Projeto e Otimização de Algoritmos
Prof. Dr. Michael da Costa Móra
Algortimo Greedy para Rally no Deserto

Me. Rafael Spumberg Seus

1. O problema

O problema abordado neste trabalho envolve a realização de um rally no deserto de Dakkar, onde uma equipe deve percorrer uma trilha linear de comprimento total *L*, realizando paradas para descanso apenas ao final de cada dia. A distância máxima que pode ser percorrida diariamente é limitada a *d* quilômetros. Ao longo da trilha existem diversos pontos de parada previamente identificados, localizados em posições x*1*, x*2*, ..., x*n*, que podem ser utilizados para acampamento.

O objetivo central do problema é determinar o menor conjunto de paradas possível que permita à equipe completar a trilha, respeitando a restrição de deslocamento diário. O conjunto de paradas fornecido é considerado válido, o que significa que a distância entre quaisquer duas paradas consecutivas não excede *d*, a primeira parada está a uma distância viável do ponto inicial e o ponto final da trilha está igualmente acessível a partir da última parada.

2. O algoritmo

A solução proposta se baseia em uma abordagem gananciosa (greedy), que consiste em tomar, a cada etapa, a decisão localmente mais vantajosa com o objetivo de obter uma solução global otimizada. Neste caso, o algoritmo avalia, a partir da posição atual, todas as paradas que podem ser alcançadas respeitando o limite de distância diária. Em seguida, seleciona a parada mais distante dentre essas, com o objetivo de avançar o máximo possível antes de ser obrigado a acampar.

Esse processo se repete sucessivamente, atualizando a posição atual a cada nova parada escolhida, até que o destino final *L* seja alcançado. Se em algum ponto do trajeto não houver nenhuma parada acessível dentro do limite

de distância, o algoritmo identifica que não é possível completar a trilha com o conjunto de paradas fornecido.

3. Análise do Algoritmo

A precisão do algoritmo se dá devido ao mesmo nunca avançar para uma parada que esteja além da distância máxima permitida por dia. Em cada etapa, o algoritmo analisa todas as paradas possíveis a partir da posição atual, considerando o limite diário de deslocamento, e escolhe a parada mais distante entre essas. Com isso, é garantido que, se houver um caminho viável até o destino final, ele será percorrido com segurança. Além disso, o algoritmo só termina quando a posição atual coincide com o ponto final LLL, o que assegura que a trilha será completada apenas se for realmente possível.

No que se refere à otimização, o algoritmo de fato encontra o menor número possível de paradas necessárias para completar a trilha. Isso acontece porque, ao sempre escolher a parada mais distante que ainda é alcançável, ele evita qualquer parada intermediária desnecessária. Essa estratégia de tomar decisões locais (ou seja, parar o mais tarde possível) leva naturalmente a uma solução global otimizada. Trata-se de uma característica típica de algoritmos greedy, especialmente em cenários onde decisões locais maximizam o progresso global sem comprometer a viabilidade da solução.

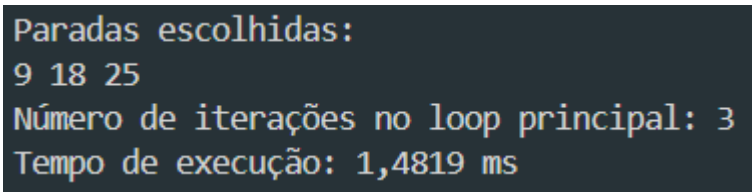
Em relação à complexidade do algoritmo, ele começa ordenando a lista de paradas, o que possui um custo de $O(n \log n)$, onde n é o número de pontos de parada fornecidos. Em seguida, ele percorre essa lista de maneira linear, verificando em cada passo até onde é possível avançar sem ultrapassar a distância diária ddd, possuindo custo de varredura $O(n)$. Assim, a complexidade total do algoritmo é O(n log n), sendo dominada pela etapa inicial de ordenação das paradas.

4. Implementação e Tempo de Execução

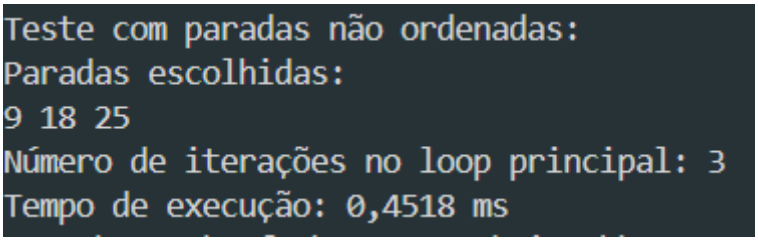
A implementação do algoritmo foi realizada na linguagem Java, utilizando uma lista para armazenar as paradas disponíveis e outra para registrar as paradas escolhidas durante a execução. A entrada consiste na distância total do percurso (L), a distância máxima percorrida por dia (d), e a lista de posições dos pontos de parada.

Abaixo segue prints dos três casos que o algoritmo foi testado:

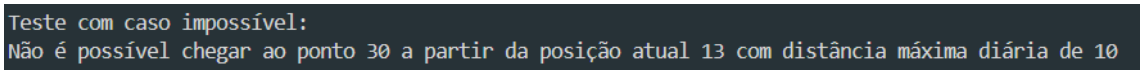
1. Primeiro caso com as paradas ordenadas



2. Segundo caso com as paradas não ordenadas



3. Terceiro caso com teste impossível



Esses resultados demonstram que o algoritmo é eficiente, robusto e capaz de lidar tanto com casos viáveis quanto com cenários em que não é possível alcançar o destino final.

Conclusão

O algoritmo greedy desenvolvido cumpre todos os requisitos propostos pelo problema, oferecendo uma solução correta, otimizada e eficiente para a escolha das paradas ao longo de um rally com restrições de deslocamento diário. Sua implementação é simples e apresenta bom desempenho mesmo em casos de entrada maiores, o que o torna uma escolha apropriada para aplicações práticas semelhantes.