Relatório Técnico sobre o Trabalho Prático, problemas intratáveis

Henrique Almeida, Gabriel Dolabela, João Pedro, Lucas Lima

Pontifícia Universidade Católica de Minas Gerais

1. Introdução

Este relatório descreve as implementações e análises de diferentes algoritmos para a distribuição de rotas igualitariamente entre caminhões.

2. Sobre o problema

O problema consiste em distribuir rotas entre caminhões de forma que a quilometragem percorrida por cada caminhão seja a mais próxima possível.

O problema é NP-completo, pois as rotas podem ou não ser alocadas para um caminhão. Também é possível verificar que o problema é NP-completo através da redução do problema para soma de subconjuntos.

O problema se mostra semelhante ao problema da soma de subconjuntos, onde é necessário encontrar um subconjunto de números que some um valor desejado. Neste caso, o valor desejado é a média de quilometragem das rotas pelo número de caminhões, todos as rotas devem ser distribuídas.

3. Sobre a implementação

3.1. Backtracking

A estratégia de backtracking foi implementada de forma recursiva, onde a cada chamada, um novo nó é adicionado a solução, e a partir dele, novas chamadas são feitas, até que não haja mais nós a serem adicionados.

Sobre a poda, o código verifica se a adição do nó ultrapassa o valor máximo esperado para a solução. Caso ultrapasse, a chamada recursiva é interrompida e o nó não é adicionado a solução.

O valor máximo é calculado com base na média de quilometragem das rotas pelo número de caminhões. Caso haja uma divisão equitativa, o seu resultado passa a ser a quilometragem máxima esperada para cada caminhão. Caso contrário, o resultado é a quilometragem máxima esperada para cada caminhão mais 10%.

O caso base ocorre quando todas as rotas foram adicionada a solução temporária. Nesse caso, ela é comparada com a solução final, e esta é copiada para a solução final caso seja melhor.

3.2. Guloso

3.2.1. Guloso com Ordem das Rotas

A implementação da estratégia gulosa baseada na ordem das rotas segue uma abordagem simples. As rotas são ordenadas em ordem crescente de quilometragem e, em seguida, distribuídas sequencialmente para os caminhões. A ordem de distribuição é invertida a cada ciclo, garantindo uma distribuição mais equitativa do que sem a inversão.

3.2.2. Guloso com Quilometragem Acumulada

A segunda estratégia gulosa utiliza a quilometragem acumulada como critério para a distribuição de rotas. As rotas são distribuídas para os caminhões com base na menor quilometragem acumulada.

É utilizada uma fila de prioridade para a implementação dessa estratégia. A cada iteração, o caminhão com menor quilometragem acumulada é retirado da fila e a rota é distribuída para ele. Em seguida, o caminhão é inserido novamente na fila, com a nova rota alocada a ele.

A classe interna Caminhao auxilia o uso da fila de prioridade implementando a interface Comparable, que permite a comparação entre os caminhões com base na quilometragem acumulada pela fila.

3.3. Divisão e Conquista

Na Divisão e Conquista, o algoritmo verifica se é possível incluir ou não a rota para manter a distribuição equitativa.

Esta verificação é feita através de um valor máximo e mínimo passados como parâmetro. Caso a inclusão da rota não ultrapasse o valor máximo, o item é adicionado a solução e a próxima chamada recursiva é feita com o valor da rota recém adicionada subtraido de máximo e mínimo.

Os valores alvo são calculados com base na média de quilometragem das rotas pelo número de caminhões, como no algoritmo de backtracking, verifica-se se a divisão é equitativa ou não. Caso não seja, o máximo é 10% maior que a média e o mínimo é 10% menor que a média.

O caso base se torna a validação de se a soma das rotas está entre o máximo e o mínimo.

Caso no final haja uma rota sem alocação, o algoritmo usa o método guloso com quilometragem acumulada para alocar a rota restante.

3.4. Programação Dinâmica

Na implementação da programação dinâmica, foi utilizado um outro problema semelhante para auxiliar na resolução das rotas. Através da soma de subconjuntos, é possível encontrar a aproximação de um conjunto de números que seja igual a um valor desejado. Nesse caso, o valor desejado segue a mesma lógica do backtracking e da divisão e conquista, onde o valor máximo é a média de quilometragem das rotas pelo número de caminhões, case a divisão não seja equitativa, o valor máximo é 10% da média de quilometragem das rotas pelo número de caminhões.

