removidos, diminuindo também o volume do dataset.

Para os dados do dataset 2, que contém dados referentes ao intervalo de vários dias, e utilizando um suporte bem baixo (0.001), obteve-se a regra com maior medida *lift* (425) como sendo relacionada ao tempo de espera da aplicação (ou seja, transação sem commit ou inatividade). Entretanto, esta regra e algumas outras não foram geradas para os experimentos neste mesmo conjunto de dados utilizando um suporte maior de 0.01. Essa outras regras são relacionadas ao consumo alto de BUFFER_GETS, CPU e espera em entrada e saída tendo como consequente o tempo transcorrido médio.

Entretanto, observou-se que, o objetivo da análise consiste em encontrar dados sobre problemas que podem ter durado alguns minutos ou horas, extrair regras para um intervalo de vários dias poderia não ser muito preciso ou representativo. Para tanto, foi utilizado o dataset 3, o qual contém dados referentes a algumas horas nas quais o problema ocorreu. Obteve-se uma regra relacionada ao atributo DIRECT_WRITES alto levando à alta escrita de dados com valor de *lift* igual a 181. Entretanto, esta regra foi obtida apenas ao ao reduzir o suporte mínimo para 0.003. Se mantido 0.01, ainda existe a regra de alto consumo de BUFFER_GETS tendo como consequente alto consumo de CPU. A Tabela 12 também mostra a consulta 37apnmb6vfuar como sendo as que atingiram valor máximo na janela de tempo das 15 horas e 15 minutos, tanto em CPU como em BUFFER_GETS.

Na situação do segundo problema investigado por meio do dataset 4, que foi percebido quando existia um script de limpeza de dados de tabelas em execução, ao analisar todos os dias, notou-se no geral que houve muito evento de espera em entrada e saída (IO_WAIT) tendo como consequência alto tempo em CPU (lift 70), alta leitura em disco DISK_READS (lift 62), seguida pelo consequente BUFFER_GETS alto (lift 37). Ao analisar o intervalo de algumas horas do problema apenas, o atributo de espera em entrada e saída saiu de evidência, e em seu lugar obteve-se um alto número de BUFFER_GETS (lift 76) levando ao consequente tempo alto de CPU e às altas leituras em disco DISK_READS (lift 61). A regra de IO_WAIT gerada tem como consequente tempo transcorrido alto (lift 34). A Tabela 18 mostra a consulta gm2brdhxrqsw6 também como sendo comum entre os atributos gerados nas regras e alguns outros, DISK_READS, BUFFER_GETS, CPU_TIME, IO_WAIT, ELAPSED_TIME, PHYSICAL_READ_REQUESTS_DELTA e PHYSICAL_READ_BYTES_DELTA. Isso mostra que, apesar das Regras de Associação serem difíceis de serem interpretadas, ao correlacioná-las a outros tipos de exploração, parece que a correlação existe, mostrando que talvez elas possam ser utilizadas, mesmo tendo todas estas limitações de compreensão.

6.2 Dificuldades

Foram encontradas algumas dificuldades no desenvolvimento do trabalho, as quais são listadas a seguir.

- É difícil saber inicialmente qual suporte mínimo utilizar para gerar os itemsets mais frequentes;
- O número de regras que se obtém também é, as vezes, impossível de ser humanamente compreendido. Portanto, deve ser utilizado algum critério de filtro, a depender do domínio do dataset explorado. No trabalho desenvolvido, foi utilizado o critério de filtrar as regras que só continham um antecedente e consequente para ajudar na interpretação das mesmas;
- Há necessidade do conhecimento do domínio da aplicação para escolha das regras, mesmo após certa filtragem, pois regras que não fazem sentido podem ser geradas;
- Notou-se também que, apesar de algumas regras possuírem *lift* mais alto, elas nem sempre são as melhores regras que representam a situação que descreve o problema estudado;
- Houve dificuldade na análise e seleção das regras geradas. Essa dificuldade foi investigada pela representação dos resultados em formato de grafo contendo o tamanho dos nós relacionado ao número de arestas que saem do nó. Isso contribuiu para visualizar quais são os atributos antecedentes que participam mais das regras geradas.

6.3 Trabalhos Futuros

Como sugestão para trabalhos futuros, pode-se citar:

- A possibilidade de se definir novos valores de suporte e confiança e investigar os resultados obtidos em termos de regras geradas.
- A aplicação de outros algoritmos de mineração de dados, como algoritmos de agrupamento.
- A comparação dos resultados obtidos e descritos neste trabalho, os quais foram gerados com a aplicação do algoritmo APRIORI, com os outros algoritmos investigados.
- Possibilidade de montar uma base de conhecimento relacionando regras obtidas em determinadas situações, na tentativa de realizar uma rotulagem manual.

REFERÊNCIAS

AGGARWAL, C. C. **Data Mining**. Springer International Publishing, 2015. 1–6 p. Available at: <https://doi.org/10.1007%2F978-3-319-14142-8>.

AGRAWAL, R.; SRIKANT, R. Fast algorithms for mining association rules in large databases. *In*: **Proceedings of the 20th International Conference on Very Large Data Bases**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1994. (VLDB '94), p. 487–499. ISBN 1558601538.

