

Relatório do projeto MAI-DAI de iniciação científica pela UTFPR.

Introdução:

O seguinte projeto trabalha com o desenvolvimento de um sistema de mineração de dados frequentes com base temporal. Para isso, têm-se, na literatura, referências de algoritmos já bem consolidados que executam essa tarefa de forma otimizada. No caso, foram considerados os algoritmos FP-Growth, o Eclat e o Apriori, e, depois de estudos, o projeto foi iniciado com a premissa de utilizar-se do segundo algoritmo mencionado. O sistema foi planejado para receber um arquivo que contenha dados temporais, como transações bancárias, chamadas telefônicas, ou qualquer outro tipo de informação que possa ser caracterizada como efetuada em um único momento no tempo. Com isso, é possível encontrar relações ou padrões recorrentes entre diferentes atividades no tempo contínuo.

Objetivos:

Dado um conjunto de dados que possuem informações temporais, têm-se como objetivo encontrar padrões e regras dentro de alguns parâmetros de tempo e qualidade que possam dizer alguma informação importante sobre o comportamento dos dados. Para isso, esse conjunto de dados deve ser pré-processado, passando por diversas manipulações a fim de transformá-lo em uma entrada adequada e otimizada ao algoritmo. O seu processamento é feito utilizando uma implementação de apriori já existente. Após isso, é necessário um pós-processamento dos dados. A saída do algoritmo deve ser formatada de forma que o usuário possa interpretá-la facilmente e que a análise das regras seja simples e direta.

Termos técnicos essenciais:

A mineração de dados é utilizada para análise automatizada de grandes bases de dados, as quais nos referiremos como datasets. O conhecimento obtido através das técnicas de mineração de dados são geralmente apresentadas em forma de regras ou padrões entre os itens dessa base de dados, onde um conjunto A, de um ou mais itens, se relaciona com outro conjunto B, de um ou mais itens, formando uma relação de antecedente e consequente.

Dentro da mineração de dados, existem alguns conceitos importantes cujo conhecimento prévio auxilia no melhor entendimento do projeto:

Suporte: é uma medida bilateral, ou seja, ela independe da direção de uma relação que poderá compor uma regra. Seu valor representa a porcentagem de vezes que os

elementos de uma relação aparecem juntos dentre todos os conjuntos de dados (baldes) que serão analisados.

Confiança: é uma medida unilateral, ou seja, a direção da relação importa. Isso depende, muitas vezes, da natureza do dado analisado e se ele de fato possui direção. Por exemplo: chamadas telefônicas, transações bancárias e envio de mensagens partem de um remetente (origem) até um destinatário (destino). Dessa forma, uma relação em que o antecedente A implica em um consequente B pode ter confiança distinta de outra em que o antecedente B implica o consequente A. Em suma, a confiança indica a probabilidade condicional do antecedente implicar um dado consequente.

