7:54 Bagui e Stanley J Big Data (2020)
https://doi.org/10.1186/s40537-020-00330-9

Journal of Big Data

PESQUISA Acesso livre

Mineração de conjuntos de itens frequentes de dados de transações de streaming usando algoritmos genéticos

Sikha Bagui* e Patrick Stanley

*Correspondência:
bagui@uwf.edu
Departamento de Ciência da Computação, University of West Florida, Pensacola, FL, EUA

Abstrato

Este artigo apresenta um estudo de mineração de conjuntos de itens frequentes de dados de streaming na presença de desvio de conceito. O streaming de dados, por ser volátil por natureza, é particularmente desafiador para minerar. Uma abordagem usando algoritmos genéticos é apresentada, e várias relações entre desvio de conceito, tamanho de janela deslizante e restrições de algoritmo genético são exploradas. A deriva de conceito é identificada por mudanças em conjuntos de itens frequentes.

A novidade deste trabalho está em determinar a deriva de conceito usando conjuntos de itens frequentes para mineração de dados de streaming, usando o framework de algoritmo genético. Fórmulas foram apresentadas para calcular contagens mínimas de suporte em streaming de dados usando janelas deslizantes. Os testes destacaram que a proporção entre o tamanho da janela e as transações por desvio era a chave para um bom desempenho. Obter bons resultados quando o tamanho da janela deslizante era muito pequeno era um desafio, pois as flutuações normais nos dados podiam parecer um desvio de conceito. O tamanho da janela deve ser gerenciado em conjunto com os valores de suporte e confiança para alcançar resultados razoáveis. Este método de detecção de desvio de conceito teve um bom desempenho quando tamanhos de janela maiores foram usados.

Palavras-chave: Conjuntos de itens frequentes, Dados de streaming, Deriva de conceito, Janela deslizante, Mineração de regras de associação, Algoritmos genéticos, Mineração de dados

Introdução

O mundo digital de hoje está constantemente gerando dados de sensores de tráfego, sensores de saúde, transações de clientes e vários outros dispositivos de Internet of Tings (IoT). Fluxos contínuos e intermináveis de Big Data estão criando novos conjuntos de desafios da perspectiva da mineração de dados. A mineração apenas de dados estáticos em instantâneos de tempo não é mais útil.

Os dados de streaming, sendo dinâmicos ou voláteis por natureza, têm padrões de mudança ao longo do tempo, e isso é mais tecnicamente conhecido como deriva de conceito. Algoritmos desenvolvidos para mineração de dados de streaming precisam ser capazes de detectar e trabalhar com desvios de conceito, daí a necessidade de novas abordagens de mineração de dados de streaming. Este trabalho aborda uma importante técnica de mineração de dados, a frequente mineração de conjuntos de itens, aplicada ao streaming de dados de transações, na presença de desvio de conceito.

A mineração frequente de conjuntos de itens, precursora da mineração de regras de associação, normalmente requer poder de processamento significativo, pois esse processo envolve várias passagens por um banco de dados e isso pode ser um desafio em grandes conjuntos de dados de streaming. Embora haja um grande progresso em encontrar conjuntos de itens frequentes e regras de associação em estáticas ou permanentes.

bancos de dados [1, 8], uma vez que o cenário de dados está mudando, os algoritmos precisam ser efetivamente estendidos para fluxo de dados com desvio de conceito. Há uma série de perguntas que devem ser respondidas para explorar isso: (i) Os conjuntos de itens frequentes podem ser extraídos enquanto os dados de streaming estão aparecendo sem referência a um banco de dados mestre permanente; (ii)
Existem técnicas para reduzir a complexidade de tempo de mineração de conjuntos de itens frequentes para gerenciar dados dinâmicos; (iii) Existem desafios que surgem como resultado de um ambiente de streaming? Para abordar a questão da complexidade do tempo, o uso de algoritmos genéticos foi considerado; esta técnica mostrou reduzir a complexidade de tempo de mineração de conjuntos de itens frequentes e regras de associação em bancos de dados estáticos [8].

O uso de algoritmos genéticos para minerar conjuntos de itens frequentes em fluxo de dados com desvio de conceito é relativamente pouco estudado, e há uma série de questões críticas que devem ser consideradas. Uma questão crítica especificamente explorada neste trabalho é a relação entre a taxa de desvio de conceito em dados de streaming e a convergência de um algoritmo genético usado para minerar conjuntos de itens frequentes desses dados.

Neste estudo, algoritmos genéticos são usados para minerar conjuntos de itens frequentes de dados de streaming na presença de desvio de conceito, com o auxílio de janelas deslizantes. A novidade deste trabalho está em determinar a deriva de conceito usando conjuntos de itens frequentes para mineração de dados de streaming, usando o framework de algoritmo genético.

O teste é feito usando um equipamento de teste, permitindo o streaming de dados conforme necessário. O equipamento de teste também permite o controle de algumas das variáveis fundamentais, incluindo os dados que estão sendo transmitidos, o número de transações antes que um desvio de conceito seja introduzido e o tamanho da janela deslizante. As outras variáveis, que estão relacionadas ao algoritmo genético, incluem o tamanho da população, probabilidade de cruzamento, probabilidade de mutação e a taxa de convergência do algoritmo genético.

O restante deste artigo é apresentado a seguir. Para fornecer o pano de fundo e o contexto, na introdução, é apresentada uma visão geral de alto nível de fluxo de dados em uma janela deslizante, desvio de conceito, conjuntos de itens frequentes de janela deslizante e, finalmente, algoritmos genéticos. A seção seguinte apresenta os trabalhos relacionados. A seção “Métodos” apresenta uma discussão detalhada da abordagem proposta. A seguir, é apresentado um exemplo que ilustra como a abordagem proposta é diferente do algoritmo padrão Apriori. Isso é seguido pelos resultados e discussões, que incluem uma discussão dos pontos de interesse que foram descobertos durante os testes. Por fim, são apresentadas as seções de conclusão e trabalhos futuros.

Streaming de dados em uma janela deslizante

Na maioria dos ambientes de dados modernos, alguns armazenamentos de dados estáticos, como um banco de dados, são assumidos, mas nem sempre é esse o caso. Os dados podem chegar a uma porta de computador de forma contínua e frequente. Além disso, a chegada desses dados é uma parte necessária de um modelo de negócios, bem como a situação dos navios que trazem carga para um porto de uma cidade. Se o porto tentasse armazenar toda a carga, o espaço poderia rapidamente se tornar um problema. Além disso, embora algumas cargas sejam normalmente armazenadas, elas devem ser processadas e removidas em média pelo menos tão rapidamente quanto a carga chega. Os sistemas de computador que lidam com a chegada persistente têm um problema semelhante em relação à chegada, espaço e processamento.

O processamento que precisa ser feito no streaming de dados geralmente envolve a extração de algumas informações úteis dos dados. Existem algumas restrições exclusivas a serem consideradas ao lidar com dados de streaming, por exemplo, os dados podem ser limitados a acesso único, os dados podem ser ilimitados em volume e a resposta em tempo real pode ser necessária para que os dados sejam úteis [7, 12]. , 13, 22]. Esses trabalhos discutem modelos de processamento de dados de streaming, sendo um deles o modelo de janela deslizante. A janela deslizante acompanha as transações atuais, permitindo que a transação mais antiga seja excluída quando a nova transação chegar. Portanto, as n transações mais recentes sempre aparecem na janela deslizante, onde n é o tamanho da janela deslizante.

A Figura 1 ilustra uma janela deslizante de tamanho 5. {Item B, Item D, Item E} não estão mais na janela. Esses dados foram excluídos. A área em caixa apresenta a janela atual.
As novas transações são as duas transações acima da janela. Eles ainda não foram incluídos na janela. Assim que {Item B, Item C} for incluído, {Item C, Item D} será removido.

