Prova Final - Otimização
Profs. André/Murilo
04 de julho de 2023

1. Considere o seguinte problema.

   Entrada: Um conjunto de inteiros positivos $A = \{a_1, a_2, ...a_n\}$ e um inteiro positivo k.

   Saída: Subconjunto de A cuja soma dos elementos tem valor mais próximo possível de k, ou seja, subconjunto $S \subseteq A$ tal que $|k - \sum_{s \in S} s|$ é a menor possível.

   (a) (15 pontos) Apresente um algoritmo que use Programação Dinâmica para resolver este problema.

   (b) (15 pontos) Apresente uma modelagem em Programação Linear Inteira para este problema.

2. Considere o PL abaixo:

   max $x + y$
   s.t.
   $ax + by \leq 1$
   $x, y \geq 0$

   Quais condições os números reais a e b devem respeitar tal que:

   (a) (10 pontos) O seja PL é inviável.

   (b) (10 pontos) O seja PL é ilimitado.

   (c) (10 pontos) Tenha solução ótima única.

3. Considere o seguinte problema.

   Uma confeitaria faz 2 tipos de tortas, uma de maçã e outra de morango. Essa confeitaria tem um Chef Confeiteiro, Pedro, e mais dois funcionários, Ana e João. Ana trabalha 4 horas diárias e seu trabalho inclui a preparação das maçãs para as tortas. O tempo gasto por Ana para preparar as maçãs para fazer uma torta é de 1 hora. João trabalha 3 horas diárias e faz a preparação dos morangos. O tempo gasto por João para preparar os morangos para uma torta é de 30 minutos. Para finalizar as tortas, Pedro, que trabalha 9 horas por dia, gasta 1 hora e 30 minutos para a de maçã e 1 hora para a de morango. A confeitaria vende cada torta de maçã por 30 reais e de morango por 50 reais. Assumindo que toda a produção é sempre vendida, quantas tortas de cada tipo devem ser fabricadas em um dia para maximizar o ganho?

   (a) (15 pontos) Faça um esboço de solução para este problema, dizendo qual técnica você usaria.

   (b) (5 pontos) Agora generalize o problema para n tipos de torta e n ajudantes do Chef (um para cada tipo), cada torta com seu valor de venda, tempo de preparação dos ingredientes, jornada de trabalho do ajudante e tempo gasto pelo Chef na finalização das tortas. Como você resolveria este problema?

   (c) (10 pontos) Apresente um outro esboço de solução usando uma técnica diferente da usada nos itens 3a e 3b.

4. (10 pontos) Em um algoritmo de *Branch and Bound*, qual a diferença entre cortes por viabilidade e cortes por otimalidade?

Gabarito

1.

   (a) Considere que $S(i,j)$ representa o subproblema (a instância) onde $A = \{a_1, a_2, \dots, a_i\}$ e $k = j$. O subproblema inicial é $S(n,k)$. A seguinte recorrência resolve o problema:

$$S(i,j) = \begin{cases} 0, & \text{se } j = 0, \text{ (a resposta é o conjunto vazio)} \\ |j|, & \text{se } i = 0, \text{ (a resposta é o conjunto vazio)} \\ \min\{S(i-1, j-a_i), S(i-1,j)\}, & \text{caso contrário. (ou } a_i \text{ está ou não está).} \end{cases}$$

   Seja $m = \sum_{i=1}^{n} a_i$. A estrutura de dados para armazenar os subproblemas é uma matriz M $(n+1)\times(m+k+1)$ indexada por $[0..n] \times [-m..k]$. Primeiramente preenchemos a base, ou seja, $M[i,j] = 0$, para todo i, j tal que $j = 0$ (coluna no meio), e $M[i,j] = |j|$, para todo i, j tal que $i = 0$ (primeira linha). A ordem de preenchimento da matriz é linha por linha, iniciando com $i = 1$ até $i = n$.

