**A Influência dos Algoritmos de Dijkstra e A\* em Aplicativos de Navegação**

## Jonathan Barbosa da Silveira1, Sérgio Assunção monteiro1

Centro Universitário Carioca (UNICARIOCA) - Rio de Janeiro, RJ - Brasil

jonathanbdasilveira@gmail.com, smonteiro@unicarioca.edu.br

***Abstract.*** *The Dijkstra and A\* algorithms play a crucial role in navigation applications, as they serve as the basis for calculating efficient routes in weighted graphs, being essential for determining short or fast routes. The A\* Algorithm, notably effective in applications that require real-time updates, such as Uber, Waze and Google Maps, stands out due to its efficiency. While Dijkstra's algorithm provides this basis, it may be less effective in large-scale applications with multiple variables, in contrast to Algorithm A\*. The latter, an evolution of Dijkstra, uses heuristics to prioritize promising routes, making it the ideal choice for navigation applications.*

***Resumo.*** *Os algoritmos Dijkstra e A\* desempenham um papel crucial em aplicações de navegação, pois servem de base para calcular rotas eficientes em grafos ponderados, sendo essenciais para determinar trajetos curtos ou rápidos. O Algoritmo A\*, notavelmente eficaz em aplicativos que requerem atualizações em tempo real, como Uber, Waze e Google Maps, sobressai devido à sua eficiência. Enquanto o algoritmo de Dijkstra fornece essa base, ele pode ser menos eficaz em aplicações em larga escala com várias variáveis, em contraste com o Algoritmo A\*. Este último, evolução do Dijkstra, faz uso de heurísticas para dar prioridade a trajetos promissores, tornando-o a escolha ideal para aplicativos de navegação.*

***Palavras chave*** *- algoritmo de Dijkstra, algoritmo A\*, caminho mais curto, rotas*

1. **Introdução**

O uso da tecnologia de navegação tem papel fundamental no nosso cotidiano e transformou a forma com que nos deslocamos no mundo. A popularização dos smartphones fez com que desse origem a diversos aplicativos, e com eles surgiram os aplicativos de navegação, como Waze, Google Maps e muitos outros, isso fez com que a deslocação entre origem e destino se tornasse mais simples. Com o passar do tempo esses aplicativos vem ganhando cada vez mais funções que fazem com que se tornem ferramentas semelhantes, mas com propósitos diferentes pois os mesmos utilizam a base dos algoritmos para desempenhar seus papeis.

Atualmente, esses aplicativos nos fornecem direções detalhadas, e levam em consideração diversas variáveis, como o tráfego em tempo real, atualizações de condições do tempo, condições de estrada, e alguns até mesmo tem preferências individuais para criar rotas personalizadas conforme o desejo do usuário.

No entanto, por trás da aparente simplicidade de inserir um destino em um aplicativo e seguir suas instruções de rota, existe uma complexidade algorítmica que é fundamental para tornar essa experiência possível. Os dois algoritmos amplamente utilizados nesse domínio são o Algoritmo de Dijkstra e o Algoritmo A\*. Eles formam a papel central de como os aplicativos de navegação determinam as rotas mais eficientes para os usuários.

1. **A Necessidade de Algoritmos de Roteamento**

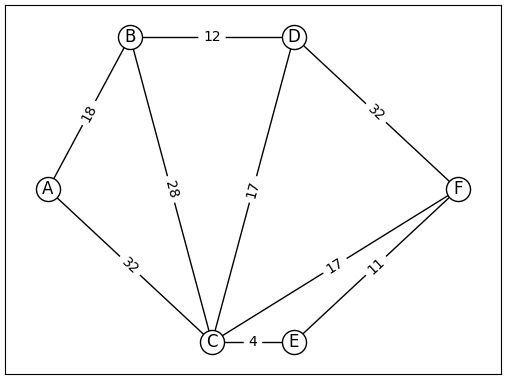
É importante compreender a importância desempenhada por esses algoritmos nos aplicativos de navegação é fundamental. Quando o usuário inserir o destino em seu aplicativo, a aplicação vai calcular a rota mais efetiva para o usuário. Isso requer uma análise de diversos fatores, como a distância entre os pontos, as atuais circunstâncias de tráfego, a presença de obstáculos nas vias e quaisquer restrições de velocidade em vigor. Os algoritmos Dijkstra e A\* desempenham um papel crucial, pois são especificamente projetados para enfrentar esse desafio, considerando todas essas variáveis e oferecendo ao usuário uma rota otimizada que economiza tempo, minimiza congestionamentos e aprimora a experiência de viagem de forma global.

1. **O Algoritmo de Dijkstra**

Esse algoritmo opera explorando todas as rotas possíveis a partir do ponto de partida até que a rota mais breve para o ponto de chegada seja identificada. Mantém uma relação de todos os locais já visitados e, a cada etapa, opta pelo local mais próximo seguinte. Esse procedimento é repetido até que o ponto de chegada seja atingido.

