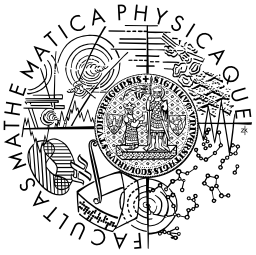

MATEMATICKO-FYZIKÁLNÍ FAKULTA
Univerzita Karlova

BAKALÁŘSKÁ PRÁCE

Samuel Puček

Rozvrhovací optimalizační úlohy ve školství

Katedra pravděpodobnosti a matematické statistiky

Vedoucí bakalářské práce:	doc. RNDr. Ing. Miloš Kopa, Ph.D.
Studijní program:	Matematika
Studijní obor:	Obecná matematika

Praha 2017

Prohlašuji, že jsem tuto bakalářskou práci vypracoval(a) samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V ..... dne ..... Podpis autora

Název práce: Rozvrhovací optimalizační úlohy ve školství

Autor: Samuel Puček

Katedra: Katedra pravděpodobnosti a matematické statistiky

Vedoucí bakalářské práce: doc. RNDr. Ing. Miloš Kopa, Ph.D., Katedra pravděpodobnosti a matematické statistiky

Abstrakt: Tato práce se zabývá teorií celočíselného programování. Po definování základních pojmů uvádí dva algoritmy vhodné pro řešení celočíselných úloh. Prvním z nich je algoritmus větvení a hranic, za ním následuje algoritmus řezných nadrovin. Dále popisuje přiřazovací problém, který je speciálním případem úlohy celočíselného programování. Uvádí maďarskou metodu a vysvětluje její použití na vzorových příkladech. Následuje praktická část práce, která řeší reálný problém z praxe. Cílem této části je najít optimální rozvrh pro první až sedmou třídu vybrané základní školy. Je v ní představeno zpracování vstupních dat, tvorba modelu a samotné řešení. Získané výsledky jsou doprovázeny krátkou diskusí.

Klíčová slova: celočíselné programování, přiřazovací problém, rozvrhovací optimalizace

Title: Scheduling optimization problems in education

Author: Samuel Puček

Department: Department of Probability and Mathematical Statistics

Supervisor: doc. RNDr. Ing. Miloš Kopa, Ph.D., Department of Probability and Mathematical Statistics

Abstract: This work deals with the theory of integer programming. After defining the basic concepts, it presents two algorithms suitable for solving integer problems. Firstly, it talks about the branch and bound algorithm and secondly, it talks about the cutting plane algorithm. Next, it presents an assignment problem, which is a special case of integer programming. The work describes the hungarian method and explains its usage on exemplary examples. The last part of the work solves the real problem from the practice. The aim of this section is to find an optimal schedule for classes one to seven of the selected elementary school. It introduces input data processing, creating a model and the solution. Obtained results are accompanied by a brief discussion.

Keywords: integer programming, assignment problem, scheduling optimization

Názov práce: Rozvrhovacie optimalizačné úlohy v školstve

Autor: Samuel Puček

Katedra: Katedra pravděpodobnosti a matematické statistiky

Vedúci bakalárskej práce: doc. RNDr. Ing. Miloš Kopa, Ph.D., Katedra pravděpodobnosti a matematické statistiky

Abstrakt: Táto práca sa zaoberá teóriou celočíselného programovania. Po definovaní základných pojmov uvádza dva algoritmy vhodné na riešenie celočíselných úloh. Prvým z nich je algoritmus vetvenia a hraníc, za ním nasleduje algoritmus rezných nadrovín. Ďalej predstavuje priraďovací problém, ktorý je špeciálnym prípadom úlohy celočíselného programovania. Uvádza maďarskú metódu a vysvetľuje jej použitie na vzorových príkladoch. Nasleduje praktická časť práce, ktorá rieši reálny problém z praxe. Cieľom tejto časti je nájsť optimálny rozvrh pre prvú až siedmu triedu vybranej základnej školy. Je v nej predstavené spracovanie vstupných dát, tvorba modelu a samotné riešenie. Získané výsledky sú sprevádzané krátkou diskusiou.

Kľúčové slová: celočíselné programovanie, priraďovací problém, rozvrhovacia optimalizácia

Na tomto mieste by som rád poďakoval vedúcemu práce doc. RNDr. Ing. Milošovi Kopovi, Ph.D., za jeho ochotu a čas, ktorý mi venoval. Jeho cenné rady a pripomienky mi veľkou mierou pomohli pri tvorbe samotnej práce.
Ďalej moje poďakovanie patrí mojej rodine a priateľom, ktorí ma počas celého štúdia podporovali.

