CENTRO UNIVERSITÁRIO UNIFECAF

GRADUAÇÃO EAD

DATA STRUCTURE STRATEGY AND IMPLEMENTATION

PROFESSOR ROBINSON LEMOS BEZERRA

TUTOR PROFESSOR MARCEL TEIXEIRA

Implementação de um Sistema de Recomendação de Produtos Utilizando Grafos

William Gama Kerpen

RA 2903

Preetz

2025

Implementação de um Sistema de Recomendação de Produtos Utilizando Grafos

Este trabalho faz parte da avaliação da disciplina de Redes de Computadores, do curso de graduação em Analise e Desenvolvimento de Sistemas, ministrado pelo Centro Universitário UniFecaf. Sob orientação do Professor Conteudista Afonso Brandão e Tutor Professor Fernando Leonid.

William Gama Kerpen

RA 2903

Preetz

2025

SUMÁRIO

1. Introdução...............................................................................................................................1

2. Objetivos.................................................................................................................................2

3. Apresentação do Projeto.........................................................................................................3

3.1. Apresentação do Projeto em Diagrama de rede.......................................................3

3.2. Apresentação do Projeto em Vídeo..........................................................................4

4. Conclusão................................................................................................................................5

**1.Introdução**

A crescente demanda por eficiência em plataformas de e-commerce tem evidenciado a importância de sistemas de busca rápidos e precisos. No caso passado a MegaStore, empresa fictícia de grande porte, enfrenta desafios significativos relacionados à lentidão e imprecisão de seu sistema de busca, comprometendo a experiência do usuário e a conversão de vendas.

Na vida real empresas como Mercado Livre possui mais de 60 Milhões de Ofertas, a Amazon 100 milhões de produtos, Shopee e Magazine Luiza não divulgam a quantidade de produtos, mas são milhões de vendedores que resultam em números tão grandes quanto a de seus concorrentes.

Com uma quantidade tão grande de dados é essencial para as empresas buscarem por algoritmos que possibilitem o rapidez na busca para seus usuários.

Este trabalho tem como objetivo explorar as causas do problema, os impactos sobre os usuários, propor soluções viáveis e discutir a implementação de estruturas de dados eficientes em Rust como alternativa de otimização para as buscas.

**2. Causas da Lentidão e Imprecisão**

As principais causas da lentidão e imprecisão do sistema de busca da MegaStore estão relacionadas ao uso de estruturas de dados inadequadas, algoritmos de busca não otimizados e à ausência de indexação eficiente. Sistemas que operam com listas lineares ou bancos de dados relacionais sem mecanismos de cache ou pré-processamento tendem a apresentar alta latência e baixa relevância nos resultados (MANNINO; RAHMAN, 2015). Além disso, a falta de personalização e de algoritmos de recomendação contribui para a imprecisão das buscas.

**3. Impacto nos Usuários**

Os principais afetados pela baixa performance do sistema de busca são os consumidores e os administradores da plataforma. Os consumidores enfrentam frustrações ao não encontrar produtos desejados, o que pode levá-los a abandonar o site e buscar alternativas em concorrentes. Já os administradores observam queda nas taxas de conversão, aumento no custo de aquisição de clientes e perda de competitividade (LAUDON; TRAVER, 2021).

**4. Soluções Possíveis**

Entre as soluções possíveis para otimizar o sistema de busca da MegaStore, destacam-se:

* **Implementação de tabelas hash**: permite acesso rápido a dados indexados por chave.
  + *Prós*: alta velocidade de busca; baixo consumo de memória.
  + *Contras*: complexidade na resolução de colisões; dificuldade em buscas por aproximação.
* **Uso de grafos para recomendação**: modela relações entre produtos e usuários.
  + *Prós*: permite sugestões personalizadas; melhora a relevância dos resultados.
  + *Contras*: maior complexidade computacional; necessidade de algoritmos especializados.
* **Tries (árvores de prefixo)**: otimizam buscas por prefixo, como autocompletar.
  + *Prós*: excelente desempenho em buscas textuais; útil para grandes volumes de dados.
  + *Contras*: alto consumo de memória; difícil manutenção.

**4. Barreiras à Implementação**

A implementação dessas soluções enfrenta barreiras como a necessidade de reestruturação do backend, capacitação da equipe técnica em Rust, e integração com sistemas legados. Além disso, a migração de dados e a adaptação dos algoritmos existentes exigem planejamento cuidadoso e testes extensivos (SEDGEWICK; WAYNE, 2011).

**5. Contribuição das Tabelas Hash em Rust**

Rust é uma linguagem de programação moderna que oferece segurança de memória e alto desempenho. A utilização de tabelas hash em Rust pode reduzir significativamente o tempo de busca por produtos, graças à eficiência da estrutura HashMap, que permite acesso constante (O(1)) a elementos indexados (KLABNIK; NICHOLS, 2018). Além disso, Rust evita erros comuns como vazamentos de memória e condições de corrida, tornando o sistema mais robusto.

**6. Requisitos de Hardware e Software**

Para implementar a solução proposta, são necessários servidores com boa capacidade de processamento paralelo (multi-core), memória RAM suficiente para armazenar as estruturas de dados em tempo real e SSDs para acesso rápido ao banco de dados. No aspecto de software, é necessário o compilador Rust, bibliotecas como serde para serialização de dados, e ferramentas de integração contínua como GitHub Actions ou Jenkins.

**7. Integração com a Plataforma**

A solução pode ser integrada à plataforma de e-commerce da MegaStore por meio de APIs RESTful escritas em Rust, que se comunicam com o frontend e o banco de dados. Essa abordagem modular permite que o novo sistema de busca funcione como um microserviço, facilitando a escalabilidade e a manutenção (NEWMAN, 2015).

**8. Indicadores de Desempenho**

Os principais indicadores de desempenho para avaliar a eficácia da solução incluem:

* Tempo médio de resposta das buscas.
* Taxa de conversão de usuários que utilizam a busca.
* Precisão e recall dos resultados apresentados.
* Uso de CPU e memória durante picos de acesso.
* Tempo de uptime e número de falhas.

**9. Implicações a Longo Prazo**

Caso o problema do sistema de busca não seja resolvido, a MegaStore poderá enfrentar queda progressiva na satisfação dos clientes, aumento da taxa de abandono do carrinho, e perda de market share. Além disso, a reputação da marca pode ser prejudicada, dificultando parcerias comerciais e expansão internacional (KOTLER; KELLER, 2012).

**10. Sistema de Buscas**

Para o caso fictício criamos uma pagina de buscas simulando a pagina inicial de um site de compras online, para isso foi utilizado a linguagem Python com o framework Flask, foi utilizado também JavaScript, HTML e CSS. O banco de dados foi carregado com 10 mil produtos fictícios com a biblioteca Faker e o uso de IA para criação de listas com nomes reais.

1. **Referências**

KLABNIK, S.; NICHOLS, C. *The Rust Programming Language*. No Starch Press, 2018.

KOTLER, P.; KELLER, K. L. *Administração de Marketing*. 14. ed. São Paulo: Pearson, 2012.

LAUDON, K. C.; TRAVER, C. G. *E-commerce 2021: business, technology, society*. Pearson, 2021.

MANNINO, M.; RAHMAN, M. *Database Design, Application Development, and Administration*. McGraw-Hill, 2015.

NEWMAN, S. *Building Microservices*. O'Reilly Media, 2015.

SEDGEWICK, R.; WAYNE, K. *Algorithms*. 4. ed. Addison-Wesley, 2011.