Deriva do conceito

A mineração de informações de dados estáticos resulta em informações estáticas. No streaming de dados, as informações aprendidas não são estáticas. É dinâmico. Quando a natureza das informações nos dados muda ao longo do tempo, isso é chamado de desvio de conceito. Também pode ser importante capturar a deriva de um conceito. Neste artigo, os desvios de conceito são medidos em termos de conjuntos de itens frequentes, pela mudança em conjuntos de itens frequentes. Outros trabalhos também analisaram o uso frequente de conjuntos de itens para mineração de dados de streaming [7]. Gama [7] menciona que o problema mais difícil na mineração de conjuntos de itens frequentes de dados de streaming é que conjuntos de itens que eram infrequentes no passado podem agora se tornar frequentes, e conjuntos de itens que eram frequentes no passado podem se tornar infrequentes. Portanto, além da detecção de desvio de conceito, o gerenciamento de desvios de conceito também é um aspecto importante da mineração de dados de streaming.

Conjuntos de itens frequentes da janela deslizante

Uma regra de associação é uma instrução condicional que diz que, se o item A existe em uma transação, é provável que o item B esteja nessa transação. Na mineração de regras de associação, o suporte é definido como itens encontrados com alguma frequência mínima em todo o conjunto de dados. A confiança é definida como a frequência de um item em relação ao conjunto de itens que contém os itens suportados. Um conjunto de itens frequente é um conjunto de itens que atende ao limite mínimo de suporte. Te

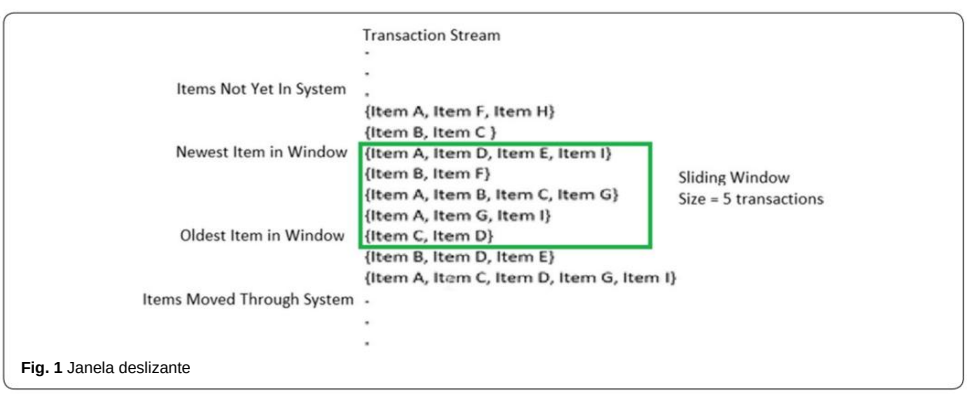

Fig. 1 Janela deslizante

O desafio é encontrar conjuntos de itens frequentes em janelas deslizantes de dados de streaming. Antes de apresentar as fórmulas que foram usadas para calcular as contagens de suporte em janelas deslizantes, é apresentado o histórico do algoritmo geral Apriori.

Dado:

- comprimento da janela deslizante = 20
- suporte mínimo=0,3
- confiança mínima=0,6.

E,

- 6 transações contêm (A), portanto, o suporte é 6/20.
- 8 transações contêm (B), portanto, o suporte é 8/20.

E, A e B aparecem juntos em uma transação 4 vezes, então:

- suporte (A **ÿ** B)=4/20 ou 0,2.
- confiança (A **ÿ** B)=P(B|A)=(conta_suporte(A **ÿ** B))/(conta_suporte(A))=4/6 ou 0,66.

Confiança (A **ÿ** B)>confiança mínima (de 0,6), então esta é uma regra de associação forte.

Portanto, para dados estáticos, um conjunto de itens frequente será qualquer conjunto de itens que tenha suporte > 0,3.

Para dados de streaming, uma janela deslizante é usada, portanto, o support_count modificado para determinar conjuntos de itens frequentes é:

support_count **=** tamanho da janela deslizante **ÿ** suporte mínimo

Portanto, o conjunto de itens frequente da janela deslizante é:

Suporte(A **ÿ** B) > tamanho da janela deslizante **ÿ** suporte mínimo

O algoritmo genético support_count será:

support_count **=** tamanho da janela deslizante **ÿ** suporte **ÿ** confiança

Assim, o conjunto de itens (A **ÿ** B) será considerado um conjunto de itens frequente do algoritmo genético quando:

Suporte (A **ÿ** B) > tamanho da janela deslizante **ÿ** suporte **ÿ** confiança

A métrica usada na função ftness deste algoritmo genético é o algoritmo genético limite de inclusão do conjunto de itens frequente do ritmo:

tamanho da janela deslizante **ÿ** suporte **ÿ** confiança

Algorítmos genéticos

O algoritmo genético imita a ação da seleção natural. Primeiro, a estrutura dos genes de um indivíduo é definida. Estes são retratados como uma lista de 0 s e 1 s. Te

os indivíduos são então avaliados quanto à adequação usando uma função de adequação. Dez os melhores indivíduos são cruzados, ou seja, alguns de seus genes são trocados. Em seguida, os indivíduos sofrem mutação, ou seja, um ou mais de seus genes podem ser alterados aleatoriamente de 1 para 0 ou vice-versa. Uma vez que os operadores genéticos tenham sido usados para definir uma nova população, a função ftness é aplicada novamente. Esse processo se repete até que um indivíduo bem-sucedido seja encontrado ou o número máximo de gerações seja encontrado.

Trabalhos relacionados

Regras de associação de mineração e conjuntos de itens frequentes têm um histórico bem estabelecido e, de fato , [15, 20, 22] discutem algoritmos para realizar essas tarefas no contexto de fluxo de dados. O modelo acumulativo, o modelo de janela deslizante e o modelo acumulativo ponderado são apresentados por Yu e Chi [22] como formas de lidar com dados de streaming.
O modelo cumulativo e os modelos cumulativos ponderados mantêm todos os dados em um armazenamento de dados. No entanto, no modelo cumulativo ponderado, os dados tornam-se menos importantes à medida que envelhecem. Em Krempl et ai. A discussão de [13] sobre os desafios enfrentados por esse tipo de pesquisa é a falta de um conjunto comumente aceito de ambientes, protocolos e métodos de benchmarking necessários para teste e comparação. No contexto de mineração de conjuntos de itens frequentes, Gama [7] afirma que a deriva de conceito é o problema mais difícil. Hoens et ai. [10] apresenta um método de detecção e adaptação à deriva de conceito com foco no aprendizado de classificação. Kim e Park [11] e Wang e Abraham [21] também consideram a detecção de desvio de conceito.

Bull [4] considera os algoritmos genéticos como um dos algoritmos de otimização mais úteis, mas somente se a função que está sendo otimizada for barata de calcular. Rabinovich e Wigderson [17], por outro lado, acreditam que podem ser usados em problemas práticos difíceis. Tem havido bastante pesquisa sobre a taxa de convergência de algoritmos genéticos. Forrest e Mitchell [5], Ruholla e Smith [19], Aldallal [2], Lin et al. [14], Angelova e Pencheva [3] e Pellerin et al. [16], todos consideram formas de melhorar a taxa de convergência em algoritmos genéticos. Esses estudos afetam a taxa de convergência manipulando os operadores genéticos. De acordo com Aldallal [2], a métrica de convergência mais comum em um algoritmo genético é o número de gerações necessárias para que um indivíduo atenda aos requisitos de aptidão. No entanto, He e Lin [9] consideram a “média geométrica normalizada da razão de redução da diferença de densidade por geração” como uma melhor métrica que ainda é facilmente calculável. Ghosh et ai. [8] e Rangaswamy e Shobha [18] discutem a mineração de conjuntos de itens frequentes e regras de associação com algoritmos genéticos.

Embora todas as peças individuais relacionadas a este trabalho tenham sido abordadas de forma independente, não há pesquisas que abordem diretamente todas essas questões em conjunto. Ou seja, não há nenhum trabalho abordando o problema de mineração de conjuntos de itens frequentes de dados de streaming com algoritmos genéticos na presença de desvio de conceito, problema abordado neste trabalho.