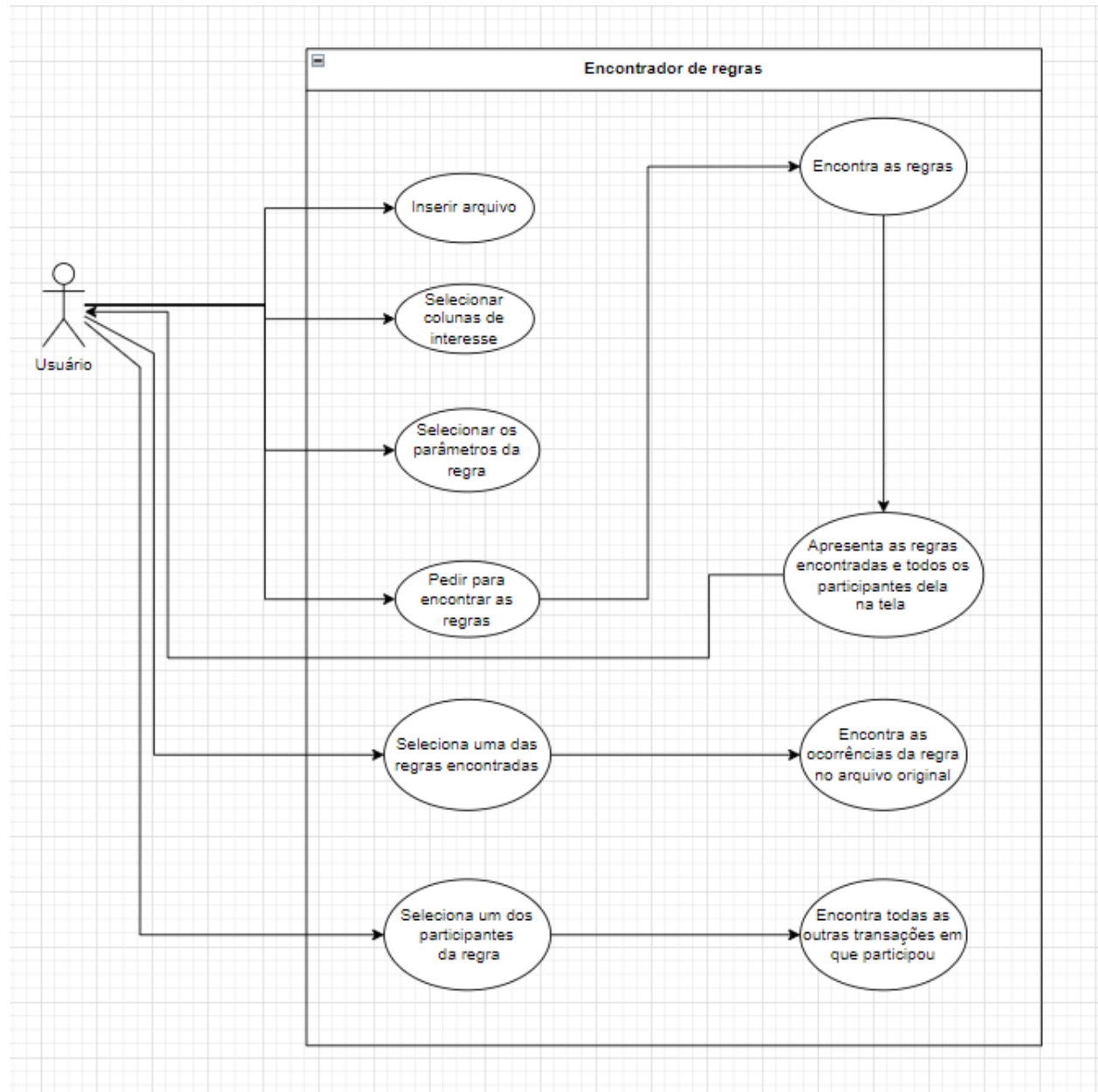
Itemsets frequentes: subconjuntos de elementos que frequentemente aparecem juntos em conjuntos mais gerais.

Regras associativas: é uma maneira de representar padrões, associações, correlações ou estruturas causais entre subconjuntos de elementos frequentes entre conjuntos de dados relacionais, transacionais, entre outros.

Ideia Geral do Sistema:

Com o uso de Python, é gerada uma interface gráfica simples que permite que arquivos sejam arrastados e soltos em uma janela. O caminho do arquivo então é utilizado para que o seu conteúdo seja aberto no sistema como um Data Frame, utilizando-se da biblioteca "Pandas" do Python, focada em análise de dados. Depois disso, é requisitado ao usuário o preenchimento das informações pertinentes à mineração dos dados e busca de padrões. Com essas informações, detalhadas mais adiante no documento, o sistema seleciona e corta linhas e colunas do dataset de entrada, retirando todas as tuplas que não correspondem aos critérios da busca, para que, somente então, sejam feitos os baldes que serão submetidos ao algoritmo apriori.

A saída do apriori, que são as regras associativas encontradas com os parâmetros fornecidos, é então formatada e apresentada ao usuário, que possui algumas opções. Ao selecionar uma das regras, o sistema procura todas as suas ocorrências dentro do dataset original, recuperando todas as colunas de informação não relacionadas com o padrão em si. Além disso, é possível selecionar apenas um elemento da regra e encontrar todas as suas relações com outros elementos fora da regra.



Método:

O algoritmo utilizado no sistema foi o Apriori. Outros algoritmos, como o eclat e o fp-growth foram considerados, no entanto, o apriori possui uma saída mais compatível com o objetivo do sistema, sendo capaz de encontrar regras associativas entre antecedentes e consequentes, enquanto outros algoritmos retornam apenas conjuntos de elementos frequentes.

Estrutura dos Dados:

O sistema espera um arquivo de dados, formatado em csv ou xlsx. Como a análise dos dados é temporal, é necessário no mínimo 2 colunas, sendo elas:

1...n coluna(s) de dados relativa à identificação

1 coluna de dados relativa à informação temporal

É possível que a identificação de uma atividade seja composta por mais de uma coluna, nesse caso, o conteúdo das colunas fornecida pelo usuário é concatenada no pré-processamento, gerando uma única coluna identificadora, para que os dados sejam processados corretamente pelo apriori. A mudança é revertida antes do resultado ser apresentado ao usuário.

Com isso, sendo especificado pelo usuário quais são essas colunas correspondentes nos dados fornecidos, é possível realizar o reconhecimento de padrões.