Obsah

Úvod

V školstve sa každoročne stretávame s problematiku tvorby rozvrhu. Menšie školy často pri rozvrhovaní jednotlivých tried nevyužívajú žiaden softvér. Rozvrhy tvoria ľudia na základe skúseností a intuície. Tento prístup však pri zvyšujúcom počte tried môže zlyhávať. Rozvrh každej triedy totiž musí spĺňať celú radu podmienok od predpísaného počtu hodín jednotlivých predmetov, cez požiadavky učiteľov, až po jeho celkovú vyváženosť.

Celočíselná optimalizácia patrí k rýchlo rozvíjajúcim sa oblastiam matematiky. Svoje uplatnenie nachádza v najrôznejších častiach sveta okolo nás. Medzi motivačné úlohy patria napríklad problém obchodného cestujúceho, optimalizácia portfólia, problém batohu, rozvrhovanie atď. Za zmienku určite stojí kniha Wolsey (1998), ktorá sa venuje výhradne celočíselnej optimalizácii. Namiesto názvu celočíselná optimalizácia budeme ďalej používať názov celočíselné programovanie. V našej práci zhrnieme základné výsledky celočíselného programovania a neskôr sa budeme venovať priraďovaciemu problému. V praktickej časti práce sa budeme snažiť na základe získaných teoretických výsledkov problém tvorby rozvrhu previesť do reči matematiky a následne za pomoci modelovacieho softvéru riešiť. Práci by mal bez väčších problémov porozumieť čitateľ, ktorý absolvoval aspoň základný kurz teórie optimalizácie.

V prvej kapitole sú z počiatku stručne predstavené dôležité poznatky celočíselného programovania. Na záver kapitoly sú uvedené dva algoritmy. Prvým z nich je algoritmus vetvenia a hraníc, za ním nasleduje algoritmus rezných nadrovín. Oba algoritmy sú vhodné na riešenie úloh celočíselného programovania.

V druhej kapitole sa zaoberáme špeciálnym prípadom celočíselného programovania – priraďovacím problémom. Po jeho predstavení nasleduje algoritmus vhodný na výpočet úloh formulovaných ako priraďovací problém.

Posledná kapitola obsahuje praktickú časť práce, kde na reálnych dátach konkrétnej školy vytvoríme optimálny rozvrh pre 1. až 7. triedu. V akom zmysle optimálny je už bližšie popísané v tretej kapitole.

1. Celočíselné programovanie

V následujúcej kapitole zadefinujeme úlohu celočíselného programovania. Uvedieme dôležité vety a tvrdenia o tvare a existencii optimálneho riešenia. Ďalej sa budeme zaoberať špeciálnym prípadom, kedy je už zo zadania úlohy zrejmé, že rýchlym a elegantným spôsobom získame optimálne celočíselné riešenie. V závere kapitoly predstavíme dva univerzálne algoritmy vhodné na riešenie celočíselných úloh. Do detailu rozoberieme všetky kroky výpočtu.

Vzhľadom na to, že v praktickej časti práce si vystačíme s lineárnym modelom, celú teoretickú časť budeme preto sústrediť na lineárne celočíselné programovanie.

1.1 Model celočíselného lineárneho programovania

V tejto časti vychádzame z Pelikán (2001, kapitola 2) a Wolsey (1998, kapitola 1.2).
Úlohou lineárneho programovania (angl. linear programming - LP) rozumieme

$$\begin{gathered}\text{maximalizovať } \mathbf{c}^T\mathbf{x} \\ \text{za podmienok } \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ \mathbf{x} \geq \mathbf{0}, \mathbf{x} \in \mathbb{R}^n,\end{gathered}$$

kde $\mathbf{A}$ je matica typu $m \times n$, $\mathbf{c}$ je n zložkový vektor a $\mathbf{b}$ je vektor s m zložkami. Prvky matice $\mathbf{A}$ a vektorov $\mathbf{c}$ a $\mathbf{b}$ sú ľubovoľné reálne čísla.
Lineárna funkcia $\mathbf{c}^T\mathbf{x}$, ktorú chceme maximalizovať, sa nazýva *účelová funkcia.* Prvky vektoru $\mathbf{x}$ nazývame *rozhodovacie premenné* a prvky vektoru $\mathbf{c}$ sú *koeficienty (ceny)* účelovej funkcie. Sústava nerovníc $\mathbf{A}\mathbf{x} \leq \mathbf{b}$ popisuje *ohraničujúce podmienky* úlohy. Sústava udáva celkom m ohraničujúcich podmienok.