Métodos

Esta seção oferece uma descrição detalhada das operações do sistema proposto.

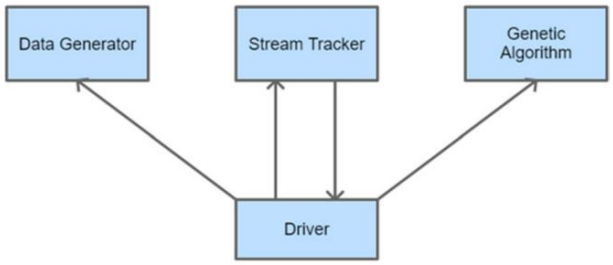

Fig. 2 Visão geral do sistema

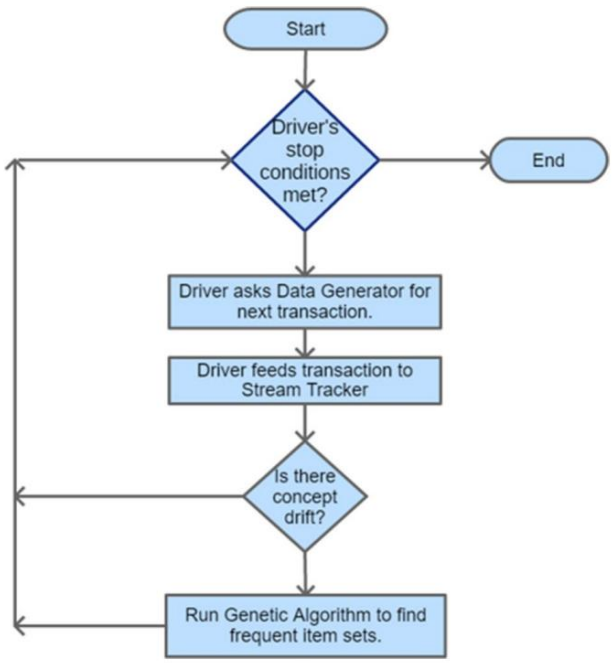

Fig. 3 Operações do sistema

Visão geral

Para estudar a dinâmica deste problema, foi desenvolvido um sistema que inclui um gerador de dados, um algoritmo genético [6], um rastreador de fluxo e um driver, conforme mostrado na Fig. 2.
O driver é o principal instrumento que define todas as variáveis necessárias. O driver solicita ao gerador de dados o próximo item, passa o item para o stream tracker, detecta a deriva do conceito e inicia o algoritmo genético quando necessário.

O fluxo de trabalho, apresentado na Fig. 3, pode ser apresentado da seguinte forma.

- O driver obtém uma transação do gerador de dados.
- O driver enfileira essa transação no stream tracker.
- O driver obtém uma referência ao stream tracker e inicializa o algoritmo genético com isso e o apoio e confiança necessários.
- Na inicialização, o algoritmo genético faz uma cópia da fila a ser usada para fur processamento durante as avaliações de integridade.

- Quando o driver detecta um desvio de conceito, ele redefine a fila no algoritmo genético antes de iniciar a função encontrar itens frequentes do algoritmo genético.

Detecção de desvio de conceito

No streaming de dados, conjuntos de itens frequentes para um conceito estável seriam identificados pelo conjunto de conjuntos de itens frequentes permanecendo constantes em número e conteúdo, apesar dos dados passarem pela janela. Em outras palavras, se o conjunto de conjuntos de itens mudar de alguma forma, será considerado como uma deriva de conceito ocorrendo.

O driver detecta o desvio de conceito mantendo uma cópia do conjunto de itens potenciais frequentes do rastreador de fluxo. Cada vez que um item é adicionado à fila, o novo conjunto de itens frequentes em potencial é comparado com o antigo e, se mudou, o conceito foi potencialmente alterado. Isso, juntamente com a verificação de que o conjunto de itens frequente tem pelo menos dois membros e que a janela deslizante está cheia, aciona o driver para iniciar o algoritmo genético.

Gerador de dados

O gerador de dados emite continuamente transações únicas quando chamado. É preciso um argumento, o número de itens que é necessário emitir, antes de mudar para um conceito diferente. Ele faz isso percorrendo e retornando elementos de um arquivo de texto de transações até que o número necessário de emissões seja atendido. Em seguida, ele obtém um arquivo diferente e faz o mesmo.

Modelo de janela deslizante: rastreador de fluxo

Em uma fila especializada, o rastreador de fluxo acompanha o número de transações especificadas como o tamanho da janela deslizante. Quando uma transação é enfileirada, se o tamanho necessário da janela for atendido, o item mais antigo será automaticamente retirado da fila. Quando uma transação é enfileirada, os itens individuais na transação são adicionados a um dicionário com as contagens na janela deslizante. Ao desenfileirar uma transação, as contagens de itens são diminuídas. Isso permite que o stream tracker retorne um conjunto de itens que podem ser considerados parte de um conjunto de itens frequentes com base em sua frequência na janela deslizante. Esse conjunto é o modelo para um indivíduo em potencial no algoritmo genético. Qualquer conjunto de itens no dicionário com uma contagem maior que o limite de inclusão do conjunto de itens frequentes do algoritmo genético (tamanho da janela deslizante*suporte*confiança) é o conjunto de conjuntos de itens frequentes em potencial. Se a contagem de um conjunto de itens frequente for menor que esse cálculo, então o item não pode fazer parte de um conjunto de itens frequente de algoritmo genético.

Algorítmos genéticos

Um algoritmo genético é um algoritmo que imita uma seleção natural. Conforme apresentado na Fig. 4, ele:

1. Gera uma população aleatória de indivíduos.
2. Avalia a aptidão de cada indivíduo usando uma função de aptidão.
3. Aplica uma operação de acasalamento com base em uma probabilidade de cruzamento e com mais ft indi indivíduos com maior chance de serem escolhidos para o acasalamento.
4. Isso cria uma nova população.