Conta Origem	Conta Destino	Data	Hora	Montante	DataHora
A	E	11/10/2006	9:37	469,92	11/10/2006 9:37
A	F	13/10/2006	12:00	20000	13/10/2006 12:00
B	G	13/10/2006	12:15	19000	13/10/2006 12:15
C	E	13/10/2006	12:45	18500	13/10/2006 12:45
D	H	13/10/2006	10:31	132,65	13/10/2006 10:31
A	F	13/10/2006	20:14	1500	13/10/2006 20:14
D	G	13/10/2006	12:24	716,61	13/10/2006 12:24
B	E	13/10/2006	14:01	138,83	13/10/2006 14:01
A	F	16/10/2006	16:00	25000	16/10/2006 16:00
D	G	16/10/2006	16:05	23800	16/10/2006 16:05
B	G	16/10/2006	14:15	824,07	16/10/2006 14:15
D	F	16/10/2006	16:20	109,43	16/10/2006 16:20
A	G	16/10/2006	8:04	37,07	16/10/2006 08:04
A	F	17/10/2006	16:10	2028,59	17/10/2006 16:10
B	G	17/10/2006	17:52	5,86	17/10/2006 17:52
A	H	17/10/2006	8:53	35,88	17/10/2006 08:53
A	F	18/10/2006	9:50	1,86	18/10/2006 09:50
B	G	18/10/2006	10:45	65,17	18/10/2006 10:45
C	H	18/10/2006	9:43	6638,35	18/10/2006 09:43

Tabela 1: Exemplo dado de entrada

Tomando a tabela 1 como exemplo, o usuário pode preencher os parâmetros da seguinte forma:



tk

Metadata attribute (Unique): DataHora

Atributo de origem: Conta Origem

Atributo de destino: Conta Destino

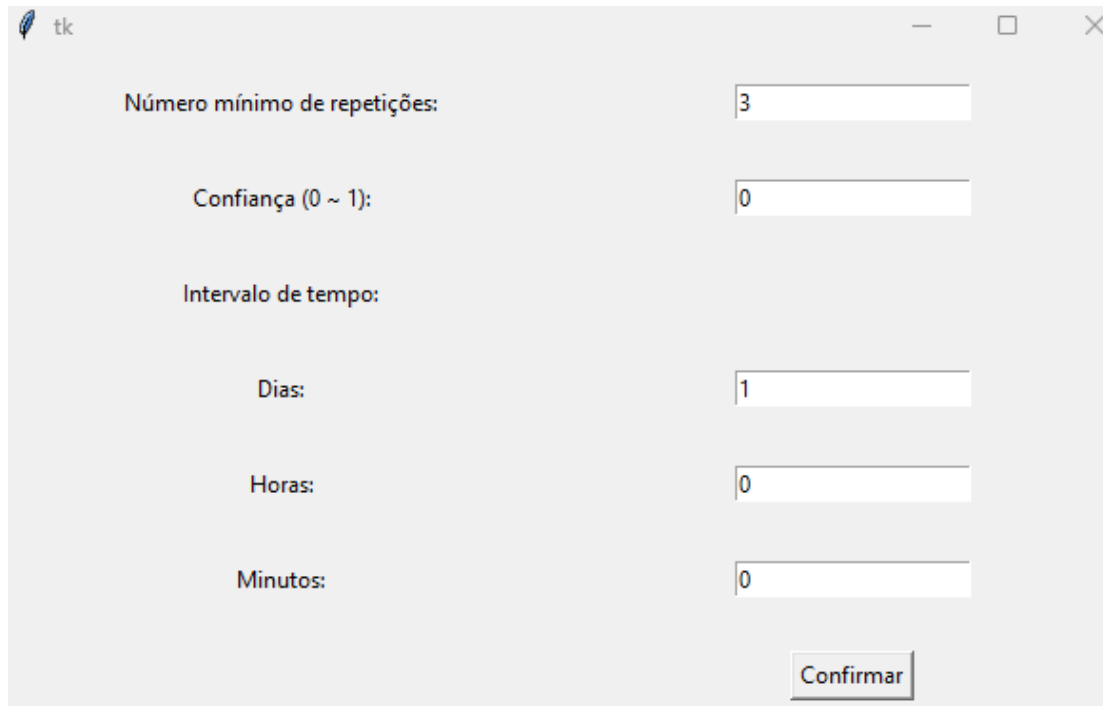
day first (dd/mm/yy) ▾

Confirm

Parâmetros das regras associativas:

Além de fornecer as colunas de interesse, o usuário deve também fornecer alguns parâmetros que limitam a procura das regras. É necessário indicar o número mínimo de repetições para as regras e a sua confiança mínima, descrita anteriormente. Como a confiança é um termo mais avançado, o usuário pode optar por não informá-la e mantê-la como 0. Além disso, também é necessário indicar a janela temporal em que eventos serão agrupados, em intervalos de minutos, horas, dias ou uma combinação desses.

Todos esses parâmetros serão utilizados para transformar os dados de entrada e formar a entrada do algoritmo apriori, que é dada em um conjunto formado de subconjuntos, chamados de baldes ou cestas.



