MC859 - Relatório 2: Parte Exata

Guilherme Guidotti Brandt (235970)

1 Formulação direta

A formulação básica que utilizei para o problema foi a seguinte, baseada nas discussões em sala:

$$
\begin{aligned}
(\mathcal{SP}) \quad & \min \sum_{i \in R} w_i y_i & & \\
\text{s.a.} \quad & \sum_{j \in N} x_{ij} = 1 & \forall i \in R \quad & (1) \\
& \sum_{i \in R} l_i x_{ij} \leq L & \forall j \in N \quad & (2) \\
& a_j - h_i x_{ij} \geq 0 & \forall i \in R, j \in N \quad & (3) \\
& y_i + M_j (1 - x_{ij}) - \sum_{k=1}^{j-1} a_k \geq 0 & \forall i \in R, j \in N \quad & (4) \\
& & & \\
& x_{ij} \in \{0, 1\} & \forall i \in R, j \in N & \\
& y_i \in \mathbb{R}_{\geq 0} & \forall i \in R & \\
& a_j \in \mathbb{R}_{\geq 0} & \forall j \in N &
\end{aligned}
$$

As variáveis dessa formulação são:

- x_{ij}, que indica se o item i está no nível j na solução
- y_i, que indica a altura da base do item i na solução
- a_j, que indica a altura de um nível (isto é, a distância da base do nível à base do nível acima)

E as restrições:

(1) Garante que cada item i está em exatamente um nível na solução. Como o problema é de minimização, seria válido também exigir que cada item esteja em pelo menos um nível.

(2) Garante que a soma das larguras dos itens em cada nível é menor ou igual ao tamanho do recipiente, L (empacotamento nos níveis).

(3) Força a variável a_j (a altura do nível j) a ser maior ou igual à altura de cada item no nível j. Isso corresponde à restrição de que a altura de um nível é igual à maior altura de um item nele, uma vez que qualquer altura estritamente maior que algum item no nível só pode aumentar o custo da solução.

(4) Quando $x_{ij} = 1$, i.e., o item i está no nível j na solução, essa restrição corresponde a $y_i \geq \sum_{k=1}^{j-1} a_k$, ou seja, a altura da base do item i é igual à soma das alturas dos níveis abaixo de j (que corresponde a dizer que a base do item é a mesma base do nível em que ele está). Quando $x_{ij} = 0$, a restrição é trivialmente satisfeita, desde que M_j seja grande o suficiente (em particular, maior ou igual a $\sum_{k=1}^{j-1} a_k$).

1.1 Variações

1.1.1 Indicadores de uso dos níveis

A formulação acima tem um problema de simetria, no sentido que uma mesma solução pode ser representada de diferentes formas, deixando níveis vazios.

Por exemplo, se L_i é o conjunto dos itens no nível i, as soluções

$$L_1 = \{1, 2\} \qquad L_2 = \{3, 4\}$$

e

$$L_2 = \{1, 2\} \qquad L_4 = \{3, 4\}$$

são equivalentes (na segunda, temos níveis 1 e 3 vazios), mas dão atribuições de valores binários diferentes às variáveis na formulação.

Podemos quebrar essa simetria adicionando uma variável para indicar o uso de um determinado nível, e forçando os níveis a obedecer uma ordem.

Adicionamos então uma variável $u_j \in \{0, 1\} \quad \forall j \in N$, e as restrições:

- $u_j \geq x_{ij}$, que garante que um item só pode estar em um nível se o nível for usado
- $u_{j-1} \geq u_j$, que garante que um nível só pode ser usado caso o nível imediatamente anterior a ele seja usado. Evidentemente, excluímos o primeiro nível (nível 1) dessa condição.

Além disso, temos a seguinte proposição:

Proposição. *A seguinte desigualdade é válida:*

$$\sum_{i \in R} x_{ij} \geq u_j \quad \forall j \in N$$

Demonstração. Evidentemente, se $u_j = 0$, então não pode existir item no nível j (pela primeira restrição adicionada), portanto $\sum_{i \in R} x_{ij} = 0$, logo a desigualdade é valida.

