

Relatório referente a etapa 1 do trabalho

Iago Floriano¹, Tiago Serique¹

¹Universidade Federal do Paraná, Curitiba, Brazil

{imf19, tsv19}@inf.ufpr.br

1. Recursos usados

Para a realização do trabalho foram utilizados a linguagem de programação Python3 devido sua facilidade para manipulação de dados, e a biblioteca Gurobi Optimizer como resolvidor de problemas lineares relaxados. Afim de facilitar a manipulação e execução, foi criado um makefile. Para instalar as dependências necessárias basta usar o comando "make install", e para executar o código, use "make exec".

2. Definindo o problema:

O problema apresentado é similar ao problema de empacotamento de caixotes (bin packing), porém usado em um contexto diferente. Neste problema, o contexto é o de um caminhão que precisa levar alguns itens ao seu destino, e deseja-se minimizar o número de viagens que este caminhão faz.

O caminhão possui uma capacidade de peso C . Existem n produtos, de I_1 até I_n , e cada um desses produtos tem um peso P_i associado a eles.

Neste problema é necessário que cada item esteja em uma viagem, com cada viagem sendo numerada de 1 a n , e que nenhuma das viagens do caminhão tenha itens com pesos somados maiores que a capacidade. Além disso, existem pares ordenados de números, ex. (3, 2), que indicam a ordem em que os itens devem ser entregues. Nesse caso, seria necessário que o item 3 seja mandado em uma viagem antes do item 2.

3. Modelagem do problema

O problema será resolvido de forma a usar programação linear, portanto é importante fazer uma modelagem do problema.

$$\text{minimizar } \sum_{i=1}^n v[i]$$

Sujeito a:

$$\sum_{j=1}^n IV[i][j] = 1 \quad \forall i / 1 \leq i \leq n, \quad \text{cada item deve ser colocado em apenas uma viagem.}$$

$$\sum_{i=1}^n IV[i][j] * w[i] \leq C * v[j] \quad \forall j / 1 \leq j \leq n, \text{ o peso de cada viagem não pode ser maior que a carga do caminhão.}$$

$v[i] \in \{0, 1\} \forall i, 1 \leq i \leq n$ ou uma viagem acontece ou não acontece.

$IV[i][j] \in \{0, 1\} \forall i, j, 1 \leq i, j \leq n$ item i está na viagem j ou não está.

$\sum_{j=1}^n IV[i_1][j] > IV[i_2][j] / \forall (i_1, i_2) \in P$ restrição de ordem dos pares ordenados

Variáveis usadas:

v , um vetor para saber se a viagem i aconteceu. Sendo $v[i] = 1$ caso viagem aconteceu, e $v[i] = 0$ caso contrário.

C , a carga máxima do caminhão

w , um vetor com os pesos dos itens, sendo $w[i]$ o peso do item I_i

n , o número de itens a serem levados, também é o número máximo de viagens que pode ocorrer (cada item precisando de uma viagem com apenas aquele item)

IV , uma matriz em que $IV[i][j]$ representa se o item I_i está na viagem $v[j]$

P , o conjunto de pares ordenados da restrição de ordem das viagens

4. Criando um subproblema

Esse problema é muito complexo de ser calculado, então primeiramente será feito estimativas otimizadas de resolução do problema para que se possa usar um algoritmo de branch and bound depois.

A estimativa desse problema foi feita tirando algumas restrições que causam com que o problema tenha resolvimento muito demorado e relaxando outras restrições para que admitissem valores reais.

As restrições removidas foram as seguintes:

$v[i] \in \{0, 1\}$ virou a restrição $v[i] \in R / 0 \leq v[i] \leq 1 \forall i, 0 \leq i \leq n$

$IV[i][j] \in \{0, 1\}$ virou a restrição $IV[i][j] \in R / 0 \leq IV[i][j] \leq 1 \forall i, j / 1 \leq i, j \leq n$

a restrição $\sum_{j=1}^n IV[i_1][j] > IV[i_2][j] / \forall (i_1, i_2) \in P$ foi removida, pois seriam muitas restrições utilizadas na utilização da biblioteca Gurobi, fazendo o cálculo dessa estimativa muito lento.

5. Exemplo usado:

Como entrada, foi utilizado o exemplo disponibilizado, onde a quantidade de itens é 5, com pesos 5, 6, 4, 8, 5, a capacidade do caminhão de 10, os pares ordenados (2, 3) e (5, 1), e uma solução parcial com 2 pares, onde a viagem do item 3 seria a de número 2 e a viagem do item 5 seria a de número 3. O resultado desse exemplo é igual a 2,8 viagens, o mesmo valor da saída do programa em feito com o resolvidor gurobi.

6. Referências bibliográficas:

Understanding and Using Linear Programming. Jiří Matoušek, Bernd Gärtner. 2007.

Gurobi Optimization. Disponível em: https://www.gurobi.com/documentation/9.5/quickstart_mac/cs_grbpy_the_gurobi_python.html. Último acesso em: 13/04/2022.