A screenshot of a Tkinter window titled 'tk'. It contains several input fields for configuring parameters. The parameters and their values are:

Parameter	Value
Número mínimo de repetições:	3
Confiança (0 ~ 1):	0
Intervalo de tempo:	
Dias:	1
Horas:	0
Minutos:	0

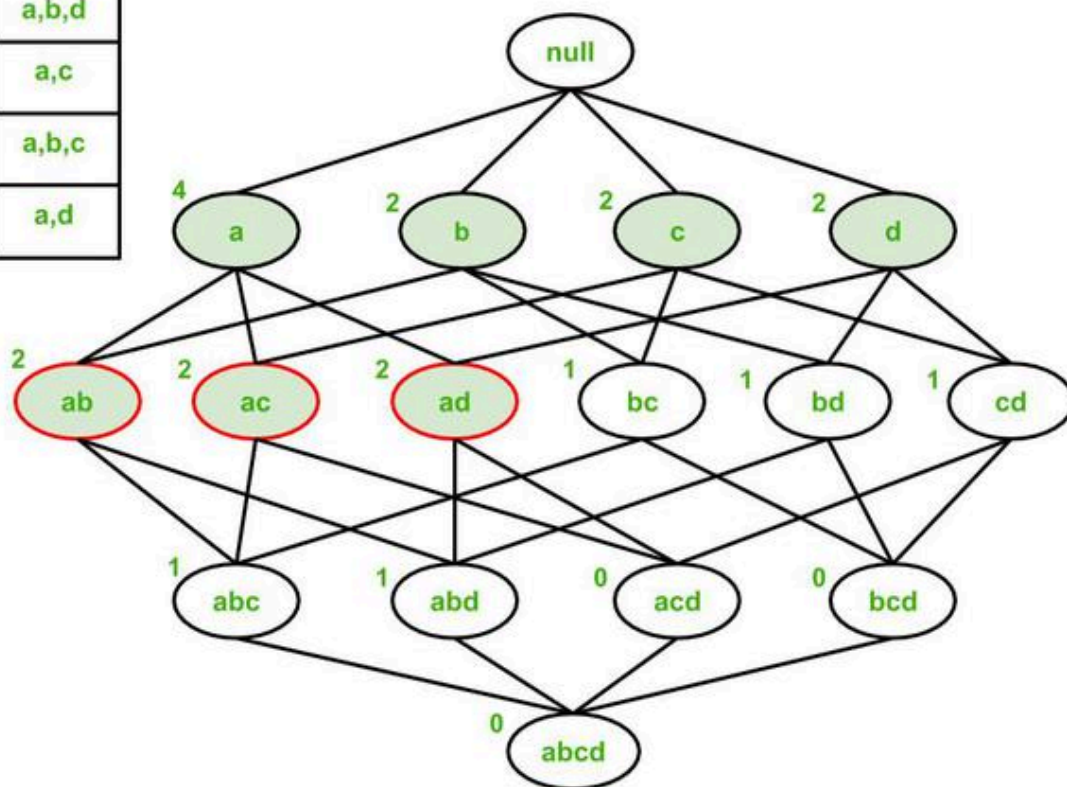
At the bottom right, there is a button labeled 'Confirmar'.

Formação dos baldes:

Na mineração de dados, existe um tipo de análise conhecida como “análise de cestas de mercado” ou “análise de regras de associação”. Visto isso, o conceito de “cestas” ou “baldes” referem-se a conjuntos de itens que frequentemente ocorrem juntos.

O objetivo principal é descobrir padrões de associação entre diferentes itens em grandes conjuntos de dados.

TID	Items
1	a,b,d
2	a,c
3	a,b,c
4	a,d



Analisando a imagem, cada “TID” corresponde a um balde, ou seja, existem 4 baldes nessa análise. No caso da entrada da tabela 1, cada item na imagem corresponde a uma transação entre uma origem e um destino, ou seja, o item “a” pode representar uma transação “A->B”, o item “b” pode representar “C->D” e assim por diante. Assim, o item “a” e o item “b” são os identificadores de cada transação.

Para isso, como os dados utilizados naturalmente não são divididos em baldes, é necessário que isso seja feito para que possamos aplicar o algoritmo e descobrir as frequências.

Podem existir diversos critérios diferentes para formação de baldes, mas aqui, o objetivo é encontrar padrões temporais, ou seja, os baldes serão formados a partir de janelas de tempo. Dessa forma, serão formados baldes com eventos que ocorreram em um dado intervalo de tempo para que o resultado final sejam os eventos dentro desses baldes que sejam recorrentes, ou seja, que apareçam em diversos baldes, com o passar do tempo.

Analisando a imagem novamente, é possível perceber que, conforme descemos no mapa de frequências, as frequências somente diminuem. Isso é lógico e intuitivo, já que um conjunto contendo A precisa que A exista.