   (b)

   Seja $(x_1, x_2, \dots, x_n)$ o vetor característico de uma solução $S \subseteq A$.

$$\begin{aligned}
&\min y \\
&\text{s.t.} \\
&y \geq k - \sum_{i=1}^{n} a_i x_i \\
&y \geq \sum_{i=1}^{n} a_i x_i - k \\
&y \geq 0 \\
&x_i \leq 1 \quad \forall i = 1..n \\
&x_i \geq 0 \quad \forall i = 1..n \\
&x_i \in \mathbb{Z} \quad \forall i = 1..n
\end{aligned}$$

2.

   (a) É sempre viável.

   (b) $a \leq 0$ OU $b \leq 0$.

   (c) $a > 0$, $b > 0$ E $a \neq b$.

3.

   (a) Podemos modelar por PLI e resolver graficamente. Seja ma e mo a quantidade de tortas de maçã e morango, respectivamente.

$$\begin{aligned}
&\max 30ma + 50mo \\
&\text{s.t.} \\
&ma \leq 4 \\
&mo \leq 6 \\
&1.5ma + 1mo \leq 9 \\
&ma, mo \geq 0 \\
&ma, mo \in \mathbb{Z}
\end{aligned}$$

   Os vértices do politopo tem coordenadas inteiras e a solução é 2 tortas de maça e 6 de morango, dando um ganho de 360 reais.

   (b) Podemos modelar por PLI o caso geral. Sejam n o número de tortas e x_i a quantidade a ser produzida da torta do tipo i. Sejam v_i, p_i, a_i, c_i e C, que são, respectivamente, o valor de cada tipo de torta, o tempo de preparação de ingredientes de cada tipo (em horas), a jornada de trabalho de cada ajudante (em horas), o tempo do chef para finalizar cada torta (em horas) e a jornada do chef (em horas). O primeiro passo é converter tudo em quantidade de tortas. Seja $m_i = sa_i/p_i$, o número máximo de tortas para as quais o ajudante da torta de tipo i pode preparar os ingredientes. Assim, a modelagem em PLI seria:

$$\max \sum_{i=1}^{n} v_i x_i$$

s.t.

$$x_i \leq m_i \quad \forall i = 1..n$$

$$\sum_{i=1}^{n} c_i x_i \leq C$$

$$x_i \geq 0 \quad \forall i = 1..n$$

$$x_i \in \mathbb{Z} \quad \forall i = 1..n$$

Podemos resolver usando PL relaxado (sem certeza de solução) ou usar um algoritmo guloso (precisaria provar que funciona) ou usar PD ou B&B.

(c) Por exemplo, podemos usar PD. Seja a recorrência abaixo, onde $T(i, j)$ é o ganho máximo ao se escolher entre as tortas dos tipos de 1 a i, sendo que o chef tem disponível apenas j horas. Os demais dados de cada torta são os mesmos.

$$T(i, j) = \begin{cases} 0, & \text{se } j = 0 \text{ (chef sem tempo) ou } i = 0 \text{ (sem tortas)} \\ \max\{T(i-1, j - kc_i) \mid 0 \leq k \leq m_i\}, & \text{caso contrário..} \end{cases}$$

Nesta recorrência, escolhemos em cada nível quantas tortas do tipo i serão feitas. A estrutura de dados seria uma matriz $(n + 1) \times (C + 1)$ e devemos calcular linha por linha de cima para baixo.

Observe que os valores de C, c_i e m_i deveriam ser inteiros, logo os tempos devem ser convertidos para unidades adequadas.

4. Cortes por viabilidade se referem aos cortes de ramos em que não existem soluções viáveis, descantando assim enumerações desnecessárias. Normalamente acontecem na escolha de candidatos para próximos nós a serem abertos.

   Já os cortes por otimalidade se refere ao uso de funções limitantes para cortar ramos que garantidamente não superam a melhor solução já encontrada até o momento.