Matematické programování

13. března 2022

1. Linerární programování
2. Celočíselné programování

Matematické programování

- Řada rozvrhovacích problémů může být formulována jako matematické programy
- Lineární programování, nelineární programování, celočíselné programování
- Předměty
  - PřF:M0160 Optimalizace
  - ESF:BKM_OPRO Optimalizace a rozhodování

Lineární programování

- Lineární program (LP)

  Minimalizace $c_1x_1 + c_2x_2 + \cdots + c_nx_n$

  za předpokladu

  $$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \geq b_1$$
  $$a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \geq b_2$$
  $$\ldots$$
  $$a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \geq b_m$$
  $$x_j \geq 0 \text{ pro } j = 1, \ldots, n$$
  $$x_j \in \mathbb{R} \text{ pro } j = 1, \ldots, n$$

  V maticovém zápisu: minimalizace $\overline{c}\,\overline{x}$

  za předpokladu $\mathbb{A}\overline{x} \geq \overline{b}$

  $\overline{x} \geq 0$

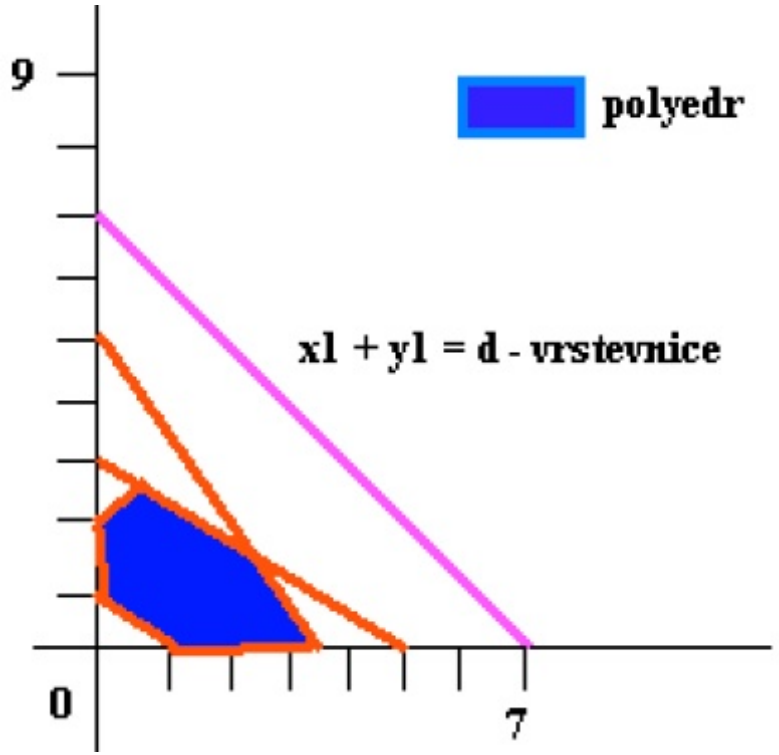
Lineární programování: příklad

- Výrobce vyrábí 3 výrobky A, B, C, na které spotřebuje materiál M a pracovní dobu P
- Maximalizujeme denní "Zisk"za předpokladu, že platí:
  - zisk z 1 kusu A = 500 Kč, spotřebujeme 4M a 2P.
  - zisk z 1 kusu B = 800 Kč, spotřebujeme 1M a 5P.
  - zisk z 1 kusu C = 300 Kč, spotřebujeme 2M a 1P.
  - přitom platí denní omezení materiálu M$\leq$30 kusů a pracovních hodin P$\leq$48 hodin (např. 6 lidí pracujících 8 hodin denně)
- Jak lze použít obecný LP vzorec?
  - max(Zisk) = 500*A + 800*B + 300*C ($c_1 = 500, x_1 =$ A,...)
  - 4*A + 1*B + 2*C $\leq$ 30 (omezení materiálu)
  - 2*A + 5*B + 1*C $\leq$ 48 (omezení prac. hodin)
- Jak bude vypadat výsledek? V jakých proměnných budeme mít uloženou odpověď na naši otázku? Co budou vyjadřovat?
- Bude to použitelné v praxi?
- Max vs. min není problém, neboť platí: $\max f(x) = -min(-f(x))$ pro $x \in \mathbb{R}^n$