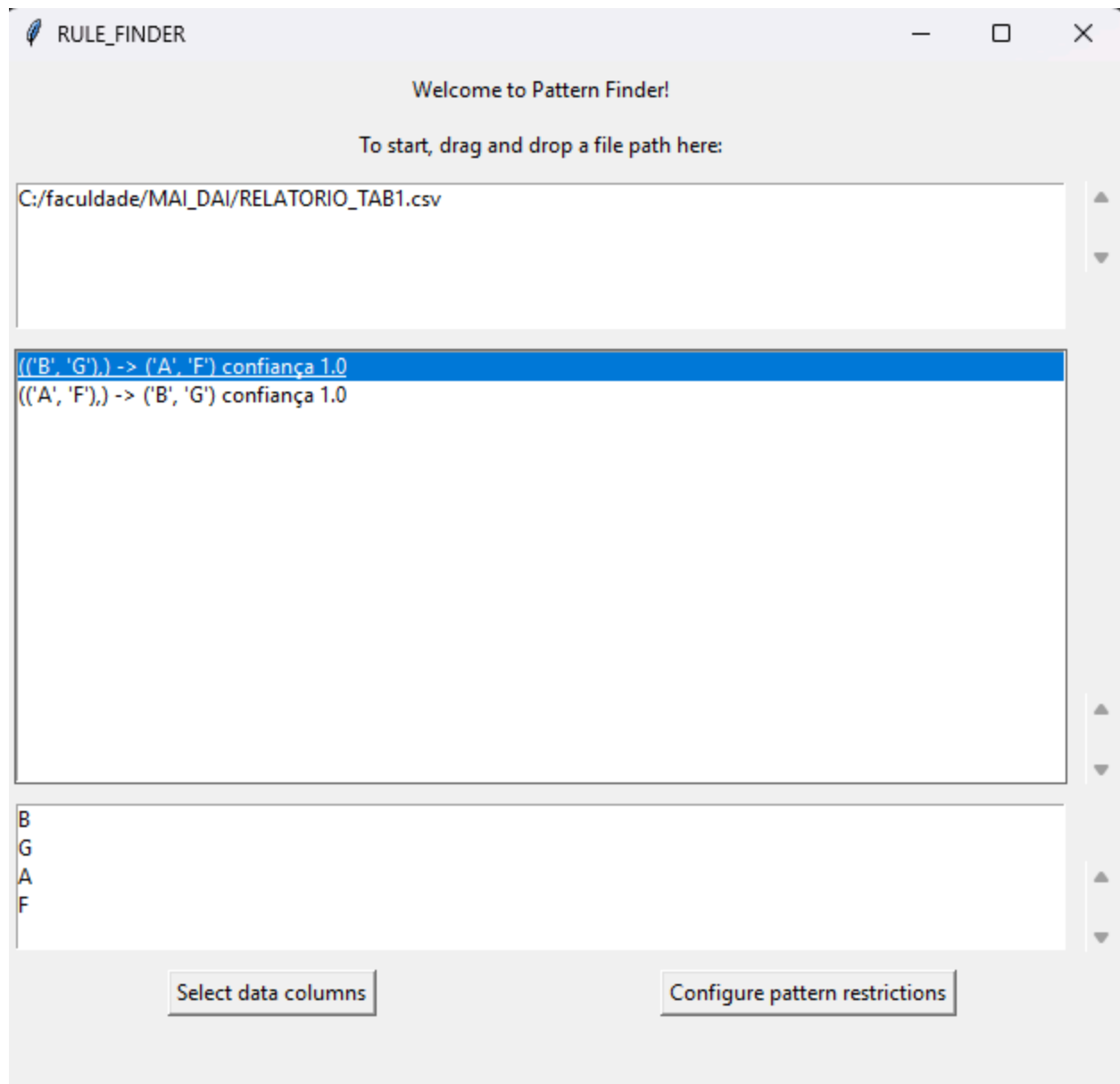
Podemos expandir isso para o nosso problema e afirmar que um subconjunto de A que apareça x vezes precisa que A apareça x vezes. Assim, para evitar a construção de um mapa de frequências tão grande, podemos fazer um pré-processamento e eliminar todas as tuplas que aparecem menos que o desejado antes mesmo de começar a construção dos baldes.

Após o pré-processamento, é aplicada a janela de tempo fornecida e os baldes a seguir são formados. Cada balde é um subconjunto de um conjunto de baldes, que é submetido ao algoritmo, juntamente do número mínimo de repetições e de uma eventual confiança.