Além disso, se $\sum_{i \in R} x_{ij} = 0$ (i.e., nenhum item está no nível j), não faz sentido usar o nível na solução (dada uma solução ótima que usasse o nível j sem nenhum item, podemos facilmente obter uma solução equivalente que não usa o nível j). Logo, se $u_j = 1$, então deve existir pelo menos um item no nível, e portanto $\sum_{i \in R} x_{ij} \geq 1 = u_j$. □

A formulação a seguir incorpora essas melhorias na formulação anterior:

$$
\begin{aligned}
(\mathcal{SP}') \quad & \min \sum_{i \in R} w_i y_i & \\
\text{s.a.} \quad & \sum_{j \in N} x_{ij} = 1 & \forall i \in R \\
& \sum_{i \in R} l_i x_{ij} \leq L & \forall j \in N \\
& a_j - h_i x_{ij} \geq 0 & \forall i \in R, j \in N \\
& y_i + M_j(1 - x_{ij}) - \sum_{k=1}^{j-1} a_k \geq 0 & \forall i \in R, j \in N \\
& u_j \geq x_{ij} & \forall i \in R, j \in N \\
& u_{j-1} \geq u_j & \forall j \in N \setminus \{1\} \\
& \sum_{i \in R} x_{ij} \geq u_j & \forall j \in N \\
& & \\
& x_{ij} \in \{0, 1\} & \forall i \in R, j \in N \\
& y_i \in \mathbb{R}_{\geq 0} & \forall i \in R \\
& a_j \in \mathbb{R}_{\geq 0} & \forall j \in N \\
& u_j \in \{0, 1\} & \forall j \in N
\end{aligned}
$$

1.2 Formulação com geração de colunas

Outra possível modificação da formulação original é uma formulação por geração de colunas.

Em particular, podemos reformular o problema original considerando o subproblema de empacotamento de itens em um nível, de forma que os problemas de precificação (considerando $|N|$ níveis, temos $|N|$ problemas de precificação independentes, um para cada nível) se torna um problema da mochila, que embora ainda NP-difícil pode ser resolvido em tempo pseudopolinomial e de forma bastante rápida na prática, especialmente para instâncias relativamente pequenas, como é o caso. Essa abordagem é similar à aplicada ao problema de *Cutting Stock* por Gilmore e Gomory [Gilmore e Gomory 1961].

O problema mestre da geração de colunas nesse caso é:

$$
\begin{aligned}
(DW) \quad & \min \sum_{i \in R} w_i y_i \\
\text{s.a.} \quad & \sum_{j \in N} \sum_{k} x_i^k \lambda_{jk} = 1 && \forall i \in R \quad (5) \\
& a_j - h_i \sum_{k} x_i^k \lambda_{jk} \geq 0 && \forall i \in R, j \in N \quad (6) \\
& y_i + M_j \left(1 - \sum_{k} x_i^k \lambda_{jk} \right) - \sum_{l=1}^{j-1} a_j \geq 0 && \forall i \in R, j \in N \quad (7) \\
& \sum_{k} \lambda_{jk} = 1 && \forall j \in N \quad (8) \\
& \lambda_{jk} \in \mathbb{R}_{\geq 0} && \forall j \in N \\
& y_i \in \mathbb{R}_{\geq 0} && \forall i \in R \\
& a_j \in \mathbb{R}_{\geq 0} && \forall j \in R
\end{aligned}
$$

E o subproblema j de geração de colunas, para $j \in N$, é exatamente um problema da mochila, usando como pesos dos itens as variáveis duais das restrições do problema mestre:

$$
\begin{aligned}
(PR) \quad & \zeta^j = \max \sum_{i \in R} (u_i^1 - h_i u_i^2 - M u_i^3) x_i + u_0 \\
\text{s.a.} \quad & \sum_{i \in R} l_i x_{ij} \leq L && \forall j \in N \quad (9) \\
& x_i \in \{0, 1\} && \forall i \in R \quad (10)
\end{aligned}
$$

Implementei a geração de colunas, mas não consegui implementar o Branch-and-Price de forma eficaz, e portanto não consegui validar completamente a viabilidade dessa formulação.

Entretanto, consegui ao menos implementar a geração de colunas para encontrar uma solução ótima para a relaxação linear do problema mestre.

Outra formulação possível por geração de colunas envolveria gerar uma atribuição completa de itens a níveis (ou seja, movendo a primeira restrição da formulação original também para subproblema de precificação); isso diminui a quantidade de subproblemas de $|N|$ para um, mas torna o problema de precificação significativamente mais complexo, e não pareceu promissor.

Um próximo passo possível seria incorporar as melhorias da seção anterior ao problema mestre da geração de colunas; não segui essa abordagem para focar em outros aspectos da implementação.