Lineární programování: metody řešení

- Pro malé 2D, 3D problémy si často vystačíme s grafickým řešením
  - cíl je rovnice s parametrem = vrstevnice v ploše
  - omezení jsou poloroviny
  - oblast řešení je průnik polorovin (polyedr, tj. mnohostěn)
  - řešení je průnik rovnice s parametrem a vzniklého polyedru

- Simplexová metoda
  - efektivní nalezení řešení
  - většinou polynomiální čas
  - použití v praxi pro řešení rozsáhlých problémů
- Elipsoidová metoda
- Metoda vnitřních bodů



Celočíselné programování

- Celočíselné programování (integer programming IP)
  - lineární programování + všechny proměnné celočíselné
- Mixed-integer programming (MIP)
  - použity celočíselné i reálné obory hodnot proměnných
- Mnohem obtížnější než lineární programování
- Pro rozvrhování mnohem užitečnější

Celočíselné programování a metody řešení

- Pracuje se s LP relaxací celočíselného programu,
  tj. z celočíselného programu jsou odstraněna omezení požadující řešení z oboru celých čísel a řešíme lineární programy a řešíme lineární programy
- Metoda řezné roviny (cutting plane)
  - generována přídavná lineární omezení (řezné roviny),
    která musí být splněna v celočíselném řešení
  - přídavná omezení zužují množinu přípustných řešení
    při zachování celočíselných řešení
  - řešení LP relaxace celočíselného programu s přídavnými omezeními
  - pokud nenalezeno řešení celočíselné, přidáváme dalsí řezné roviny a opakujeme postup
- Metoda větví a mezí (branch and bound)
  - větvení na rozhodovacích proměnných
    ($x_j = r$ v LP řešení pak přidáme $x_j \leq \lfloor r \rfloor$, $x_j \geq \lceil r \rceil$)
  - lineární programy s přidanými omezeními poskytují hranice (meze)
- Hybridní metody
  - kombinace různých metod
  - např. kombinace metody větví a mezí a řezných rovin (branch and cut)

Ukázka problému: rozvrhování směn

- Směna: množina period, kdy zaměstnanec pracuje
  - často je směna chápána jako množina po sobě jdoucích period
  - př. denní směna 6:00-18:00, noční směna 18:00-6:00
- Problém rozvrhování směn
  - cyklus je dán předem
    - např. 1 den je cyklus, časová jednotka (perioda) je hodina
  - je dáno několik vzorků směn s odlišnou cenou
  - cíl je minimalizovat celkovou cenu

- Problém si popíšeme jako matematický celočíselný program

$$\text{minimalizace} \quad \sum_{j=1}^{n} c_j x_j$$

$$\text{za předpokladu:} \quad \sum_{j=1}^{n} a_{ij} x_j \geq b_i \quad \forall i : 1 \leq i \leq m$$

$$x_j \geq 0 \quad \forall j : 1 \leq j \leq n$$

$$x_j \in \mathbb{Z} \quad \forall j : 1 \leq j \leq n$$

Formulace problému a řešení

- m period (hodin)
  - př. od 10:00 do 21:00
- V periodě $i, i = 1, \ldots, m$ je potřeba b_i zaměstnanců
  - př. 10:00: 3, 11:00: 4, 12:00 6, ...
- n vzorků směn
  - př. 10:00 – 18:00, 13:00 – 21:00, 18:00 – 21:00, ...
- Každý zaměstnanec přiřazen právě jednomu vzorku
- Vzorek směny j je vektor $(a_{1j}, a_{2j}, \ldots, a_{mj})$
  - $a_{ij} = 1$: i je pracovní perioda
  - $a_{ij} = 0$: jinak
  - př. směna 18:00 – 21:00 odpovídá (0,0,0,0,0,0,0,0,1,1,1) pro 10:00 – 21:00
- Cena přiřazení zaměstnance na směnu j: c_j
- x_j: proměnná reprezentující počet zaměstnanců přiřazených ke směně j
- Cíl: minimalizace celkové ceny přiřazených zaměstnanců