A	B	C	D	E
Bucket 1	Bucket 2	Bucket 3	Bucket 4	Bucket 5
('A', 'E')	('D', 'H')	('A', 'G')	('A', 'H')	('A', 'F')
	('A', 'F')	('B', 'G')	('A', 'F')	('B', 'G')
	('B', 'G')	('A', 'F')	('B', 'G')	
	('D', 'G')	('D', 'G')		
	('B', 'E')	('D', 'F')		
	('A', 'F')			

Tabela 2: Baldes gerados

Com isso, o algoritmo é capaz de encontrar as regras:



Todas as regras na saída aparecem no arquivo original pelo menos igual ao número mínimo de repetições fornecido pelo usuário.

Junto a cada regra encontrada, também é apresentada a sua respectiva confiança. No caso, uma confiança 1.0 significa que toda vez que o elemento antecedente ocorre, a ocorrência do elemento consequente no mesmo balde é garantida. A confiança indica a probabilidade condicional da presença dos elementos, assim, se a confiança fosse 0.8, por exemplo, significa que em 80% das vezes em que o antecedente ocorre, o consequente ocorre.

A interpretação da regra depende dos parâmetros fornecidos.

No exemplo dado, existem atividades em 5 dias. A janela de tempo foi de 1 dia e o número mínimo de repetições é 3. Assim, as regras encontradas serão pertinentes a quais dessas atividades ocorreram juntas pelo menos 3 vezes em um intervalo máximo de 1 dia uma da outra.

Ao selecionar uma das regras, o sistema dá a opção de analisar, por elemento, todas as suas ocorrências no dataset original, para além da regra. Ademais, é gerada uma tabela contendo todas as ocorrências da regra encontrada, como pode ser verificado a seguir.

A	B	C	D	E	F
	Conta Origem	Conta Destino	Data	Hora	Montante
2006-10-13 12:00:00	A	F	13/10/2006	12:00	20000
2006-10-13 12:15:00	B	G	13/10/2006	12:15	19000
2006-10-13 20:14:00	A	F	13/10/2006	20:14	1500
2006-10-16 14:15:00	B	G	16/10/2006	14:15	824,07
2006-10-16 16:00:00	A	F	16/10/2006	16:00	25000
2006-10-17 16:10:00	A	F	17/10/2006	16:10	2028,59
2006-10-17 17:52:00	B	G	17/10/2006	17:52	5,86
2006-10-18 9:50:00	A	F	18/10/2006	9:50	1,86
2006-10-18 10:45:00	B	G	18/10/2006	10:45	65,17

Tabela 3: Ocorrências da regra no dataset original

Selecionando o elemento A:

A ▼	B	C	D	E	F
Conta Origem	Conta Destino	Data	Hora	Montante	DataHora
A	E	11/10/2006	9:37	469,92	2006-10-11 9:37:00
A	F	13/10/2006	12:00	20000	2006-10-13 12:00:00
A	F	13/10/2006	20:14	1500	2006-10-13 20:14:00
A	G	16/10/2006	8:04	37,07	2006-10-16 8:04:00
A	F	16/10/2006	16:00	25000	2006-10-16 16:00:00
A	H	17/10/2006	8:53	35,88	2006-10-17 8:53:00
A	F	17/10/2006	16:10	2028,59	2006-10-17 16:10:00
A	F	18/10/2006	9:50	1,86	2006-10-18 9:50:00

Tabela 4: Tuplas contendo o elemento selecionado A

Testes:

Como o objetivo do sistema inclui lidar com uma massa de dados volumosa, é necessário que o método utilizado seja sólido em relação à complexidade computacional, almejando bons resultados em tempos admissíveis.

Para que seja possível se certificar que o método proposto consegue os resultados esperados, foram realizadas duas baterias de testes.

Testes de resultado: nesses testes foram implantados padrões em uma massa de dados aleatória que foi submetida ao algoritmo apriori, esperando que os padrões fossem encontrados.


Testes de desempenho: nesses testes foram geradas massas de dados sintéticos de tamanhos colossais que foram submetidas ao algoritmo apriori, esperando que o tempo de processamento se mantivesse admissível.

Testes de resultado:

Para os testes de resultado do sistema, foram utilizados diversos datasets e contextos, com tamanhos e parâmetros diferentes. Foram fornecidos datasets de transações bancárias e de ligações telefônicas com padrões encontrados por outros métodos, e os resultados do sistema foram aprovados.

Além disso, também submeteu-se o sistema à datasets sintéticos, com resultados positivos. A seguir pode-se verificar a análise de um desses testes.

Primeiramente, uma massa de dados aleatória foi gerada para que fosse utilizada:

 `forged_data`

Testes variando os parâmetros, com o objetivo de não encontrar nenhuma regra no dataset sintético gerado. Observa-se que o número de repetições é o mínimo possível para que se configure algum padrão:

1. Número mínimo de repetições = 2
Confiança = 0.7
Intervalo de tempo = 2 horas

Regras encontradas = nenhuma.

2. Número mínimo de repetições = 3
Confiança = 0.5
Intervalo de tempo = 4 horas

Regras encontradas = nenhuma.

