10. kapitola. O principu optimálnosti

In: Jaroslav Morávek (author): O dynamickém programování. (Czech). Praha: Mladá fronta, 1973. pp. 71–74.

Persistent URL: http://dml.cz/dmlcz/403802

10. kapitola

O PRINCIPU OPTIMÁLNOSTI

V této závěrečné kapitole si řekneme několik slov o tzv. principu optimálnosti dynamického programování, na kterém se zakládaly všechny konkrétní algoritmy i důkazy probrané v této knížce. Během naší diskuse vysvětlíme rovněž, proč se používá termínu „dynamické programování".

Všimneme si první z našich aplikací dynamického programování, která záležela v určení maxima funkce

$$y = g_1(x_1) + \ldots + g_n(x_n) \tag{19}$$

na množině

$$\left\{(x_1, \ldots, x_n) \;\middle|\; \begin{array}{l} x_1, \ldots, x_n \text{ celá a nezáp.} \\ x_1 + \ldots + x_n = a \end{array}\right\} \tag{20}$$

Každé řešení maximalizačního problému (19), (20) má jistou zajímavou vlastnost, jak ukazuje následující věta.

Věta 23: Nechť $(x_1^{(0)}, \ldots, x_n^{(0)})$ je řešení maximalizačního problému (19), (20) a nechť i, j jsou daná celá čísla, splňující vztahy $1 \leqq i \leqq j \leqq n$. Potom pro $(j - i + 1)$-tici $(x_i^{(0)}, x_{i+1}^{(0)}, \ldots, x_j^{(0)})$ nabývá funkce

$$y = g_i(x_i) + g_{i+1}(x_{i+1}) + \ldots + g_j(x_j) \tag{21}$$

své maximální hodnoty na množině

$$\left\{(x_i, \ldots, x_j) \;\middle|\; \begin{array}{l} x_i, \ldots, x_j \text{ celá, nezáp.} \\ x_i + \ldots + x_j = a(i, j) \end{array}\right\} \tag{22}$$

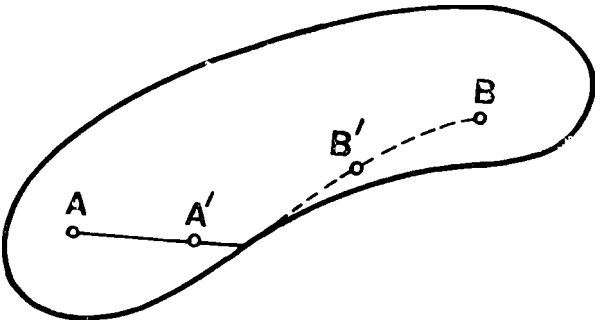
kde $$a(i, j) = x_i^{(o)} + \ldots + x_j^{(o)}$$

Důkaz: Větu dokážeme sporem. Připusťme, že v množině (22) existuje nějaká $(j - i + 1)$-tice $(\hat{x}_i, \ldots, \hat{x}_j)$, pro kterou $g_i(\hat{x}_i) + g_{i+1}(\hat{x}_{i+1}) + \ldots + g_j(\hat{x}_j) > g_i(x_i^{(o)}) + \ldots + g_j(x_j^{(o)})$. Sestrojme nyní n-tici $(\tilde{x}_1, \ldots, \tilde{x}_n)$ tak, že $\tilde{x}_r = x_r^{(o)}$ pro $r = 1, \ldots, i - 1, j + 1, \ldots, n$, a $\tilde{x}_r = \hat{x}_r$ pro $r = i, i + 1, \ldots, j$. Snadno lze ověřit, že $(\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_n)$ je prvkem množiny (20), a že platí $g_1(\tilde{x}_1) + \ldots + g_n(\tilde{x}_n) > g_1(x_1^{(o)}) + \ldots + g_n(x_n^{(o)})$. Získaná nerovnost je však ve sporu s tím, že $(x_1^{(o)}, \ldots, x_n^{(o)})$ je řešením problému (19), (20), což dokončuje důkaz.

