1 O Problema do General Bárbaro

Para ver se realmente compreendemos as ideias que vimos anteriormente, é sempre útil ver se conseguimos aplicá-las a algo concreto. Para isso, desenvolveremos aqui um problema específico (o "problema do general bárbaro"[1]). Mas faremos isso não com vistas em colocá-lo diretamente no *framework* desenvolvido até agora, mas sim em entendê-lo, pensar em diversos métodos de resolvê-lo e comparar esses métodos.

1.1 O Problema

O general romano Sulla, em suas campanhas pela Grécia, fez (entre outras ocisas) o que qualquer outro general bárbaro faria em seu lugar, isto é, saqueou tesouros locais para obter vantagem na guerra (diferente de outros generais bárbaros, Sulla prometeu devolver parte do saque). Mas isso constituiu um problema, já que ele tinha uma grande variedade de tesouros para escolher levar ou não, enquanto que a capacidade de carga de seu exército era muito limitada, umas vez que dispunha de (relativamente) poucos navios. Mas o que criava a maior dificuldade nesse projeto era não o volume dos tesouros, mas o seu peso.

O valor de cada peça de tesouro é dado por uma função complicada de interesses internos e externos (no que influem, entre outros fatores, o preço pelo qual Sulla poderia vender dado item, no valor que cada soldado atribui ao item, no valor que seus conterrâneos atribuiriam ao item e na reputação que carregar um item lhe conferiria) e é sumarizada no que chamamos de *valor Sulla*, $\mathscr{S}$, de um item. Depois de calcular o $\mathscr{S}$ de cada item relevante, Sulla compilou a seguinte tabela preliminar com o peso médio e valor médio de cada item[2]:

Item	Valor	Peso	Quantidade
Livros	10	5	1000
Barras de Ouro	100	500	85
Estátuas de Ouro	500	4000	7
Jóias	50	50	200

Sem muito sucesso em resolver o problema, Sulla mandou que chamassem um matemático local para resolver o problema. Theon de Smyrna, estava por perto e foi convocado. No início, se sentiu incomodado no por ter que deixar de lado seu trabalho com os sólidos de Platão em favor de um ajudar um bárbaro a resolver um problema desse tipo inferior, como considerava, mas, notando que pode ser mais difícil para um morto desenvolver trabalhos com os sólidos de Platão, decidiu ajudar Sulla com seu problema.

[1] Também conhecido como "o problema da mochila", ou *knapsack problem*, em inglês.

[2] Não há evidências de que tal tabela tenha sido feitas, mas é mantida aqui pela conveniência ao exemplo.

Ao ter a situação explicada, Theon deu uma leve risada e disse o que se segue. “Você me dá um problema grandioso e difícil, oh grande Sulla! Os Deuses devem ter nos posto juntos, porque eu devo ser um dos poucos helenos, se não o único, que sabe resolvê-lo! Isto porque vi meu pai o resolvendo quando era criança. Espero que não se incomode em ouvir por alguns momentos um velho como eu relembrar a infância, ainda mais que isso te será útil.”

“Continue.” - disse Sulla.

“Meu pai era um fazendeiro muito forte, mas também tinha o seus limites. Ás vezes, precisava realizar trabalhos em partes distantes e, para isso, precisava levar suas ferramentas. Cada ferramenta faria o seu trabalho mais fácil, ou o ajudaria a terminá-lo mais rápido, ou lhe proporcionaria maior conforto ao realizá-lo. Cada uma, também, tinha o seu peso, e meu pai, forte como era, não conseguia carregar todas. Então, ele compilou uma tabela parecida com a sua, com um “valor” que cada ferramenta lhe proporcionaria, assim como seu peso.”

“Mas, meu caro, esse problema parece muito mais simples do que o meu: seu pai só precisava decidir entre levar uma ferramenta ou não, enquanto que eu preciso decidir entre quantos de cada tipo de item levar!” - disse Sulla.