2 Formulação com fluxo

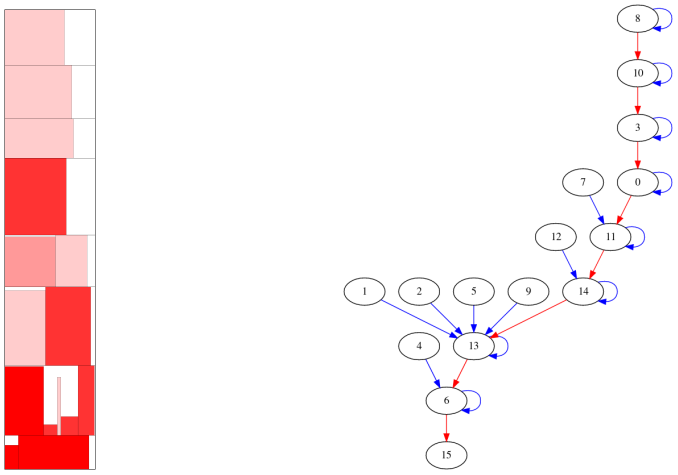
Uma terceira formulação envolve imaginar o problema como um problema em grafos, considerando a função de custo e restrições de altura como fluxo precificado.

Em particular, podemos considerar cada item como um nó, e definir que um item i está no nível do item j se existe um arco do item i para o item j (arcos item $\rightarrow$ nível), aproveitando que, no máximo, existem tantos níveis quanto itens. Para definir a ordem relativa dos níveis, basta determinar um segundo conjunto de arcos entre os itens que definem níveis (arcos de nível). Podemos definir que um item define um nível se ele está no próprio nível, i.e., se existe um arco item $\rightarrow$ nível que é um laço no nó correspondente.

Para traduzir a função de custo, podemos dizer que cada nó de item produz fluxo igual ao peso do item, e cada arco de um nível i para o nível anterior j tem custo igual à altura de j. Dessa forma, o custo do fluxo seja a altura do nível imediatamente abaixo vezes o peso dos itens acumulado de cada nível acima. Adicionamos também um único consumidor, um nó extra que não corresponde a nenhum item, e que representa a "base" da solução; o arco de nível que entra neste nó tem custo 0, correspondente a itens na base da solução terem custo multiplicado por 0.

É possível notar que a soma dos custos dos fluxos nos arcos de nível corresponde exatamente à função de custo do problema original.

Por conta da estrutura do problema (cada item está em exatamente um nível), o problema se torna análogo a encontrar uma árvore geradora (ou, no caso, uma ramificação) que minimiza o custo total do fluxo. As figuras abaixo ilustram a equivalência entre as duas representações de soluções do problema, para uma instância com 15 itens.



(a) Solução de strip packing em níveis. (b) Solução de ramificação correspondente.

A formulação por fluxo é a seguinte:

$$
\begin{aligned}
(\mathcal{F}) \quad & \min \sum_{i \in R} \sum_{\substack{j \in R \\ i \neq j}} w_i f_{ij} \\
\text{s.a.} \quad & \sum_{j \in R} x_{ij} = 1 && \forall i \in R && (11) \\
& \sum_{i \in R} l_i x_{ij} \leq L x_{jj} && \forall j \in R && (12) \\
& h_i x_{ij} \leq h_j x_{jj} && \forall i \in R, j \in R && (13) \\
& \sum_{j \in R} y_{ji} \leq x_{ii} && \forall i \in R && (14) \\
& \sum_{j \in R} y_{j\emptyset} \leq 1 && && (15) \\
& \sum_{j \in R \cup \{\emptyset\}} f_{ij} \leq W y_{ij} && \forall i \in R, j \in R \cup \{\emptyset\} && (16) \\
& \sum_{j \in R \cup \{\emptyset\}} f_{ij} = \sum_{j \in R} w_j x_{ji} + \sum_{j \in R} f_{ji} && \forall i \in R && (17) \\
& x_{ij} \in \{0,1\} && \forall i \in R, j \in R \\
& y_{ij} \in \{0,1\} && \forall i \in R, j \in R \cup \{\emptyset\} \\
& f_{ij} \in \mathbb{R}_{\geq 0} && \forall i \in R, j \in R \cup \{\emptyset\} && (18)
\end{aligned}
$$

Onde temos as variáveis:

- x_{ij}, que indica se o item i está no nível do item j na solução (arcos item $\rightarrow$ nível)
- y_{ij}, que indica se o nível j está logo abaixo do nível i (arcos de nível)
- f_{ij}, que dá o fluxo em um arco de nível (i, j).

E as restrições:

(11) Que determina que cada item está em exatamente um nível (i.e. existe exatamente um arco item $\rightarrow$ nível para cada item).

(12) Que garante o empacotamento dos itens nos níveis (ou seja, a soma das larguras dos itens em um nível não excede o limite L). Aqui podemos multiplicar L por x_{jj} (que indica se j define um nível na solução), uma vez que, se um item j não define nível, não pode haver nenhum item no nível j (e portanto a largura total deve ser zero). Isso fortalece a relaxação da formulação, uma vez que $x_{jj} \leq 1$.