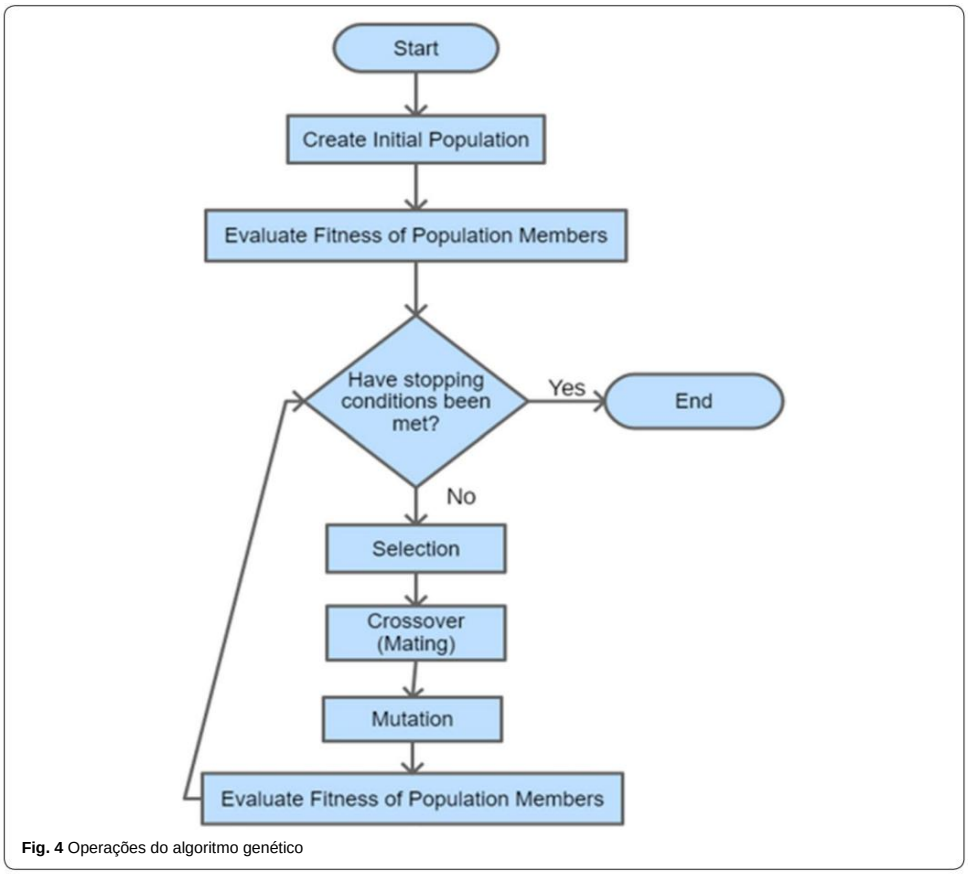

Fig. 4 Operações do algoritmo genético

5. A nova população é submetida a mutação com a chance de cada gene sofrer mutação sendo determinada pela probabilidade de mutação, geralmente uma chance muito baixa.
6. A aptidão da nova população é avaliada. Se a aptidão de um melhor indivíduo não atingiu algum nível desejado e o número máximo de gerações não foi alcançado, o processo recomeça na etapa 3.

Para implementar o algoritmo genético, foi utilizada a biblioteca open source, Distributed Evolutionary Algorithms in Python [6] . Essa estrutura permite ao usuário definir facilmente as partes de um algoritmo genético que são bastante comuns, como a definição de um indivíduo como uma lista de 0 s e 1 s com cada 0 ou 1 representando um único gene. Ao mesmo tempo, oferece a flexibilidade de definir os métodos e registrá-los no framework conforme necessário para definir o algoritmo genético necessário.

Indivíduos e genes

Nesta implementação, os indivíduos foram compostos por uma lista de 0s e 1s que mapeiam para a lista ordenada de possíveis itens frequentes do stream tracker, por exemplo:

Itens frequentes: [maçã, banana, doces, fraldas, sabonete]
Indivíduo aleatório: [0,1,1,0,1]

Fornece um conjunto de itens frequentes em potencial que consiste em {banana, candy, soap}

Tamanho da população

Para encontrar vários conjuntos de itens suficientes em vez de apenas um melhor conjunto de itens, é importante manter uma população de indivíduos únicos. Portanto, o tamanho da população foi baseado no comprimento do conjunto de itens potencial ou em algum tamanho máximo. A regra usada para o tamanho da população foi:

mínimo(2(comprimento de _genesÿ1), 100).

Isso manteve o tamanho da população baixo, garantindo indivíduos únicos em toda a população.

Função de condicionamento físico

Para determinar a ftness de um indivíduo, a função ftness obteria os itens que foram mapeados para os 1 s no indivíduo e, em seguida, percorreria um instantâneo da fila verificando se o conjunto de itens do indivíduo era um subconjunto de cada transação e conte isso. A função de ajuste fnal foi:

contagem de transações/tamanho da janela deslizante.

Seleção de torneio para acasalamento

A seleção do torneio para acasalamento envolve escolher aleatoriamente um pequeno número de indivíduos (neste caso, 3) e pegar aquele com a maior aptidão e colocá-lo na nova população. Isso foi feito sem remover os indivíduos originais da população antiga.

Cruzamento de dois pontos

Uma vez que os indivíduos foram escolhidos para o acasalamento, eles foram submetidos a uma operação de cruzamento de dois pontos. Isso envolveu encontrar uma seção intermediária aleatória de dois indivíduos e trocar as seções. Dois números aleatórios foram escolhidos, o primeiro maior que o índice 0 e depois desse índice para o penúltimo índice da lista. Uma vez que o meio seção foi encontrada, poderia ser trocada com a mesma seção do outro indivíduo (ver Fig. 5).

Mutação

A última operação genética que foi aplicada antes de reavaliar a aptidão de cada indivíduo foi o operador de mutação (ver Fig. 6). Normalmente, um número aleatório é gerado para cada bit no gene e se o número aleatório for menor que a mutação

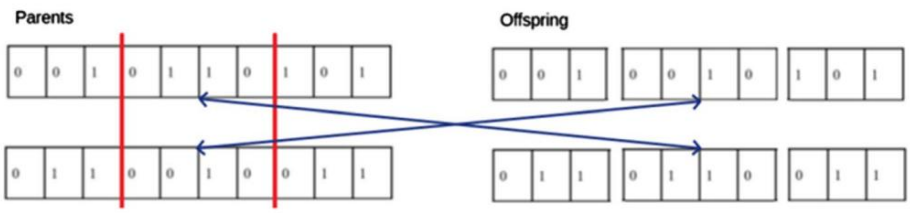

Fig. 5 Operação de cruzamento

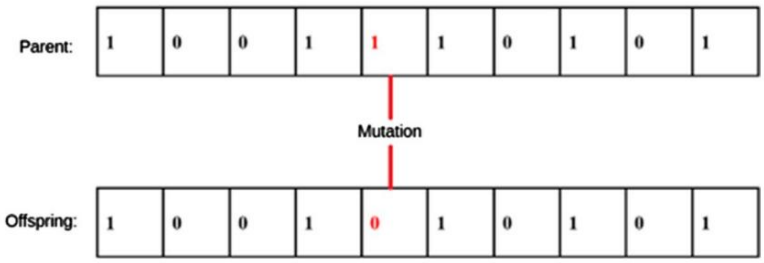

Fig. 6 Operação de mutação

```
Reset GA at: 100
--Parameters:  100 2000 0.25 0.5 50 0.5
-- Evolution concluded in --50 generations producing 5 frequent item sets
Reset GA at: 2023
--Parameters:  100 2000 0.25 0.5 50 0.5
-- Evolution concluded in --50 generations producing 2 frequent item sets
Reset GA at: 2025
--Parameters:  100 2000 0.25 0.5 50 0.5
-- Evolution concluded in --50 generations producing 3 frequent item sets
```

Fig. 7 Exemplo de tela

probabilidade, então o bit é desviado. A probabilidade de mutação era muitas vezes muito baixa, ou seja, abaixo de 5%. No entanto, devido aos requisitos de exclusividade, o operador de mutação foi usado para remover duplicatas da população. Cada novo indivíduo que estava sendo adicionado à população era convertido em uma string e se essa string já estivesse em um conjunto temporário de strings, o indivíduo era mutado novamente. Dados difíceis não foram mantidos sobre a taxa de mutação, a verificação local indicou que isso resultou em uma taxa incomumente alta de mutação.

Procedimento de avaliação

Testar esse aplicativo envolveu definir várias variáveis no driver e capturar o fluxo de saída e avaliar a saída resultante. Por exemplo, a Fig. 7 mostra os parâmetros neste ponto:

- Tamanho da janela deslizante: 100 transações
- Desvio de conceito a cada: 2.000 transações
- Suporte: 0,25
- Confiança: 0,5
- Número de gerações: 50
- Probabilidade de cruzamento: 0,5.

Quando a janela deslizante atingiu a capacidade de 100 transações, o algoritmo genético evoluiu por 50 gerações e conseguiu encontrar 5 conjuntos de itens frequentes. O conceito foi detectado como estável até a transação 2023, onde o algoritmo foi acionado novamente, mas apenas 2 conjuntos de itens foram encontrados. Na transação 2025, o algoritmo foi novamente acionado e 3 conjuntos de itens foram encontrados.

Um exemplo comparativo

Esta seção apresenta um exemplo do algoritmo Apriori trabalhando em uma janela deslizante e, em seguida, apresenta a geração de conjuntos de itens frequentes pelo algoritmo genético usando uma janela deslizante. Quando janelas deslizantes são usadas a priori, support_count é defnido como o tamanho da janela deslizante multiplicado pelo suporte mínimo; quando algoritmos genéticos são usados em cálculos frequentes de conjuntos de itens, o support_count de Apriori é modificado pela multiplicação da confiança pelo support_count de Apriori. Isso torna os conjuntos de itens frequentes do algoritmo genético mais inclusivos. Ou seja, mesmo que um conjunto de itens não fosse incluído como um conjunto de itens frequente usando a abordagem Apriori, ele seria incluído como um conjunto de itens frequente de algoritmo genético. Isso é ilustrado a seguir - primeiro é apresentada a geração de conjuntos de itens frequentes para o algoritmo Apriori com a janela deslizante e, em seguida, é apresentada a geração de conjuntos de itens frequentes do algoritmo genético com a janela deslizante.