“É verdade que o problema de meu falecido pai parece muito mais simples do que o seu. Mas toda pessoa bem educada, como tenho certeza ser o seu caso, oh conquistador de nações!, sabe que não se pode confiar nas aparências. Neste caso, ver que os problemas são essencialmente os mesmos é muito simples. Toda criança sabe que qualquer número natural pode ser decomposto como uma soma de potências de dois. Por exemplo, $11 = 2^0 + 2^1 + 2^3$. Além disso, essa decomposição é única, cada potência aparecendo no máximo uma vez. Como pode ver, então, o seu problema é um problema de decisão: decidir quais potências de 2 comporão as quantidades de cada tesouro que levará. Esse problema só tem o posssível incoveniente de fazer uso de mais variáveis. Mas não muito mais: $\lfloor log_2(q_i) \rfloor$, em que q_i é a quantidade a ser levada do item i.[3]”

“Mas esse problema adiciona complicações extras que não me parecem tão necessárias.” - disse Sulla.

“Não tanto assim, mas tudo bem. Não quero tomar muito mais tempo seu, então vou direto ao ponto. Um modelo mais geral ao problema que deseja resolver é o seguinte:

$$
\begin{aligned}
maximizar \sum_{i=1}^{n} q_i \mathscr{S}_i, \\
tal\ que \sum q_i p_i \le C, \\
0 \le q_i \le l_i, q_i\ inteiro
\end{aligned}
$$

[3]Estamos adicionando, aqui e mais adiante, notação um pouco mais moderna do que a que o velho Theon poderia ter usado, por conveniência ao exemplo.

Em que q_i é a quantidade a ser levada do item i, p_i o peso do item i (em $\mathscr{S}$), C a capacidade máxima a ser levada e l_i limite máximo do item i a ser levado (não podemos levar mais do que l_i do item i, talvez porque não existam mais do que l_i desses itens).

Esse é um problema de programação inteira. Existem muitas formas de resolver problemas de programação inteira. Se tiver algum tempo sobrando, eu poderia desenvolver uma outra forma com você. Será divertido e intrutivo."

"Vá em frente." - disse Sulla.

"Imagine que você sabe resolver esse problema para $C' < C$ e para todo subconjunto dos itens (um subconjunto de $\{1, ..., n\}$) e tenha em mãos uma solução para ele. Se retirarmos dessa solução um item i, de peso p_i, deve ser claro que a solução restante é ótima para o problema cuja capacidade de $C - p_i$ e conjunto de itens $\{1, ..., n\} - i$. Isso indica que uma solução ótima goza de uma subestrutura ótima. Isto é, o retiro de itens dessa solução gera uma solução que é ótima para outro problema, de fato um subproblema, mais simples. Se temos a solução para algum desses subproblemas, só precisamos expandi-lo, considerando itens que não foram considerados. Para cada um desses itens, sabendo que a nova solução ótima $op(i, c)$ para uma capacidade c, ao considerar a adição de um item i, é:

$$op(i, c) = \begin{cases} op(i-1, c), & se\ p_i > c \\ max\{op(i-1, c), op(i-1, c-p_i) + \mathscr{S}_i\} & se\ p_i \leq c \end{cases}$$

Fazendo $op(0, C) := 0$ e computando os valores de $op(i, c)$ por essa recursão[4] nos dá o valor ótimo que o meu pai queria. Se quisermos resolver o seu problema, precisamos fazer apenas uma pequena alteração, como a seguinte:

$$op(i, c) = max_{q_i}\{op(i-1, c - q_i p_i) + q_i \mathscr{S}_i \mid c \geq q_i p_i\}$$

"

[4]Conhecida como recursão de Bellman

Leituras adicionais

[1] Niederliński, Antoni, “A gentle guide to constraint logical programming via Eclipse”, 3rd edition, Jacek Skalmierski Computer Studio, Gliwice, 2014