Logística Urbana para Entrega de Mercadorias

Desenho de Algoritmos - DA

Grupo 80

Alexandre Costa (up202005319)
Ana Beatriz Fontão (up202003574)
Ana Rita Oliveira (up202004155)

U.PORTO
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

Descrição do problema

Neste projeto foi-nos proposto gerir a entrega de uma lista de encomendas de modo a maximizar ou minimizar certos pontos especificados.

Para as encomendas, criámos uma classe (DeliveryPackage) caracterizada por peso, volume, custo, duração e um id, único e diferente para todas as encomendas. Para as carrinhas, criamos a classe DeliveryVan constituída por peso máximo, volume máximo, custo e um id.

Foi-nos também fornecido um dataset com 50 instâncias de carrinhas e 450 instâncias de encomendas.

Os três problemas principais apresentados vão ser explorados em maior detalhe a seguir.

1º Cenário - Formalização

O primeiro cenário deste problema propõe uma distribuição de encomendas de modo a minimizar o número de estafetas necessário.

Conforme o enunciado, sabemos que os estafetas estão sempre disponíveis e que estes só realizam uma viagem por dia.

Logo, consideramos que uma carrinha não pode ser utilizada duas vezes no mesmo dia e, sendo assim, cada carrinha é apenas considerada uma única vez.

Além disso, sabemos que caso todas as carrinhas já estejam preenchidas e ainda existam encomendas por distribuir estas retornam ao fornecedor.

1º Cenário - Algoritmos Relevantes

Para a resolução deste problema, criámos dois algoritmos que nos deram resultados diferentes.

O **primeiro algoritmo**, baseado no problema do ***bin packing***, organiza as encomendas de forma crescente e as carrinhas de forma decrescente de modo a conseguir entregar o máximo número de encomendas no menor número de carrinhas possível.

No **segundo algoritmo**, à medida que as encomendas são distribuídas por diferentes carrinhas, escolhemos ter em conta a diferença entre volume e peso.

Isto é, se o volume restante numa carrinha for maior que o peso restante, optamos por colocar uma encomenda que tenha o menor peso, de modo a maximizar o número de encomendas por carrinha e vice versa.

1º Cenário - Análise de Complexidade

Para a primeira resolução, a função principal é a *distributepackages*, onde distribuímos as encomendas pelas várias carrinhas, tem complexidade O(P*V) em que P é o número de encomendas e V é o número de carrinhas.

Na segunda resolução, existem duas funções principais: a minimizeVans e a place_package. A primeira tem complexidade temporal $O(n^2 log(n))$ e a segunda tem de $O(n^2)$.

Todas estas funções têm complexidade espacial constante - O(1).

1º Cenário - Resultados da Avaliação Empírica

A **primeira solução** proposta é um **algoritmo greedy** que pretende otimizar o número de carrinhas a ser utilizadas. Apesar de a sua complexidade temporal ser melhor em relação à segunda proposta, o resultado obtido (**24 carrinhas**) é mais elevado. Isto iria afetar os resultados de datasets com dimensões superiores.

Na **segunda resolução** obtemos **22 carrinhas** necessárias para a entrega. Isto deveu-se ao facto de, à medida que percorremos os dados fornecidos, adaptamos a distribuição conforme o peso e volume restantes em cada carrinhas. Desta forma, obtemos um valor muito mais ajustado às dimensões de cada carrinha.

Concluindo, para as dimensões relativamente pequenas do dataset apresentado, pensamos que a melhor solução será a segunda resolução apresentada.

2º Cenário - Formalização

O segundo cenário deste problema propõe uma distribuição de encomendas de forma a **maximizar o lucro** da empresa.

Neste cenário tivemos também em conta as restrições do primeiro cenário, acrescentando apenas o facto de os estafetas cobrarem uma determinada quantia pelas entregas desse dia.

Para este problema será necessário então ter em conta:

- O **volume, peso** e **custo** que cada **estafeta/carrinha** terá.
- O **volume**, **peso** e **recompensa** que cada **encomenda** tem.

Deste modo, consideramos o lucro a **diferença entre a recompensa total e o custo total**.

2º Cenário - Algoritmos Relevantes

Para este cenário nós utilizamos o segundo algoritmo descrito para o cenário 1, mas com algumas modificações, nomeadamente:

- Em vez de ordenar as carrinhas de forma crescente em relação ao peso e ao volume, ordenados por ordem crescente de custo
- Adicionamos uma restrição que limita encomendas cuja recompensa seja demasiado baixa.