3. Número mínimo de repetições = 2
Confiança = 0.6
Intervalo de tempo = 4 horas

Regras encontradas = nenhuma.

4. Número mínimo de repetições = 2
Confiança = 0.5
Intervalo de tempo = 6 horas

Regras encontradas = nenhuma.

Com fortes indícios de que nenhuma regra está presente no dataset, agora podemos fazer o implante de regras com o objetivo de que sejam encontradas pelo sistema.

Implante de regras no dataset.

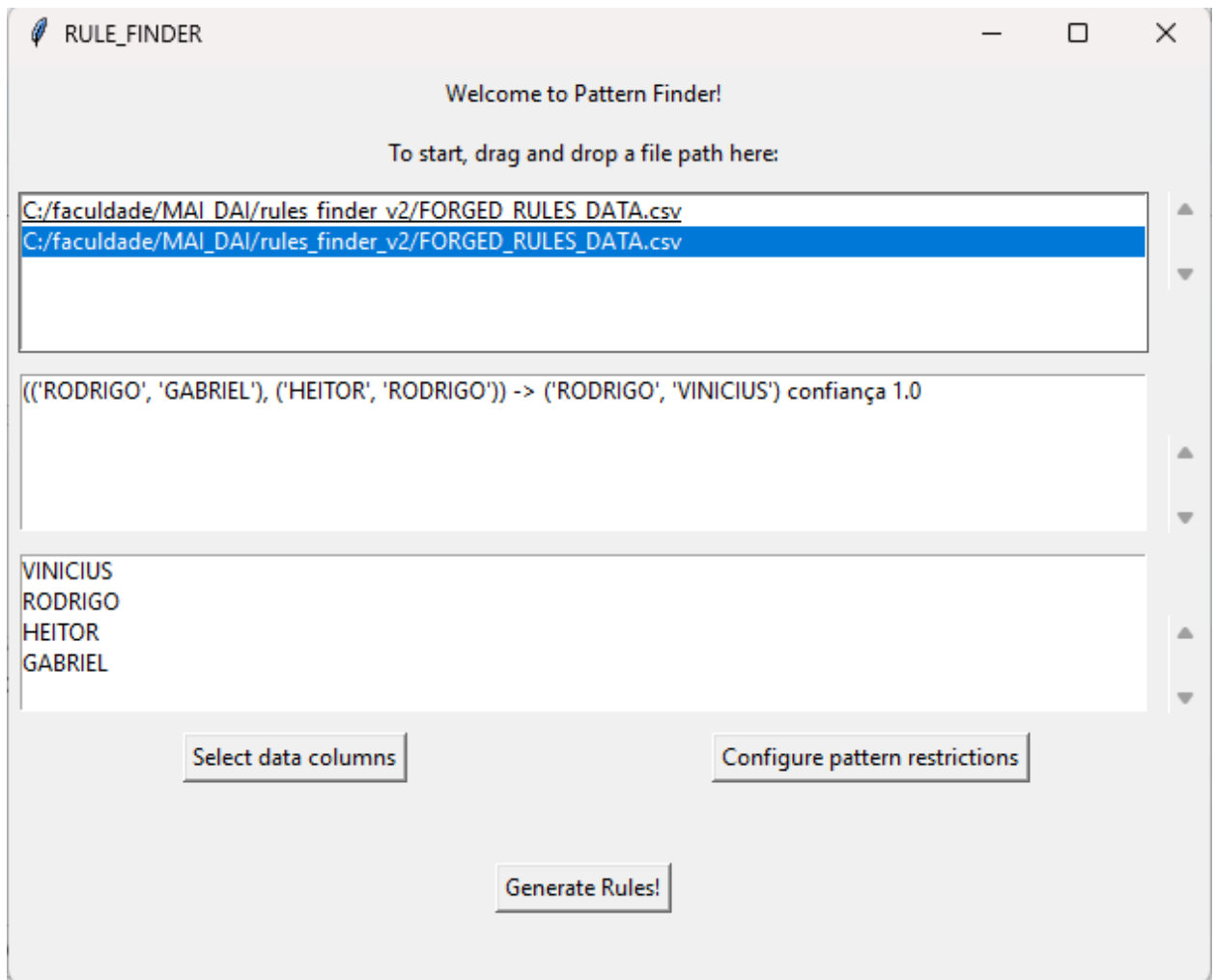
Através do google sheets, o dataset foi ordenado cronologicamente.