Ak budeme navyše požadovať, aby všetky premenné boli celé čísla, hovoríme o úlohe celočíselného programovania (angl. integer programming - IP), ktorú zapisujeme v tvare

$$\begin{gathered}\text{maximalizovať } \mathbf{c}^T\mathbf{x} \\ \text{za podmienok } \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ \mathbf{x} \geq \mathbf{0},\ \mathbf{x} \in \mathbb{Z}^n.\end{gathered}$$

Úlohou zmiešaného celočíselného programovania (angl. mixed integer programming - MIP) je

$$\begin{gathered}\text{maximalizovať } \mathbf{c}^T\mathbf{x} + \mathbf{d}^T\mathbf{y} \\ \text{za podmienok } \mathbf{A}\mathbf{x} + \mathbf{D}\mathbf{y} \leq \mathbf{b} \\ \mathbf{x} \geq \mathbf{0},\ \mathbf{x} \in \mathbb{Z}^n \\ \mathbf{y} \geq \mathbf{0},\ \mathbf{y} \in \mathbb{R}^n.\end{gathered}$$

V prípade, že všetky premenné sú binárne, tzn. 0 alebo 1, hovoríme o bivalentnej úlohe

$$\begin{gathered}\text{maximalizovať } \mathbf{c}^T\mathbf{x} \\ \text{za podmienok } \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ x \in \{0,1\}^n.\end{gathered}$$

1.2 Tvar a vlastnosti úlohy celočíselného programovania

Rozlišujeme medzi dvomi základnými zápismi úlohy, v súlade s Berežný a Kravecová (2012, kapitola 2.5):

Úlohou (IP) v kanonickom tvare je

$$\begin{gathered}\text{maximalizovať } \mathbf{c}^T\mathbf{x} \\ \text{za podmienok } \mathbf{Ax} \leq \mathbf{b} \\ \mathbf{x} \geq \mathbf{0},\ \mathbf{x} \in \mathbb{Z}^n.\end{gathered}$$

Úlohou (IP) v štandardnom tvare je

$$\begin{gathered}\text{maximalizovať } \mathbf{c}^T\mathbf{x} \\ \text{za podmienok } \mathbf{Ax} = \mathbf{b} \\ \mathbf{x} \geq \mathbf{0},\ \mathbf{x} \in \mathbb{Z}^n.\end{gathered}$$

Veta 1 vychádza z Vety 2.5.1 uvedenej v skriptách Berežný a Kravecová (2012) na strane 35. My uvádzame jej znenie v kontexte celočíselného programovania.

Veta 1. *Kanonický a štandardný tvar úlohy celočíselného programovania sú navzájom ekvivalentné.*

Dôkaz. Dôkaz vyplýva z dole uvedených vzťahov.

□

Na prevod úlohy medzi štandardným a kanonickým tvarom sa používajú tieto úpravy:

1. Ohraničenia v tvare nerovností typu $\geq$ prevedieme na nerovnosti typu $\leq$ vynásobeným príslušného riadku -1. Podobne prevádzame $\leq$ na $\geq$.
2. Ohraničenia v tvare nerovností prevedieme na rovnosti pridaním doplnkových (kĺzavých) premenných. Tie sú nezáporné a v účelovej funkcii majú koeficienty rovné 0.
3. Ohraničenia v tvare rovností môžeme zapísať pomocou dvoch nerovností.
4. *(v úlohe LP)* Každú premennú x bez ohraničení (tzn. $x \in \mathbb{R}$) nahradíme rozdielom dvoch nezáporných premenných x^+ a x^-, tj. $x = x^+ - x^-$.

Poznámka. Stačí, ak budeme uvažovať úlohy, v ktorých chceme účelovú funkciu maximalizovať. Platí

$$\min f(x) = -\max -f(x).$$

Značenie. Úlohu (IP) v kanonickom tvare budeme značiť

$$\text{(IP) } \max \{c^T x : Ax \leq b, x \in \mathbb{Z}^n, x \geq 0\}.$$

Podobne budeme značiť úlohu (IP) v štandardnom tvare.

Uvažujme maximalizačnú úlohu (IP) v kanonickom tvare. Definujeme dva dôležité pojmy v súlade so zavedenými základnými pojmami v Dupačová (1982, str. 49).

Definícia 1. Množinu prípustných riešení úlohy (IP) *definujeme ako*

$$S = \{x \in \mathbb{Z}^n : Ax \leq b, x \geq 0\}.$$

Prvky množiny S nazývame prípustné riešenia.