- Problém je NP-těžký, nicméně
  - $\mathbb{A}$ má speciální tvar, kde směna je dána jako kontinuální posloupnost 1
  - platí: řešení tohoto lineárního programu je vždy celočíselné

Příklad rozvrhování směn v obchodě

- Obchod otevřen od 10:00 do 21:00
- 5 vzorků (typů) směny

Vzorek	Doba	Hodin	Cena
1	10-18	8	50
2	13-21	8	60
3	12-18	6	30
4	10-13	3	15
5	18-21	3	16

- Požadavky na počet zaměstnanců v obchodě

Hodina	10	11	12	13	14	15	16	17	18	19	20
Počet zaměstnanců	3	4	6	4	7	8	7	6	4	7	8

- Lineární relaxace

$$\overline{c} = (50, 60, 30, 15, 16) \qquad \mathbb{A} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix} \qquad \overline{b} = \begin{bmatrix} 3 \\ 4 \\ 6 \\ 4 \\ 7 \\ 8 \\ 7 \\ 6 \\ 4 \\ 7 \\ 8 \end{bmatrix}$$

- Řešení $(x_1, x_2, x_3, x_4, x_5) = (0, 0, 8, 4, 8)$

Omezující podmínky

13. března 2022

3 Problém splňování podmínek

Omezení (*constraint*)

- Dána
  - množina (doménových) proměnných $Y = \{y_1, \ldots, y_k\}$
  - konečná množina hodnot (doména) $D = D_1 \cup \ldots \cup D_k$

  Omezení c na Y je podmnožina $D_1 \times \ldots \times D_k$
  - omezuje hodnoty, kterých mohou proměnné nabývat současně
- Příklad:
  - proměnné: A,B
  - domény: $\{0,1\}$ pro A $\{1,2\}$ pro B
  - omezení: $A \neq B$ nebo $(A,B) \in \{(0,1),(0,2),(1,2)\}$
- Omezení c definováno na $y_1, \ldots y_k$ je splněno, pokud pro $d_1 \in D_1, \ldots d_k \in D_k$ platí $(d_1, \ldots d_k) \in c$
  - příklad (pokračování): omezení splněno pro $(0,1), (0,2), (1,2)$, není splněno pro $(1,1)$

Problém splňování podmínek (CSP)

Dána

- konečná množina proměnných $V = \{v_1, \ldots, v_n\}$
- konečná množina hodnot (doména) $D = D_1 \cup \ldots \cup D_n$
- konečná množina omezení $C = \{c_1, \ldots, c_m\}$
  - omezení je definováno na podmnožině V

Problém splňování podmínek je trojice (V, D, C)
(constraint satisfaction problem CSP)

Příklad:

- proměnné: A,B,C
- domény: {0,1} pro A {1} pro B {0,1,2} pro C
- omezení: A≠B, B≠C

Řešení CSP

- přiřazení hodnot všem proměnným, které splňuje všechna omezení
- $(d_1, \ldots, d_n) \in D_1 \times \ldots \times D_n$ je řešení (V, D, C)
  - pro každé $c_i \in C$ na $v_{i_1}, \ldots v_{i_k}$ platí $(d_{i_1}, \ldots d_{i_k}) \in c_i$

Filtrace domén

- Příklad:
  - $D_a = \{1, 2\}, D_b = \{1, 2, 3\}$
  - $a < b$

  $\Rightarrow$ hodnota 1 může být z D_b bezpečně vyřazena
- Podmínky se používají aktivně pro odstranění nekonzistencí z problému
  - nekonzistence = hodnota, která nemůže být součástí žádného řešení (nesplňuje nějakou podmínku)
- Tato tzv. filtrace domén je realizována procedurou REVISE, kterou má každá podmínka