	A	B	C
1	timestamp	o	d
2	2023-01-01 0:15:50	Freddie Robison	4580
3	2023-01-01 0:28:53	Mattie White	1254
4	2023-01-01 0:34:57	Leslie Bolton	1007
5	2023-01-01 0:40:34	HEITOR	RODRIGO
6	2023-01-01 0:45:44	Jorge Hodge	4254
7	2023-01-01 0:55:55	RODRIGO	VINICIUS
8	2023-01-01 1:01:32	Paul Harris	4149
9	2023-01-01 1:18:13	RODRIGO	GABRIEL
10	2023-01-01 2:51:29	Tim Starkey	1799
11	2023-01-01 3:05:47	Arlene Murray	3604
12	2023-01-01 3:12:55	Pricilla Williams	1911
13	2023-01-01 3:34:29	Joyce Acosta	1809
14	2023-01-01 3:38:06	John Bonsall	4668
15	2023-01-01 4:34:25	John Bonsall	1390

Tabela 5: Exemplo do implante das regras no dataset sintético

O padrão ((HEITOR->RODRIGO) ,(RODRIGO->VINICIUS)) -> (RODRIGO->GABRIEL)) foi implantado no dataset, com uma janela de tempo entre o primeiro e último acontecimento de, até, 2 horas.

Foram implantadas 5 aparições desse padrão ao longo de todo o dataset. Agora, vejamos se o algoritmo é capaz de encontrá-las.



Atribuindo os parâmetros

Número mínimo de repetições = 3

Confiança = 0.9

Intervalo de tempo = 2 horas

Regras encontradas:

((RODRIGO->GABRIEL) ,(HEITOR->RODRIGO)) -> (RODRIGO->VINICIUS))

Como queríamos.

Apesar da regra não ter sido encontrada exatamente na mesma ordem, elementos no mesmo balde são considerados simultâneos. Por isso, todos os padrões em que os elementos do antecedente e do consequente são permutados entre si são válidos.

Testes de desempenho:

Como o Apriori teve uma performance surpreendente nas observações e análises feitas durante os testes de resultado, e que a presença ou não de padrões não interferia na performance, foi proposto então um teste de desempenho com essas características:

- Datasets sintéticos
- Sem presença de padrões específicos
- Tamanhos de baldes e itens por balde pré-definidos
- Parâmetros do apriori fixos

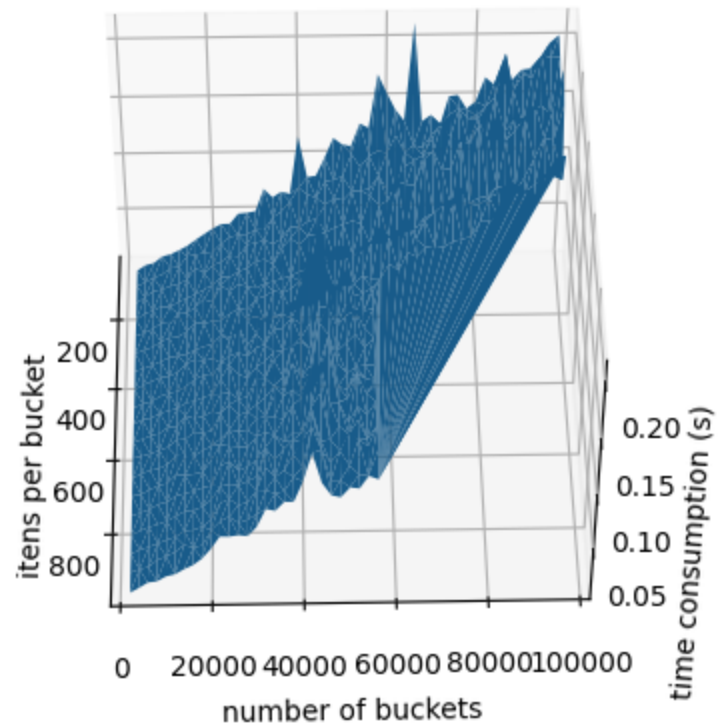
Para a execução desses testes, foram consideradas duas características da massa de dados de entrada do apriori, para melhor entender os limites do algoritmo.

1. Número de baldes (1 a 100,000)
2. Número de itens por balde (50 a 1000)

Partindo de uma análise dessas duas características, foram gerados diversos dados aleatórios de tamanhos 50 até 100,000,000, de forma que o número de baldes e de itens por balde para o apriori fossem controlados.

À medida que o número de transações se aproxima de 100,000,000, existe um problema relacionado com a capacidade de memória RAM para processar uma massa de dados tão grande. No entanto, percebe-se pelo perfil da superfície que o tempo médio de processamento do algoritmo é linearmente proporcional com o número de baldes, e que o número de itens por balde não interfere significativamente na velocidade do processamento.

3D Plot



Problemas nas formações de baldes.

Quando o número mínimo é 3, sobram mais tuplas depois da pré-filtragem e no final são formados 4 baldes, fazendo com que alguns padrões apareçam 4 vezes. No entanto, quando o número mínimo é 4, aumentando a restritividade, são formados 3 baldes, e o padrão não é encontrado... como resolver?