Definícia 2. *Nech x^* je prípustné riešenie. Povieme, že x^* je* optimálne riešenie, *ak pre každé prípustné riešenie x platí*

$$c^T x^* \geq c^T x.$$

1.3 Formulácia úlohy

Správne formulovanie reálneho problému do reči matematiky je nedielnou súčasťou každej úlohy. Jedná sa o prvotnú časť riešenia, na ktorú neexistujú žiadne algoritmy ani postupy. Riešiteľ musí spracovať problém tak, aby vystihol jeho podstatu, správne určil neznáme premenné, množinu prípustných riešení a na záver účelovú funkciu. Akékoľvek pochybenie v tejto časti riešenia môže viesť k nesprávnym výsledkom. Ani najlepší algoritmus nevie správne vyriešiť úlohu, ktorá je už vo svojej podstate zle definovaná. Pri riešení daného problému riešiteľ vychádza z praktických skúseností a zo znalosti príbuzných modelov.

Formulácie populárnych celočíselných problémov sú často uvádzané v literatúre, viď Berežný a Kravecová (2012, kapitola 2.2), Pelikán (2001, kapitola 3), Wolsey (1998, kapitola 1.3). Čitateľ tu môže nájsť množstvo zaujímavých príkladov.

Na tomto mieste sa ďalej formuláciou reálnych problémov zaoberať nebudeme. V druhej kapitole do detailu predstavíme priraďovací problém.

1.4 Relaxácia a dualita

Pri riešení úloh celočíselného programovania sa často stáva, že nie sme schopný nájsť optimálne riešenie. Preto sa uspokojíme s približným riešením, ktoré sa od optimálneho riešenia líši maximálne o predom zvolenú odchýlku, napr. 1 %. Toto riešenie následne budeme považovať za optimálne, s vedomím chyby, ktorej sme sa dopustili. Takto získané riešenie sa niekedy označuje ako suboptimálne riešenie.

Dôvodov, prečo takýto prístup je v praxi používaný, je hneď niekoľko. Všeobecne platí, že výpočetná zložitosť celočíselných úloh je omnoho väčšia ako pri úlohách lineárneho programovania. Ak by sme tvrdohlavo chceli presné optimálne celočíselné riešenie, mohlo by sa stať, že aj pri moderných výpočetných prostriedkoch by sme naň museli čakať týždne či roky.

V následujúcej časti čerpáme z Pelikán (2001, kapitola 4.6) a Wolsey (1998, kapitola 2).

Okolie optimálneho riešenia je určené takzvanou hornou a dolnou hranicou. Pripomeňme, že uvažujeme úlohu (IP) max $\{c^T x : Ax \leq b, x \in \mathbb{Z}^n, x \geq 0\}$. Jej optimálne riešenie označme symbolom x^*. Hľadáme teda také hodnoty $\underline{x}$ a $\overline{x}$, aby platilo

$$\begin{aligned} \overline{x} &\geq x^*, \\ \underline{x} &\leq x^*, \\ \overline{x} - \underline{x} &\leq \varepsilon, \end{aligned}$$

kde ε je predom zvolená malá kladná konštanta.

Každé prípustné riešenie úlohy (IP) tvorí zároveň dolnú hranicu pre optimálne riešenie. V niektorých úlohách nájdeme prípustné riešenie ľahko. V iných je nájdenie prípustného riešenia zhruba rovnako náročné ako nájdenie optimálneho riešenia. K určeniu hornej hranice zavedieme nový pojem relaxácia.

Definícia 3. *Wolsey (1998, str. 24, Definition 2.1)*
Úlohu (RP) max $\{f(x) : x \in S_R\}$ *nazývame* relaxáciou úlohy (IP) *max* $\{c^T x : x \in S\}$, *ak platí:*

(a) $S \subset S_R$,

(b) $c^T x \leq f(x)$ *pre každé* $x \in S$.

Voľne povedané, relaxovaná úloha má väčšiu množinu prípustných riešení a jej účelová funkcia na množine prípustných riešení pôvodnej úlohy je väčšia alebo rovná pôvodnej účelovej funkcii. Hornú hranicu získame ako optimálne riešenie úlohy (RP).

Tvrdenie 2. *Wolsey (1998, str. 26, Proposition 2.3)*

(1) Ak relaxovaná úloha (RP) nemá prípustné riešenie, potom ani pôvodná úloha (IP) nemá prípustné riešenie.