Os conjuntos de itens frequentes Apriori com a janela deslizante

O algoritmo Apriori é o algoritmo padrão para mineração de conjuntos de itens frequentes. Os conjuntos de itens frequentes podem ser processados posteriormente para gerar regras de associação. Primeiro, uma definição de alguns termos em relação ao algoritmo Apriori: Uma transação é uma linha ou linha no conjunto de dados. Um conjunto de itens é qualquer subconjunto do conjunto de todos os itens individuais. K-itemset é um conjunto de itens que contém k itens. O suporte é a contagem do número de transações nas quais um determinado conjunto de itens pode ser encontrado. O suporte também é representado como a porcentagem de transações nas quais um determinado conjunto de itens pode ser encontrado. A confiança é usada ao determinar as regras de associação. Confiança é a probabilidade de um subconjunto de transações ocorrer em um determinado conjunto de itens. O suporte e a confiança são definidos pelas condições do problema e podem ser configurados conforme necessário. Um conjunto de itens frequente é qualquer conjunto de itens em que a contagem de suporte está acima de algum valor.

Dado um banco de dados de transações:

{Item 2, Item 4, Item 6}
{Item 2, Item 4, Item 1, Item 5}
{Item 6, Item 4, Item 1, Item 3}
{Item 2, Item 6, Item 4, Item 1}
{Item 2, Item 6, Item 4, Item 3}

A priori segue o seguinte padrão:

- Gere todos os conjuntos de itens 1 candidatos e calcule seu suporte.
- Elimine quaisquer conjuntos de 1 itens que não atendam aos critérios mínimos de suporte.
- Combine os 1-itemsets restantes em todos os 2-itemsets candidatos possíveis e cal
  cular o seu apoio.
- Elimine quaisquer conjuntos de 2 itens que não atendam aos critérios mínimos de suporte.
- Combine os 2-itemsets restantes em todos os 3-itemsets candidatos possíveis e cal
  cular o seu apoio.
- Elimine quaisquer conjuntos de 3 itens que não atendam aos critérios mínimos de suporte.

• Isso é repetido até que nenhum novo conjunto de itens “candidato” possa ser criado.

Dado:

• Suporte mínimo>0,5
• Support_count=2.5.

O support_count para o algoritmo Apriori usando uma janela deslizante de 5 é calculado
Como:

Support_count **=** tamanho da janela deslizante **ÿ** suporte mínimo

qual é

5 **ÿ** 0,5 **=** 2,5.

Portanto, o support_count é 2,5.

A Tabela 1 apresenta o padrão geral de Apriori.

Da Tabela 1: Primeiro, os conjuntos de 1 itens são contados. Os itens 3 e 5 não possuem o mini
mum support_count, portanto, são eliminados. Conjuntos de 2 itens são criados a partir dos conjuntos de 1 itens restantes. Os conjuntos {Item 1, Item 2} e {Item 1, Item 6} não atingiram o mínimo necessário support_count e, portanto, são eliminados. Em seguida, os 3 conjuntos de itens são gerados. Existem 4 conjuntos de 3 itens possíveis e todos são eliminados, exceto {Item 2, Item 4, Item 6}. Não há conjuntos de 4 itens possíveis. Isso deixa um total de 9 conjuntos de itens frequentes que atendem ao suporte critério.

O algoritmo genético freqüenta conjuntos de itens com a janela deslizante

Quando algoritmos genéticos são usados para criar conjuntos de itens frequentes, o primeiro passo é procurar conjuntos de 2 itens ou mais. O critério support_count também é ajustado, conforme discutido anteriormente. Isso é ilustrado usando o conjunto de itens {Item 1, Item 6} do exemplo a priori acima.

Dado:

• Suporte mínimo=0,5
• Confiança mínima=0,6.

Para uma regra: Item 1 **ÿ** Item 6

Tabela 1 Exemplo a priori

1-Conjuntos de itens	Contar	2-Conjuntos de itens	Contar	3-Conjuntos de itens	Contar
Item 1	3	~~Item 1, Item 2~~	~~2~~	~~Item 1, Item 2, Item 4~~	~~2~~
Item 2	4	Item 1, Item 4	3	~~Item 1, Item 2, Item 6~~	~~1~~
~~Item 3~~	~~2~~	~~Item 1, Item 6~~	~~2~~	~~Item 1, Item 4, Item 6~~	~~2~~
Item 4	5	Item 2, Item 4	4	Item 2, Item 4, Item 6	3
~~Item 5~~	~~1~~	Item 2, Item 6	3		
Item 6	4	Item 4, Item 6	4		

Este conjunto de itens {Item 1, Item 6} não atendeu aos critérios para um conjunto de itens frequente no algoritmo Apriori regular. Mas, o item 1 está em 3 transações que satisfaz o suporte necessário. E, das 3 transações em que o Item 1 aparece, o Item 6 aparece em 2 delas.
Ou seja, esta regra tem uma confiança de 2/3 ou 0,66, que excede o mínimo declarado de 0,6, portanto, esta é uma regra de associação válida.

A implementação do algoritmo genético proposta usa o limite de inclusão do conjunto de itens frequente do algoritmo genético (tamanho da janela deslizante*suporte*confiança) como o suporte_
contagem necessária para inclusão como um conjunto de itens frequente válido. Nesse caso, o support_count é:

$$5 \times 0{,}5 \times 0{,}6 = 1{,}5.$$

Portanto, o conjunto de itens {Item 1, Item 6} seria um conjunto de itens frequente do algoritmo genético. O limite de inclusão de conjuntos de itens frequentes do algoritmo genético é configurado dessa maneira para poder incluir conjuntos de itens que contêm o consequente e o antecedente de uma regra de associação potencial como parte do conjunto de itens frequentes.

À medida que as transações chegam, é mantida uma contagem de quantas transações cada indivíduo aparece, e aquelas que não atendem ao limite mínimo de contagem de suporte são eliminadas. Neste exemplo, o item 5 é eliminado de um possível mapeamento no genoma.
Dado que existem cinco itens elegíveis neste conjunto de dados, o primeiro passo é começar criando uma população aleatória de 2(5ÿ1) ou 16 indivíduos, cada um com 5 genes e cada gene mapeando para um dos itens possíveis.

Nota: Nem todos os indivíduos estão incluídos no exemplo a seguir apresentado na Tabela 2.

[Item 1, Item 2, Item 3, Item 4, Item 5, ~~Item 6~~]

Tabela 2 Exemplo de algoritmo genético

Mapas individuais para		A aptidão é verificada	Selecione para acasalar	Novo Indivíduos	Mudar Novo Indivíduos	Mapas para	Ginástica
[01101]	[Item 2, Artigo 3, Item 6]	1	Indivíduos 2, 3	**[01001]** Criança 1	[01001]	[Item 2, Item 6]	3
[00011]	[Item 4, Item 6]	4	Indivíduos 2, 3	**[10010]** Criança 2	[10010] Nenhum gene mutado	[Item 1, Item 4]	3
[11000]	[Item 1, Item 2]	2	Indivíduos 2, 4	[00011] Sem cruzamento. Probabilidade de cruzamento não excedida Indivíduo 2 levado para a próxima geração	[00011]	[Item 4, Item 6]	4
[01010]	[Item 2, Item 4]	4	Indivíduos 2, 4	[01010] Sem cruzamento. Probabilidade de cruzamento não excedida Indivíduo 4 levado para a próxima geração	[01010] Nenhum gene mutado	[Item 2, Item 4]	4

Fig. 8 Gerações máximas

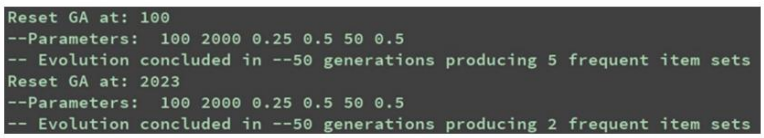

Fig. 9 Detecção de desvio de conceito

O processo acima é repetido até que alguma condição de parada seja atendida. Essa condição de parada inclui um número máximo de gerações ou alguma condição de integridade. Apenas a geração de conjuntos de itens frequentes é mostrada, e não as regras de associação que se seguiriam.