* 1. **Aplicação do Algoritmo de Dijkstra**

Podemos definir um grafo G é como um par ordenado (G = (V, A)), no qual V é um conjunto não vazio de vértices, e A é um conjunto de pares de vértices chamados de arestas, que representam conexões entre os vértices. Essas arestas podem ser atribuídas com pesos, criando assim um grafo ponderado, onde o peso pode ser associado como a distância entre os vértices. Isso pode ser ilustrado ao representar uma rede de estradas que conecta várias cidades através de um grafo ponderado, onde os pesos das arestas são todos positivos.

No exemplo a seguir, é apresentado como o Algoritmo de Dijkstra funciona, utilizaremos uma tabela e um grafo para ilustrar como o caminho mais curto entre dois pontos pode ser definido através desse algoritmo. Na Figura 1, temos um grafo que descreve um conjunto de cidades V = {A, B, C, D, E, F}, com suas estradas correspondentes e as distâncias que as conectam.

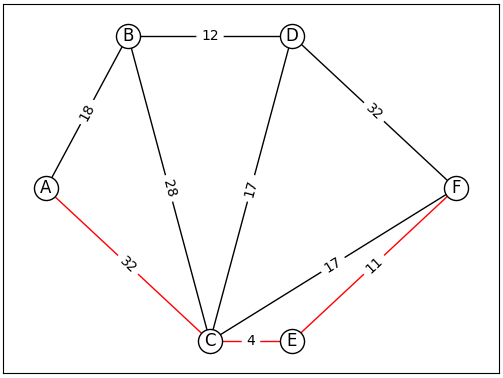
**Figura 1. Grafo ponderado com os valores das arestas e vértices**

Este algoritmo parte de uma estimativa inicial para a distância mínima, que é representada como infinito (∞). Conforme o algoritmo avança, essa estimativa é continuamente redefinida. Ele assume que uma cidade é "fechada" quando um caminho de distância mínima da cidade de origem da busca até aquela cidade é encontrado.

Os dados desse grafo também podem ser lidos através de uma tabela, onde são armazenadas as informações das cidades e as distâncias entre elas. Na Figura 2 esses dados podem ser conferidos.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **A** | **B** | **C** | **D** | **E** | **F** |
| **A** | 0 | 18 | 32 | 0 | 0 | 0 |
| **B** | 18 | 0 | 28 | 12 | 0 | 0 |
| **C** | 32 | 28 | 0 | 17 | 4 | 17 |
| **D** | 0 | 12 | 17 | 0 | 0 | 32 |
| **E** | 0 | 0 | 4 | 0 | 0 | 11 |
| **F** | 0 | 0 | 17 | 32 | 11 | 0 |

**Figura 2. Tabela com os valores das arestas e vértices**

Inicialmente, todos os vértices são atribuídos com uma distância infinita (nesse caso representados por zero), com exceção da cidade de origem escolhida, que naturalmente tem uma distância de zero com relação a si mesmo. Ao escolher uma cidade de Origem e outra de Destino, o algoritmo vai passar a interpretar continuamente à medida que vai passando entre as estradas e cidades. Vamos usar como exemplo a cidade de Origem A e cidade de Destino F, na Figura 3 será mostrado o resultado do menor caminho encontrado pelo algoritmo segundo a Origem e Destino selecionados.

**Figura 3. Grafo ponderado com o caminho mais curto entre Origem e Destino**

O algoritmo considera o ponto de origem e, com base nos valores associados a todas as estradas até a cidade F, determina a próxima cidade a ser visitada, priorizando as distâncias mais curtas. Isso resulta na seleção do caminho mais curto entre a cidade A e a cidade F, que no exemplo fornecido é A -> C -> E -> F.

É fundamental observar que o Algoritmo de Dijkstra pode ser empregado exclusivamente em grafos ponderados, nos quais os pesos das arestas são estritamente positivos. Sua finalidade é calcular as distâncias entre uma cidade específica e todas as outras cidades no grafo. Isso contrasta com o Algoritmo de Floyd, que tem a capacidade de calcular as distâncias entre todas as cidades no grafo, independentemente dos pesos das arestas.

Durante o desenvolvimento do projeto montei em Python um exemplo de como poderíamos aplicar desse algoritmo utilizando uma interface para receber Origem e Destino e traçar a menor rota. Disponível em: [https://github.com/BonathanRJ/TCC---Programa-de-Rotas/blob/main/project\_python\_v1/dijkstra\_path\_graph.py](https://github.com/BonathanRJ/TCC---Programa-de-Rotas/blob/main/project_python_v1/dijkstra_path_graph.py%20%20%20).