2º Cenário - Análise de Complexidade

Como foi referido anteriormente, o algoritmo utilizado para resolver este cenário é semelhante ao segundo algoritmo utilizado no cenário 1, pelo que as complexidades são também elas semelhantes.

Apesar das modificações feitas no algoritmo, a complexidade das funções utilizadas não foi alterada, pelo que a complexidade temporal das funções maximizeProfit e placePackage é O(n^2 log(n)) e O(n^2), respetivamente.

A complexidade espacial de todas as funções é O(1).

2º Cenário - Resultados da Avaliação Empírica

Na fase de desenvolvimento deste cenário foram testadas várias opções:

1. Inicialmente corremos o 2º algoritmo utilizado no cenário 1 sem nenhuma das alterações anteriormente referidas. Isto resultou num **lucro de 162 225€**.
2. Depois, após correr alguns testes, aplicamos uma função que exclui encomendas cuja recompensa não seja suficientemente alta para o peso e volume da encomenda. Isto resultou num **lucro de 198 598€**.
3. Por fim, mudamos a forma como damos sort às carrinhas, ordenando-as por custo crescente e enchendo-as da com menos custo para a com maior custo. Isto resultou num **lucro final de 205 677€**.

Desta forma, utilizamos um algoritmo greedy com estas alterações de forma a tentar chegar a uma aproximação da solução.

3º Cenário - Formalização

O terceiro cenário deste problema propõe uma distribuição de encomendas expresso de modo a **minimizar o tempo médio das entregas num dia**.

Conforme o enunciado, sabemos que está a ser utilizada uma única viatura que faz cada pedido de cada vez.

Sabemos também que as entregas expressos só podem ser entregues durante o horário comercial, das 9:00 às 17:00, o que equivale a 8 horas.

Para calcular este tempo médio, considerou-se que o tempo de cada entrega é o tempo que ela demora a chegar ao destino desde o momento que a carrinha sai da origem no início do dia. Soma-se, então, o tempo de cada entrega e divide-se pelo número total de entregas completas para obter a média.

3º Cenário - Algoritmos Relevantes

Um algoritmo relevante para este cenário é o std::sort(). Usa-se o std::sort() em conjunção com a função auxiliar sortByTime() para ordenar o vetor deentregas por ordem crescente de tempo de delivery.

Foi também aplicado um **algoritmo de escalonamento ganancioso** que percorre o vetor de deliveries ordenado, adicionando todos as deliveries à carrinha enquanto a soma total de tempo das deliveries não fosse superior ao tempo disponibilizado para as express deliveries.

3º Cenário - Análise de Complexidade

Devido à simplicidade deste algoritmo, a complexidade temporal da função que implementa o algoritmo é a mesma que a da função std::sort(), utilizada na ordenação das encomendas, isto é, a complexidade temporal é O(n log(n)).

A complexidade espacial de todas as funções é O(1).

3º Cenário - Resultados da Avaliação Empírica

A solução proposta é um algoritmo greedy que pretende ordenar o vetor de entregas por ordem crescente de tempo de entrega.

Uma vez que o tempo médio é calculado somando o tempo total de entrega (desde o início do dia até chegar ao destino) de cada entrega, entregas com o menor tempo de entrega minimizam efetivamente o tempo médio das entregas.

Desta forma, o tempo médio foi minimizado para 231.065 segundos, isto é, cerca de **3 minutos e 51 segundos** por encomenda, sendo que seriam entreges 124 das 450 encomendas.

Destaque de algoritmo

Na nossa opinião, todas as nossas funcionalidades estão bem implementadas sendo que nenhuma delas se destaca particularmente.

Os **algoritmos greedy** procuram, durante a sua execução, encontrar a melhor solução possível para cada caso apresentado.

Visto que maioritariamente as funcionalidades a serem implementadas tinham como objetivo a otimização (quer minimização como maximização) de certos fatores, optámos pela implementação destes algoritmos.

Principais Dificuldades

1. A organização dos dados tendo em conta as características das carrinhas e encomendas
2. Os algoritmos mais adequados a usar em cada situação
3. Testar o código

Avaliação do Grupo

Todos os elementos do grupo trabalharam de forma equivalente no projeto.

Alexandre Costa - 33 %
Ana Beatriz Fontão - 33 %
Ana Rita Oliveira - 33 %