Resultados e discussões

Os resultados dos testes foram apresentados para: (i) gerações máximas; (ii) detecção de desvio de conceito; (iii) tamanhos variados de janelas deslizantes; e, por fim, (iv) alguns exemplos de operações ideais.

Gerações máximas

A Figura 8 apresenta os resultados das gerações máximas. Os parâmetros listados na Fig. 8 são (em ordem):

- Tamanho da janela deslizante: 100 transações
- Desvio de conceito a cada: 2.000 transações
- Suporte: 0,25
- Confiança: 0,5
- Número máximo de gerações: 50
- Probabilidade de cruzamento: 0,5.

Detecção de desvio de conceito

A detecção de desvio de conceito envolvia acompanhar uma lista de itens que ocorreram mais vezes do que a contagem mínima de suporte. Essas contagens foram mantidas em um dicionário e atualizadas para cada nova transação, e então o dicionário foi repetido para criar uma nova 'lista de conjuntos de itens possíveis'. A nova lista foi comparada com a lista antiga e, se fossem diferentes, considerava-se que o conceito estava à deriva. Essa abordagem exigia uma atualização do dicionário para cada item em uma nova transação, além de um loop por todos os itens do dicionário para montar a nova lista. Para esses testes, esse método de detecção de desvio de conceito acionou conforme o esperado e teve um bom desempenho, especialmente quando um tamanho de janela maior foi usado. A Figura 9 apresenta a detecção de desvio de conceito.

A partir da Fig. 9, um conceito estável na transação número 100 pode ser detectado. Nenhum trabalho é necessário até a transação 2023, quando um novo conceito é detectado pela primeira vez. Neste ponto o

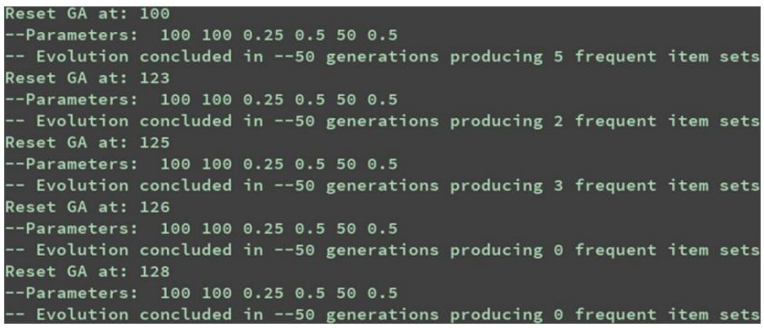

Fig. 10 Ilustração de pequena deriva de conceito, ilustração 1

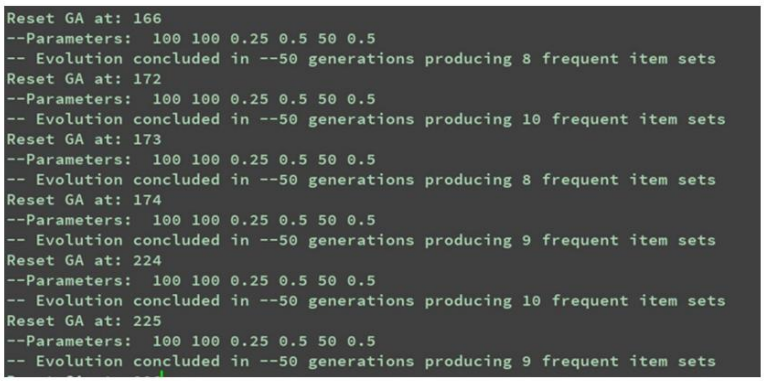

Fig. 11 Ilustração de pequena deriva de conceito, ilustração 2

todo o segundo arquivo de dados (novo conceito) é carregado na janela deslizante, enquanto 77% da janela deslizante ainda contém o conceito antigo.

Este teste destacou que a proporção entre o tamanho da janela e as transações por desvio era a chave para um bom desempenho. Isso ocorre porque, se um tamanho de janela aleatório fixo for usado, à medida que as transações por desvio aumentam, os dados da perspectiva da janela deslizante começam a parecer estáticos. Um exemplo disso pode ser visto na Fig. 9 , que mostra um desvio de conceito de 2.000 transações.

Na Fig. 10, tanto a janela deslizante quanto as transações por desvio são definidas como 100. Como esperado, com os novos dados preenchendo a janela imediatamente após o preenchimento com o primeiro conjunto de dados, não há estabilidade. A Figura 11 mostra a próxima mudança de dados e a instabilidade no número de conjuntos de itens encontrados durante essas execuções repetidas. Além disso, nesta execução, o algoritmo genético foi chamado para ser executado cerca de 60 vezes durante 400 transações com apenas 27 execuções produzindo qualquer conjunto de itens.

Além disso, havia problemas para obter resultados quando o tamanho da janela deslizante era muito pequeno. Quando a contagem de suporte era muito pequena, ou seja, 0, 1, 2, havia problemas devido a flutuações normais nos dados que pareciam ser um desvio de conceito.

Variando os tamanhos das janelas deslizantes

Na Fig. 12 , o tamanho da janela é 10 e o desvio de conceito é 100. Observando uma região de gatilhos de desvio de conceito entre as transações 72 e 95, o conceito deve se tornar estável.

```
Reset GA at: 62
--Parameters:  10 100 0.25 0.5 50 0.5
-- Evolution concluded in --50 generations producing 6 frequent item sets
Reset GA at: 72
--Parameters:  10 100 0.25 0.5 50 0.5
-- Evolution concluded in --50 generations producing 4 frequent item sets
Reset GA at: 75
--Parameters:  10 100 0.25 0.5 50 0.5
-- Evolution concluded in --50 generations producing 1 frequent item sets
Reset GA at: 78
--Parameters:  10 100 0.25 0.5 50 0.5
-- Evolution concluded in --50 generations producing 1 frequent item sets
Reset GA at: 82
--Parameters:  10 100 0.25 0.5 50 0.5
-- Evolution concluded in --50 generations producing 7 frequent item sets
Reset GA at: 92
--Parameters:  10 100 0.25 0.5 50 0.5
-- Evolution concluded in --50 generations producing 5 frequent item sets
Reset GA at: 95
--Parameters:  10 100 0.25 0.5 50 0.5
-- Evolution concluded in --50 generations producing 0 frequent item sets
```

Fig. 12 Tamanho da janela deslizante pequena, ilustração 1

```
Reset GA at: 110
--Parameters:  10 100 0.25 0.5 50 0.5
-- Evolution concluded in --50 generations producing 30 frequent item sets
Reset GA at: 111
--Parameters:  10 100 0.25 0.5 50 0.5
-- Evolution concluded in --50 generations producing 43 frequent item sets
Reset GA at: 112
--Parameters:  10 100 0.25 0.5 50 0.5
-- Evolution concluded in --50 generations producing 27 frequent item sets
```

Fig. 13 Tamanho da janela deslizante pequena, ilustração 2

A contagem frequente de suporte do conjunto de itens para isso seria 10*0,25*0,5=1,25. Assim, quase todos os itens podem fazer parte de um conjunto de itens frequente, e um grande número de conjuntos de itens frequentes pode ser gerado, como pode ser visto na Figura 13. Esses conjuntos não têm valor informativo. Isso mostra que, usando essa técnica, o tamanho da janela deve ser gerenciado em conjunto com os valores de suporte e confiança para obter resultados razoáveis.