Na matriz de programação dinâmica, cada linha representa uma rota, cada coluna representa um valor de 0 ao valor desejado, e cada célula representa a inclusão ou não da rota para a soma de sua coluna.

Assim, mesmo que a célula [i][j] termine com o valor 0, é possível decrementar o valor de j até que o valor da célula seja 1, e assim encontrar uma aproximação para o valor desejado.

Então, o algoritmo calcula a as rotas incluídas na solução, as alocando para um caminhão, e as retira da lista de rotas. Em seguida, o algoritmo recalcula a matriz de programação dinâmica, e repete o processo até que todas as rotas sejam alocadas.

Como no caso da divisão e conquista, o algoritmo usa o método guloso com quilometragem acumulada para alocar a rota restante caso não seja possível alocar todas as rotas.

4. Análise das execuções

4.1. Backtracking

O algoritmo de backtracking foi capaz de resolver instâncias de até 19 rotas em menos de 30 segundos.

Nota-se um crescimento exponencial semelhante à força bruta no tempo de execução do algoritmo, o que é esperado para o backtracking, que apenas reduz o número de chamadas recursivas.

Tamanho	Tempo (ms)
6	0
7	0
8	1
9	2
10	2
11	5
12	12
13	14
14	48
15	77
16	248
17	590
18	3109
19	8305
20	36712

4.2. Guloso

Em ambas as implementações, o algoritmo guloso foi capaz de resolver instâncias de até 10T em menos de 1 ms. Entretanto, o Guloso com Quilometragem Acumulada se mostrou mais eficaz em encontrar soluções sub-ótimas. Este comportamento era esperado, pois a estratégia de ordenação das rotas depende de uma escolha arbitrária de ordem.

O tempo de execução de ambos já era esperado, a complexidade do Guloso com Ordem das Rotas é $O(n)$, pois apenas insere as n rotas nos caminhões. Já o Guloso com Quilometragem Acumulada é $O(n * m)$, onde n é o número de rotas e m o número de caminhões, pois antes da inserção de cada rota, é necessário rearranjar os caminhões na fila de prioridade.

Tamanho	Tempo (ms)
20	0
40	0
60	0
80	0
100	0
120	0
140	0
160	0
180	0
200	0
220	0

4.3. Divisão e Conquista

O algoritmo de divisão e conquista foi capaz de resolver instâncias de até 10T em menos de 1 ms.

A complexidade do algoritmo se mostrou$O(2^{n})$, pois a cada rota adicionada, verifica-se se é possível incluir ou não a rota para manter a distribuição equitativa.

Tamanho	Tempo (ms)
20	3

4.4. Programação Dinâmica

O algoritmo de programação dinâmica foi capaz de resolver as instâncias de 10T em menos de 2 ms.

A complexidade gerada pela matriz de programação dinâmica é$O(n * m * k)$, onde n é o número de rotas, m a soma das rotas e k o número de caminhões. Entende-se que a complexidade é$O(n * m * k)$, pois forma-se uma matriz de n linhas e m colunas para inserir as rotas em um caminhão, e para cada caminhão, é necessário reconstruir esta matriz.

Tamanho	Tempo (ms)
20	0
40	0
60	0
80	0
100	0
120	1
140	1
160	1
180	1
200	2
220	2

5. Conclusão

O algoritmo de Backtracking demonstrou eficácia para instâncias de até 19 rotas em menos de 30 segundos. No entanto, a complexidade semelhante à força bruta no tempo de execução, conforme observado, mostra sua limitação para instâncias maiores.

O Guloso, embora rápido, apresenta soluções não ótimas, principalmente a implementação de ordem das rotas. Resolvendo instâncias de até 10T em menos de 1 ms, sua eficiência é notável.

A Divisão e Conquista se mostrou menos eficiente que os Gulosos e a Programação Dinâmica pois ele tem complexidade semelhante ao Backtracking sem trazer a otimalidade.

A Programação Dinâmica demonstrou bom desempenho. Essa abordagem pode ser uma escolha equilibrada, oferecendo eficiência sem comprometer a otimalidade.