Tvrzení věty 23 dáme nyní názornější význam. Problém tvaru (21), (22) nazveme *úsekem problému* (19), (20), a $(j - i + 1)$-tici $(x_i^{(o)}, \ldots, x_j^{(o)})$ nazveme *úsekem řešení* $(x_1^{(o)}, \ldots, x_n^{(o)})$. Nyní lze smysl věty 23 vyjádřit názorně takto: „Úsek řešení problému (19), (20) je řešením úseku tohoto problému."

Zformulovaná jednoduchá vlastnost je právě speciálním důsledkem obecného principu optimálnosti dynamického programování, který umožňuje použití rekurentní metody. Abychom se ještě více přiblížili k jeho neformálnímu smyslu, rozebereme tento geometrický příklad. Na povrchu nějakého (geometrického) tělesa **T** jsou zvoleny dva různé body **A**, **B**. Naším úkolem je spojit oba body nejkratší čarou $\mathscr{C}$, ležící na povrchu tělesa. Čtenář se středoškolskou úrovní matematického vzdělání musí ovšem zformulovanou úlohu chápat pouze intuitivně, neboť pojmy jako čára (křivka), délka čáry apod. se definují s dostačující úrovní přesnosti až ve vysokoškolských přednáškách z matematické analýzy a geometrie.

Zformulovaný problém je v obecném případě značně obtížný, neboť povrch tělesa může být složitá plocha s množstvím „výstupků" a „prolákliň". Za velmi obecných předpokladů o vlastnostech povrchu tělesa **T** lze však dokázat následující, intuitivně velmi názorné tvrzení: Čára $\mathscr{C}$ na povrchu tělesa **T**, spojující **A** s **B** je nejkratší ze všech takových čar právě tehdy, jestliže pro každé dva body **A'** a **B'** ležící na $\mathscr{C}$, je úsek $\mathscr{C}(\mathbf{A}', \mathbf{B}')$ čáry $\mathscr{C}$ ležící mezi **A'** a **B'** sám nejkratší čarou na povrchu **T**, spojující **A'** s **B'** (viz obr. 2).



Obr. 2

Tato zdánlivě jednoduchá vlastnost vyjadřuje výstižně podstatu hlubokého principu optimálnosti, podmiňujícího použití metod dynamického programování. Speciálně extremální úlohy řešené v naší knížce by bylo možné interpretovat jako problémy určení jistých nejkratších čar; vyložené rekurentní metody pak vlastně „slepují" úseky hledané nejkratší čáry, až je nakonec určeno celé řešení.

Úloze o nejkratší čáře lze dát ještě názornou fyzikální interpretaci. Představme si, že po povrchu (reálného fyzikálního) tělesa leze z **A** do **B** konstantní rychlostí brouk, který chce vykonat tuto cestu v nejkratším možném čase, čili — což je v daném případě totéž — po dráze nejkratší délky. Tuto cestu budeme nazývat *optimální*. Slovo „optimální" je latinského původu a jeho význam odpovídá významu českého slova „nejlepší". V daném případě máme konkrétně na mysli, že hledaná cesta má být optimální (nejlepší) ve smyslu minimální doby pohybu (neboli minimální délky dráhy pohybu). Princip optimálnosti lze nyní vyslovit též takto:

Pohyb (v našem konkrétním případě pohyb brouka) probíhá optimálním způsobem právě tehdy, jestliže optimálním způsobem probíhá na každém úseku.

V této obecné formulaci lze principu optimálnosti použít při určování pohybů letadel, raket a družic, optimálních vzhledem k minimální době letu, minimální spotřebě pohonných hmot apod. Lze jej však použít i k určování různých optimálních procesů, např. v technologii, technice a ekonomii, neboť takovéto procesy lze též chápat jako jisté pohyby v některém (obecně vícerozměrném) prostoru. Odtud je též patrný původ názvu „dynamické programování" (slovo „dynamický" odpovídá svým významem českému ekvivalentu „pohybový"). Tím ovšem nechceme říci, že dynamické programování se hodí pouze pro řešení problémů, kde ve zjevné formě vystupuje prostor a čas. (Extremální problémy řešené v naší knížce to ostatně jasně ukazují.) Formulace problémů a metod dynamického programování se obvykle provádí ve zcela abstraktní formě a interpretace pomocí pohybu probíhajícího v prostoru a čase se většinou používá pouze pro názornost a heuristickou cenu.