BLANCHARD, J. *et al.* Exploratory visualization for association rule rummaging. *In*: **KDD-03 workshop on multimedia data mining (MDM-03)**. [*S.l.: s.n.*], 2003. v. 3.

BODON, F. A trie-based apriori implementation for mining frequent item sequences. *In*: **Proceedings of the 1st International Workshop on Open Source Data Mining: Frequent Pattern Mining Implementations**. New York, NY, USA: Association for Computing Machinery, 2005. (OSDM '05), p. 56–65. ISBN 1595932100. Available at: <https://doi.org/10.1145/1133905.1133913>.

BRAMER, M. **Principles of Data Mining**. 2nd. ed. [*S.l.: s.n.*]: Springer Publishing Company, Incorporated, 2013. ISBN 1447148835.

BRIN, S. *et al.* Dynamic itemset counting and implication rules for market basket data. **SIGMOD Rec.**, Association for Computing Machinery, New York, NY, USA, v. 26, n. 2, p. 255–264, jun 1997. ISSN 0163-5808. Available at: <https://doi.org/10.1145/253262.253325>.

DUNDJERSKI, D.; TOMAšEVIć, M. Automatic database troubleshooting of azure sql databases. **IEEE Transactions on Cloud Computing**, v. 10, n. 3, p. 1604–1619, 2022.

HAN, J.; PEI, J.; YIN, Y. Mining frequent patterns without candidate generation. *In*: **Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data**. New York, NY, USA: Association for Computing Machinery, 2000. (SIGMOD '00), p. 1–12. ISBN 1581132174. Available at: <https://doi.org/10.1145/342009.335372>.

HOFMANN, H.; SIEBES, A. P.; WILHELM, A. F. Visualizing association rules with interactive mosaic plots. *In*: **Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining**. [*S.l.: s.n.*], 2000. p. 227–235.

IVKOVIC, S.; YEARWOOD, J.; STRANIERI, A. Visualizing association rules for feedback within the legal system. *In*: **Proceedings of the 9th International Conference on Artificial Intelligence and Law**. New York, NY, USA: Association for Computing Machinery, 2003. (ICAIL '03), p. 214–223. ISBN 1581137478. Available at: <https://doi.org/10.1145/1047788.1047835>.

KOTU, V.; DESHPANDE, B. Chapter 6 - association analysis. *In*: KOTU, V.; DESHPANDE, B. (ed.). **Data Science (Second Edition)**. Second edition. Morgan Kaufmann, 2019. p. 199–220. ISBN 978-0-12-814761-0. Available at: <https://www.sciencedirect.com/science/article/pii/B978012814761000006X>.

MITCHELL, T. **Machine Learning**. McGraw-Hill Education, 1997. (McGraw-Hill international editions - computer science series). ISBN 9780070428072. Available at: <https://books.google.com.br/books?id=xOGAngEACAAJ>.

NAVARRO, J. M.; HUET, A.; ROSSI, D. Human readable network troubleshooting based on anomaly detection and feature scoring. **Computer Networks**, v. 219, p. 109447, 2022. ISSN 1389-1286. Available at: <https://www.sciencedirect.com/science/article/pii/S1389128622004819>.

ORACLE: Documentação oracle sobre a visualizações estáticas. 2023. Available at: <https://docs.oracle.com/en/database/oracle/oracle-database/12.2/refrn/static-data-dictionary-views.html#GUID-8865F65B-EF6D-44A5-B0A1-3179EFF0C36A/>. Access at: 31 ago. 2023.

ORACLE: Documentação oracle sobre a área de memória buffer cache. 2023. Available at: <https://docs.oracle.com/en/database/oracle/oracle-database/23/dbiad/db_dbbuffercache.html>. Access at: 25 nov. 2023.

ORACLE: Documentação oracle sobre a visualização dba_hist_sqlstat. 2023. Available at: <https://docs.oracle.com/en/database/oracle/oracle-database/12.2/refrn/DBA_HIST_SQLSTAT.html#GUID-F5A246E0-C04A-406C-9E10-AC26E7742F06/>. Access at: 21 ago. 2023.

RAMíREZ-GALLEGO, S. *et al.* Data discretization: taxonomy and big data challenge. **WIREs Data Mining and Knowledge Discovery**, v. 6, n. 1, p. 5–21, 2016. Available at: <https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/widm.1173>.

SCIKIT-LEARN: Documentação scikit-learn sobre o kbinsdiscretizer. 2023. Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.KBinsDiscretizer.html>. Access at: 23 ago. 2023.

TAN, P.-N.; STEINBACH, M.; KUMAR, V. **Introduction to Data Mining, (First Edition)**. USA: Addison-Wesley Longman Publishing Co., Inc., 2005. ISBN 0321321367.

WITTEN, I. H.; FRANK, E. **Data mining**. 2. ed. Oxford, England: Morgan Kaufmann, 2005. (The Morgan Kaufmann Series in Data Management Systems).

YAMAMOTO, C. H. **Visualização como suporte à extração e exploração de regras de associação**. 2009. Tese (Doutorado) — Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, Brasil, 2009.