Exemplos de operações ideais

Sob operações ideais, enquanto o algoritmo genético estava sendo repetidamente acionado durante uma alta taxa de desvio de conceito, o número de conjuntos de itens frequentes válidos gerados cairia para zero e, então, à medida que os novos dados começassem a preencher a janela, a contagem de conjuntos de itens aumentaria. ao número de conjuntos de itens frequentes esperados para o novo conjunto de dados. O algoritmo genético não seria acionado novamente até que o detector de conceito começasse a ser acionado. Além disso, embora esse algoritmo gerasse um conjunto muito bom de regras, às vezes ele perderia uma, como seria esperado pelo processo aleatório de um algoritmo genético.

A Figura 14 apresenta um conceito estável na transação 100 com 5 conjuntos detectados. Esta situação é estável até a transação 1023, onde o conceito está começando a derivar e as 5 regras diminuem para 0 regras pela transação 1026. Da transação 1026 a 1063 o algoritmo genético é acionado 11 vezes sem nenhum conjunto frequente detectado. A Figura 15 apresenta o algoritmo sendo acionado e encontrando 7, 8, 10, 8 e finalmente 9 conjuntos de itens frequentes.

O conceito é estável até a transação 2024. Observe que o conceito estável de 9 conjuntos de itens na transação 1074 é menor do que os 10 conjuntos encontrados apenas 2 transações anteriores. Desde o

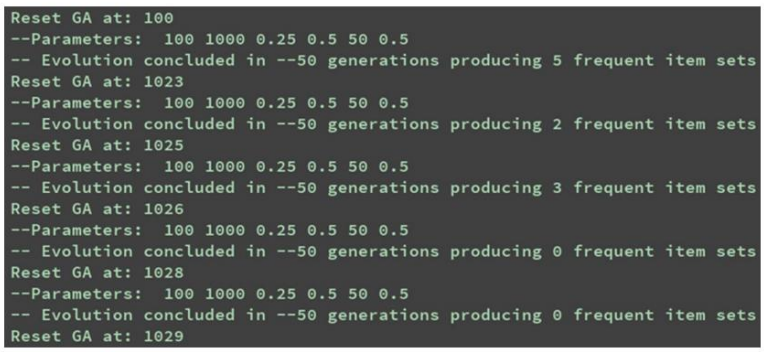

Fig. 14 Ilustração de operação ideal 1

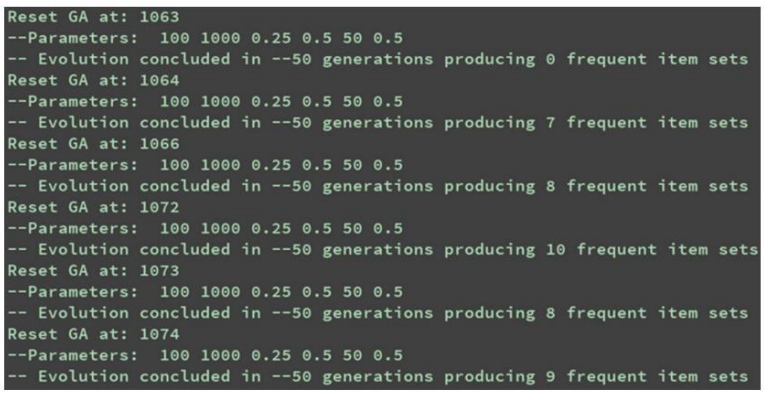

Fig. 15 Ilustração de operação ideal 2

janela está se enchendo de dados para este conceito, geralmente, um número crescente de conjuntos de itens seria esperado. No entanto, os algoritmos genéticos não devem necessariamente obter todos os conjuntos de itens o tempo todo.

Alterar a função ftness para permitir a convergência antecipada seria uma grande melhoria. Uma maneira simples de fazer isso seria permitir que um pequeno número de regras relativas ao total de regras possíveis esperadas constituam convergência.

Discussão

Alguns problemas levaram à mitigação de alguns dos controles esperados do algoritmo genético. Inicialmente o algoritmo estava convergindo rapidamente em 1 ou 2 conjuntos de itens. Esse é o comportamento normal esperado para um algoritmo genético que está procurando o 'melhor' indivíduo. No entanto, o objetivo era encontrar o maior número possível de conjuntos de itens frequentes. Essa convergência nos máximos locais deveu-se ao fato de a população se tornar dominada por indivíduos com exatamente o mesmo conjunto de itens genéticos. Isso levou a limitar o tamanho da população e alterar a função de mutação para garantir uma população de indivíduos únicos. Isso resultou na perda da capacidade de controlar esses dois fatores do algoritmo genético. Uma vez que o requisito de parada foi baseado no indivíduo mais fraco atendendo ao requisito de aptidão ou o número máximo de gerações sendo atendido, o resultado foi que o número máximo de gerações sempre foi atendido. Isso implica que as variáveis de

o algoritmo genético não era tão livre para ser manipulado. A primeira conclusão está relacionada a essa questão de diversidade na população genética para esse tipo de problema. A natureza desse problema exigia encontrar todas ou quase todas as soluções, porém, como o algoritmo genético tendia a otimizar para uma determinada solução, a composição da população foi alterada. Para resolver esse problema, todos os indivíduos da população deveriam ser únicos. O efeito desta decisão foi duplo. Primeiro, isso limitou o tamanho da população a um máximo de 2 (número de itens possíveisÿ1), garantindo a possibilidade de uma população de indivíduos únicos, mas tirando o controle do tamanho da população. Em segundo lugar, para garantir a singularidade na população, a operação de mutação foi rastreada e, se um novo indivíduo fosse uma duplicata, esse indivíduo sofreria uma mutação. Isso implicava que a probabilidade de mutação não refletia mais a taxa de mutação real, nem era um mecanismo de controle válido. Restava apenas a probabilidade de cruzamento, que parecia ter pouco efeito sobre o funcionamento do algoritmo genético. O resultado mais preocupante desse desenvolvimento foi que o algoritmo genético nunca convergiu antes que o número máximo de gerações fosse concluído. Mesmo na melhor circunstância, quando a janela foi inicialmente preenchida com cinco cópias do mesmo arquivo de dados e nenhuma deriva de conceito, o algoritmo genético estava terminando no número máximo de gerações.

Embora o algoritmo tenha oferecido boas regras dentro de certos parâmetros, o teste foi feito em um ambiente muito controlado. Em condições de produção o desempenho pode ser muito diferente. Existem várias coisas que podem ser feitas para ajudar, por exemplo, a função ftness pode ser paralelizada, ou, cada vez que o algoritmo genético é iniciado, todo o processo pode ser iniciado em seu próprio thread. Se os valores de conceito não foram alterados, pode ser possível determinar conjuntos de itens válidos na transmissão ao vivo. Isso evitaria a necessidade de o método de instantâneo ser usado agora, mas adicionaria a complexidade de encerrar um encadeamento como 'falhou' se o conceito flutuasse enquanto estava em execução. Também pesquisar maneiras de ajudar o algoritmo genético a encontrar múltiplas regras enquanto ainda alcança convergência sem a interrupção artificial das 'gerações máximas' levaria a melhores dados sobre esse assunto.

Conclusão

Este artigo apresenta um estudo de mineração de conjuntos de itens frequentes de dados de streaming. Uma abordagem usando algoritmos genéticos é apresentada e várias relações entre a taxa de deriva do fluxo de dados (concept drift), o tamanho da janela deslizante e as restrições do algoritmo genético foram exploradas. Fórmulas úteis são apresentadas para calcular contagens mínimas de suporte para determinar conjuntos de itens frequentes em dados de streaming usando janelas deslizantes. Além disso, detectar desvios de conceito envolvia acompanhar uma lista de itens que ocorreram mais vezes do que a contagem mínima de suporte.