(13) Que, para evitar algumas simetrias, determina que a altura do item que define o nível deve ser a maior altura no nível. A mesma técnica de multiplicação do lado direito por x_{jj} da restrição anterior é válida aqui.

(14) Que determina que no máximo um arco de nível entra em cada nível definido por um item, e nenhuma entra em um nível que não é usado (indicado pelo laço no nó do item correspondente).

(15) Que determina que no máximo um arco de nível entra no nó especial que serve como consumidor do fluxo.

(16) Que determina a capacidade de um arco de nível. Um arco de nível (i, j) tem capacidade W se for usado (i.e. o nível j está imediatamente abaixo do nível i) e 0 caso contrário. O valor de W só precisa ser grande o suficiente para suportar a soma dos pesos de cada item na solução.

3 Resultados computacionais

Executei os programas em uma máquina com o sistema operacional Manjaro Linux, com 32 GB de RAM e um processador Intel i9-12900K (especificado na tabela 1), utilizando a build v10.0.3rc0 (linux64) do Gurobi, com a licença acadêmica.

Modelo	i9-12900K
Fabricante	Intel®
Microarquitetura	Alder Lake
Núcleos	8 performance / 8 eficiência
Threads	24
Frequência	3.20GHz (performance) / 2.40 GHz (eficiência)
Cache	30 MB Intel® Smart Cache
Cache L2	14 MB

Tabela 1: Especificação da CPU

Cada instância foi aplicada ao programa com a formulação ($\mathcal{SP}'$) e com a formulação ($\mathcal{F}$) separadamente.

Para cada instância, uma solução heurística é gerada com iterações randomizadas das heurísticas best-fit e first-fit e melhorada com o BRKGA, com um tempo fixo de 30s. Defini o tempo limite para o Gurobi em 600 segundos.

As tabelas 2 e 3 apresentam os resultados para a formulação ($\mathcal{SP}'$) e ($\mathcal{F}$), respectivamente. Onde o programa encontra a solução ótima, o melhor limitante é omitido.

De forma geral, ambas as formulações conseguem lidar com instâncias de tamanho 10 razoavelmente bem. Para instâncias de tamanho 15, o tempo de execução varia consideravelmente, mas a formulação por fluxo apresenta tempos consistentemente piores que a formulação ($\mathcal{SP}'$).

Instância	Valor	Heurística	Melhor limitante	Tempo (Gurobi)
1_10	**1504**	**1504**	–	0,19s
2_10	4211	4313	–	0,60s
3_10	**1553**	**1553**	–	1,10s
1_15	**5036**	**5036**	–	185,10s
2_15	**4928**	**4928**	–	45,79s
3_15	**4634**	**4634**	–	9,06s
1_20	–	8461	6225,831	**600,01s**
2_20	–	6953	3944,267	**600,00s**
3_20	–	11036	6515,507	**600,01s**

Tabela 2: Resultados computacionais para a formulação ($\mathcal{SP}'$)

Instância	Valor	Heurística	Melhor limitante	Tempo (Gurobi)
1_10	**1504**	**1504**	–	0,35s
2_10	**4211**	**4211**	–	2,23s
3_10	**1553**	**1553**	–	0,90s
1_15	**5036**	**5036**	–	562,22s
2_15	–	**4928**	–	600,02s
3_15	**4634**	**4634**	–	81,48s
1_20	–	8461	3915,152	**600,01s**
2_20	–	6953	2640,020	**600,01s**
3_20	–	11036	3798,881	**600,02s**

Tabela 3: Resultados computacionais para a formulação ($\mathcal{F}$)

Nenhuma das duas formulações foi capaz de resolver de forma satisfatória instâncias de tamanho 20, ambas atingindo o tempo limite. Mesmo nesse caso, a primeira formulação conseguiu obter limitantes inferiores melhores que a formulação por fluxo.

É possível notar que, em praticamente todas as instâncias onde o ótimo foi provado, a solução heurística condiz com o valor ótimo obtido. Isso indica que a heurística é boa, e o gargalo dos algoritmos está provavelmente relacionado à dificuldade de obter um limitante inferior adequado.

Referências

[Gilmore e Gomory 1961]GILMORE, P. C.; GOMORY, R. E. A linear programming approach to the cutting-stock problem. *Operations Research*, INFORMS, v. 9, n. 6, p. 849–859, 1961. ISSN 0030364X, 15265463. Disponível em: <http://www.jstor.org/stable/167051>.