Hranová konzistence *(arc consistency AC)*

- Říkáme, že podmínka je hranově konzistentní, pokud
  pro každou hodnotu každé její proměnné
  existuje kombinace hodnot pro další proměnné v podmínce tak, že je podmínka splněna

  Říkáme, že hodnota je podporována/má podporu.
- REVISE procedura pro hranovou konzistenci odstraní z domén proměnných všechny hodnoty, které nemají podporu

- Příklad: A=B+C je hranově konzistentní pro B,C $\in \{1,2\}$, A $\in \{2,3,4\}$
  (hodnota 3 proměnné A má např. podporu (3,1,2) pro (A,B,C))

  A$\neq$B je hranově konzistentní pro A $\in \{0,1\}$, B $\in \{1,2,3\}$

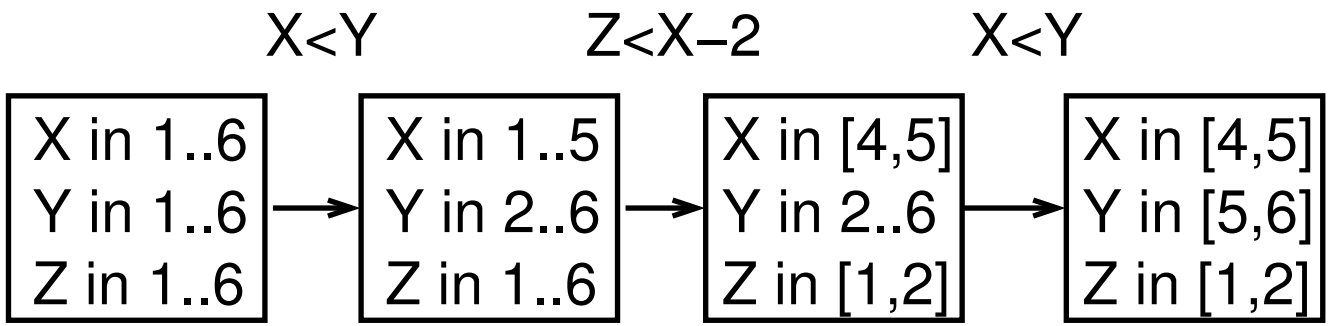
  A=B není hranově konzistentní pro A $\in \{1,2,3\}$, B $\in \{1,2\}$
  (hodnota 3 proměnné A není podporována)

- CSP je hranově konzistentní, pokud jsou všechny podmínky hranově konzistentní.

Zajištění hranové konzistence

- Jak zařídit hranovou konzistenci v CSP?
- Každá podmínka musí být zrevidována
- Příklad:

  X in 1..6, Y in 1..6, Z in 1..6, X<Y,Z<X-2



- Stačí to?

⇒ Nestačí ale zrevidovat každou podmínku pouze jednou

- Revize je potřeba opakovat, dokud se mění doména nějaké proměnné (algoritmus AC-1)

Hranová konzistence prakticky

- Použijeme frontu proměnných se změněnou doménou.
  - uživatelé mohou pro každou podmínku specifikovat, kdy se má provést její revize v závislosti na typu změny domény
- Vychází z algoritmus AC-3 (pracuje s frontou podmínek k revizi)
  a je nazývaný AC-8 (pracuje s frontou proměnných)
- procedure AC-8(V,D,C)
  Q := V
  while Q neprázdná do
  vyber v z Q
  for c ∈ C takovou, že v je omezeno c do
  D' := c.REVISE(D)
  if (libovolná doména v D' prázdná) then return(fail,D')
  Q := Q ∪ {u∈V | $D'_u \neq D_u$}
  D := D'
  return(true,D)
  end AC-8
- Poznámka: do fronty vkládáme jen ty proměnné, které tam ještě nejsou
  - revize se pro ně provede, protože už ve frontě jsou

Příklad: sudoku

			2		5			
	9					7	3	
		2			9		6	
2						4		9
				7				
6		9						1
	8		4			1		
	6	3					8	
			6		8			

Přiřaď prázdným polím čísla tak, že:

čísla odlišná na řádku, ve sloupci a v bloku