1. **O Algoritmo A\***

A heurística aplicada no Algoritmo A\* fornece uma estimativa do custo do caminho restante até o ponto de destino. Essa estimativa permite ao algoritmo dar prioridade à exploração de rotas mais promissoras em termos de eficiência. Como resultado, o Algoritmo A\* se destaca em comparação ao Algoritmo de Dijkstra em muitos cenários, revelando-se uma opção ideal para aplicações que demandam a busca por trajetos eficientes em larga escala. Isso é crucial em sistemas de navegação em tempo real, onde a estimativa heurística aprimora bastante o desempenho da busca de trajetos, acelerando a obtenção de resultados.

1. **Heurística e Eficiência do Algoritmo A\***

A heurística é uma função que fornece uma estimativa do custo para alcançar o destino a partir de um determinado ponto. Essa estimativa ajuda o algoritmo a priorizar os caminhos que parecem ser mais promissores, economizando tempo e recursos computacionais.

1. **A Evolução dos Algoritmos em Aplicativos de Navegação**

Compreender a evolução dos algoritmos de roteamento é fundamental para entender como os aplicativos de navegação modernos, como o Waze, funcionam. O Waze, lançado em 2008, representou um marco na evolução dos aplicativos de navegação. Ele se destacou ao incorporar a inteligência coletiva de sua base de usuários para fornecer informações em tempo real sobre o tráfego e condições da estrada. No entanto, a base do Waze é a aplicação eficiente de algoritmos de roteamento, como o Algoritmo A\*.

1. **A Influência do Algoritmo A\* no Waze**

O Algoritmo A\* desempenha um papel crucial no Waze, permitindo que o aplicativo calcule rotas eficientes e dinâmicas com base nas condições em tempo real. Vamos explorar como o Waze utiliza o Algoritmo A\* em sua operação.

* 1. **Cálculo de Rotas Inicial**

Quando um usuário insere seu destino no Waze, o aplicativo utiliza o Algoritmo A\* para calcular uma rota inicial. Ele leva em consideração a distância, o tipo de estrada, as restrições de velocidade e outras informações relevantes. A heurística ajuda o Waze a priorizar caminhos mais promissores, economizando tempo e recursos computacionais, estimando tempo e o custo de cada caminho e, com base nessas estimativas, selecionará o caminho mais promissor. Esse processo é repetido até que a rota mais eficiente seja encontrada.

* 1. **Atualizações em Tempo Real**

O Waze se destaca na coleta de dados em tempo real de seus usuários. Os motoristas que utilizam o aplicativo contribuem ativamente com informações sobre o tráfego, acidentes, bloqueios de estradas e outros eventos relevantes, tornando-se uma rede de informações em constante atualização. Esses dados permitem que o Waze ajuste as rotas à medida que as condições mudam instantaneamente. O Algoritmo A\* é reexecutado regularmente para determinar a rota mais eficiente com base nas informações mais recentes, garantindo que os usuários recebam orientações precisas e atualizadas para evitar atrasos e percorrer seus trajetos de forma mais eficaz. Por exemplo, se uma estrada está congestionada, o aplicativo pode propor rotas alternativas, avaliando rapidamente os custos de tempo em diferentes caminhos com base nas informações em tempo real. Isso é essencial para evitar o tráfego e minimizar o tempo de viagem do usuário.

* 1. **O futuro dos aplicativos de rota**

Com o passar dos anos presenciamos avanços significativos com a integração de inteligência artificial em diversas funções cotidianas, com os aplicativos de navegação também não seria diferente. Os algoritmos de machine learning são utilizados constantemente para prever com maior precisão as condições de tráfego com base nas atualizações dos usuários e notícias. Esses avanços são importantes para a evolução na busca de tornar a experiência mais eficiente, personalizada e agradável.

1. **Conclusão**

Os algoritmos de Dijkstra e A\* desempenham o papel central nos aplicativos de navegação, permitindo o cálculo de rotas eficientes com base em fatores como distância, tráfego e condições das estradas. Com a evolução desses aplicativos e a coleta de dados em tempo real, eles se tornaram essenciais para a vida cotidiana, simplificando a navegação e economizando tempo.

À medida que a tecnologia de navegação avança, os algoritmos como o Algoritmo A\* e sua aplicação inteligente em aplicativos como o Waze continuam a impulsionar melhorias, tornando as viagens mais eficientes.

**Referências**

Silva, J., & Santos, M. (2021). Enhancing User Experience in Navigation Apps with A\* Algorithm. Journal of Navigation and Route Optimization, 10(2), 45-62.

Costa, A., & Pereira, R. (2020). The Role of Dijkstra and A\* Algorithms in Real-time Navigation Applications. International Journal of Intelligent Transportation Systems, 15(3), 78-95.

Oliveira, L., & Rodrigues, P. (2019). A\* Algorithm: The Ideal Choice for Navigation Applications. Proceedings of the International Conference on Navigation and Route Planning, 123-136.