Os testes destacaram que a proporção entre o tamanho da janela e as transações por desvio era a chave para um bom desempenho. Mas, obter bons resultados é difícil quando o tamanho da janela deslizante era muito pequeno, porque então as flutuações normais nos dados pareciam ser um desvio de conceito. Portanto, o tamanho da janela deve ser gerenciado em conjunto com os valores de suporte e confiança para alcançar resultados razoáveis. Este método de detecção de desvio de conceito teve um bom desempenho quando tamanhos de janela maiores foram usados.

Usando a configuração descrita, as variáveis importantes do algoritmo genético foram finalmente decididas pelos dados em tempo de execução. Taxa de mutação, tamanho individual e população

tamanho estavam todos além do controle externo. O cruzamento foi especificado, mas não teve efeito na velocidade de convergência para o algoritmo genético, pois o algoritmo genético sempre levava o número máximo de gerações e ainda alcançava bons resultados.

Trabalho futuro

Existem áreas promissoras para trabalhos futuros. Primeiro, o algoritmo genético pode ser encadeado, permitindo que o fluxo continue enquanto os conjuntos de itens frequentes estão sendo gerados. Em segundo lugar, o algoritmo genético poderia ser alterado para permitir a convergência em um pequeno número de regras válidas. Isso permitiria a convergência antes do número máximo de gerações. Terceiro, melhorias e possível paralelização da função ftness devem ser consideradas. Quarto, o teste em dados ao vivo pode levar a insights que não são considerados devido à natureza de alguns dados de teste específicos. Além disso, armazenar em buffer os dados de entrada além da janela deslizante, dentro de limites, pode permitir mais tempo para o algoritmo genético
convergem.

Reconhecimentos

Este trabalho foi parcialmente apoiado pelo Instituto Askew da University of West Florida.

Contribuições dos autores

Este trabalho foi principalmente conceituado por SB e PS. A programação foi feita principalmente pelo PS. Ambos os autores participaram da redação do artigo. Ambos os autores leram e aprovaram o manuscrito final.

Informações dos autores

Dr. Sikha Bagui é Professor e Askew Fellow no Departamento de Ciência da Computação da The University West Florida, Pensacola, Flórida. O Dr. Bagui é ativo na publicação de artigos de periódicos revisados por pares nas áreas de design de banco de dados, mineração de dados, BigData, reconhecimento de padrões e computação estatística. Dr. Bagui trabalhou em projetos de pesquisa financiados e não financiados e tem inúmeras publicações revisadas por pares. Ela também é coautora de vários livros sobre banco de dados e SQL. Bagui também atua como Editor Associado e faz parte do conselho editorial de vários periódicos.

Patrick Stanley é mestre em ciências pela The University West Florida e atualmente trabalha como Full Stack Software Engineer na University of North Florida.

Financiamento

Este trabalho não foi financiado.

Disponibilidade de dados e materiais

Não aplicável.

Interesses competitivos

Nenhum dos autores tem interesses conflitantes.

Recebido: 12 de março de 2020 Aceito: 17 de julho de 2020
Published online: 25 July 2020

Referências

1. Agrawal R, Imielinski T, Swami A. Regras de associação de mineração entre conjuntos de itens em grandes bancos de dados. In: Proceedings of the ACM SIGMOD international conference on management of data, Washington, DC, USA. 1993. pp. 207-216.
2. Aldalal AS. Evitando a convergência prematura de GA em sistemas de recuperação de informação. Int J Intell Syst Appl Eng. 2015;2(4):80. https://doi.org/10.18201/ijisae.78975.
3. Angelova M, Pencheva T. Ajustando os parâmetros GA para melhorar o tempo de convergência. Int J Chem Eng. 2011. https://doi.org/10.1155/2011/646917.
4. Touro AD. Taxas de convergência de algoritmos de otimização global eficientes. J Mach Aprenda Res. 2011;12(88):2879–904.
5. Forrest S, Mitchell M. O que torna um problema difícil para um AG? Alguns resultados anômalos e sua explicação. Gás Mach Aprenda. 1993. https://doi.org/10.1007/978-1-4615-2740-4_6.
6. Fortin FA, De Rainville FM, Gardner MA, Parizeau M, Gagné C. DEAP: algoritmos evolutivos facilitados. J Mach Aprenda Res. 2012;13:2171–5.
7. Gama J. Uma pesquisa sobre aprendizado de fluxos de dados: tendências atuais e futuras. Prog Arti Intell. 2012;1(1):45–55. https://doi.org/10.1007/s13748-011-0002-6.
8. Ghosh S, Biswas S, Sarkar D, Sarkar PP. Mineração de conjuntos de itens frequentes usando GA. Int J Artif Intell Appl. 2010;1(4):133–43. https://doi.org/10.5121/ijaia.2010.1411.
9. He J, Lin G. Taxa média de convergência de algoritmos evolutivos. Computação Trans Evol IEEE. 2016;20(2):316–21. https://doi.org/10.1109/tevc.2015.2444793.

10. Hoens TR, Polikar R, Chawla NV. Aprendendo com o streaming de dados com desvio e desequilíbrio de conceito: uma visão geral. Prog Artif Intell. 2012;1(1):89–101. https://doi.org/10.1007/s13748-011-0008-0.
11. Kim Y, Park CH. Um método de detecção de desvio de conceito eficiente para streaming de dados sob rotulagem limitada. IEICE Trans Inf Syst. 2017;100(10):2537–46. https://doi.org/10.1587/transinf.2017edp7091.
12. Kolajo T, Daramola O, Adebiyi A. Análise de fluxo de big data: uma revisão sistemática da literatura. J Grandes Dados. 2019; 6:47. https: // doi.org/10.1186/s40537-019-0210-7.
13. Krempl G, Zliobaite I, Brzezinski D, Hullermeier E, Last M, Lemaire V, Noack T, Shaker A, Sievi S, Spiliopoulou M, Stefanowski J. Desafios abertos para pesquisa de mineração de fluxo de dados. ACM SIGKDD Explorar Newsl. 2014;16(1):1–10. https ://doi.org/10.1145/2674026.2674028.
14. Lin WY, Lee WY, Hong TP. Adaptação das taxas de cruzamento e mutação em AGs. J Inf Sci Eng. 2003;19(5):889–903.
15. Nedunchezhian R, Geethanindhini K. Associação de mineração de regras em Big Data – uma pesquisa. Int J Eng Res Technol. 2016;5(5):42–6.
16. Pellerin E, Pigeon L, Delisle S. Parâmetros auto-adaptativos em algoritmos genéticos. In: Proceeings SPIE 5433. Data mining e descoberta de conhecimento: teoria, ferramentas e tecnologia VI. 2004. https://doi.org/10.1117/12.542156.
17. Rabinovich Y, Wigderson A. Técnicas para limitar a taxa de convergência de GAs. Algoritmos de Estrutura Aleatória. 1999;14(2):111–38. https://doi.org/10.1002/(sici)1098-2418(199903)14:2%3c111:aid-rsa1%3e3.0.co;2-6.
18. Rangaswamy S, Shobha G. Mineração de regras de associação otimizada usando GA. J Comput Sci Eng Inf Technol Res. 2012;2(1):1–9.
19. Ruholla JM, Smith BK. Fluido GA (FGA). J Comput Des Eng. 2017; 4 (2): 158–67. https://doi.org/10.1016/j. jcde.2017.03.001.
20. Vijayarani S, Sathya P. Um algoritmo eficiente para mineração de itens frequentes em fluxos de dados. Int J Innov Res Comput Com mun Eng. 2013;1(3):742–7.
21. Wang H, Abraham Z. Detecção de desvio de conceito para transmissão de dados. In: 2015 International Joint Conference on Neural redes (IJCNN). 2015. https://doi.org/10.1109/ijcnn.2015.7280398.
22. Yu PS, Chi Y. Associação de mineração de regras em córregos. Enciclopédia de Sistemas de Banco de Dados. 2009. https://doi.org/10.1007/ springerreference_63188.

Nota do editor