(2) Nech x^* *je optimálne riešenie úlohy (RP). Ak* x^* *je prípustné riešenie pôvodnej úlohy (IP) a* $c^T x^* = f(x^*)$, *potom* x^* *je optimálne riešenie úlohy (IP).*

Dôkaz. (1) Predpokladáme že množina prípustných riešení S_R úlohy (RP) je prázdna. Z inklúzie $S \subset S_R$ vyplýva, že aj množina prípustných riešení S úlohy (IP) je prázdna.
(2) Označme z = max $\{c^T x : x \in S\}$. Z definície 3 platí $z \leq f(x^*)$. Bod x^* je prípustným riešením úlohy (IP) a teda platí $z \geq c^T x^* = f(x^*)$. Z toho dostávame záver $z = c^T x^* = f(x^*)$.

□

Poznámka. Úlohy IP, kedy k nájdeniu optimálneho riešenia stačí riešiť relaxovanú úlohu sú v praxi ojedinelé. Ďalej sa nimi budeme zaoberať v kapitole 1.5.

Za pozornosť stojí špeciálny typ relaxácie - lineárna relaxácia. Lineárnu relaxáciu získame z úlohy (IP) vypustením podmienky na celočíselnosť.

Definícia 4. *Pelikán (2001, str. 62)*
Uvažujme úlohu (IP) $max\ \{c^T x : Ax \leq b, x \in \mathbb{Z}^n, x \geq 0\}$. *Úlohu (LP)* $max\ \{c^T x : Ax \leq b, x \in \mathbb{R}^n, x \geq 0\}$ *nazývame* lineárnou relaxáciou *úlohy (IP).*

Iná možnosť, ako získať hornú hranicu pre riešenie úlohy IP, je pomocou riešenia duálnej úlohy.

Definícia 5. *Pelikán (2001, str. 66)*
Uvažujme úlohu (IP) $z = max\{c^T x : x \in S\}$, *kde* $S = \{x \in \mathbb{Z}^n : Ax \leq b, x \geq 0\}$. *Úlohu (DP)* $z_D = min\{d^T u : u \in S_D\}$ *nazývame* slabo duálnou *úlohou k úlohe (IP), ak pre každé* $x \in S$ *a pre každé* $u \in S_D$ *platí*

$$c^T x \leq d^T u.$$

V prípade, že $z = z_D$, *hovoríme o* silno duálnej *úlohe.*

Tvrdenie 3. *Wolsey (1998, str. 28, Proposition 2.6)*
Nech (DP) je slabo duálna úloha k úlohe (IP). Potom platí:

(1) Ak je úloha (DP) neohraničená, potom úloha (IP) nemá prípustné riešenie.

(2) Nech pre $x^* \in S$ *a* $u^* \in S_D$ *platí* $c^T x^* = d^T u^*$. *Potom* x^* *je optimálne riešenie úlohy (IP) a* u^* *je optimálne riešenie úlohy (DP).*

Poznámka. Neohraničená úloha je taká úloha, v ktorej účelová funkcia môže nadobúdať ľubovoľne malé hodnoty bez porušenia ohraničujúcich podmienok.

Je dôležité si všimnúť, že každé prípustné riešenie duálnej úlohy nám dáva hornú hranicu. Zatiaľ čo v prípade relaxovanej úlohy hornú hranicu získame až po nájdení optimálneho riešenia.

1.5 Totálna unimodularita

Uvažujme úlohu (IP) $max\ \{c^T x : Ax \leq b, x \in \mathbb{Z}^n, x \geq 0\}$ v kanonickom tvare a k nej lineárnu relaxáciu (LP) $max\ \{c^T x : Ax \leq b, x \in \mathbb{R}^n, x \geq 0\}$. Predpokladajme že pôvodná aj relaxovaná úloha majú optimálne riešenie. Relaxovanú úlohu riešime pomocou metód lineárneho programovania, napr. simplexovým algoritmom. Nastáva prirodzená otázka, či je možné, aby nájdené optimálne riešenie úlohy (LP) bolo celočíselné. Ak sú splnené predpoklady tvrdenia 2, potom nájdené riešenie je optimálnym riešením pôvodnej úlohy. Ďalej budeme vychádzať z Papadimitriou a Steiglitz (1998, kapitola 13.2) a Wolsey (1998, kapitola 3.2).

Definícia 6. *Papadimitriou a Steiglitz (1998, str. 316, Definition 13.1)*
Celočíselná štvorcová matica B sa nazýva unimodulárna, *ak jej determinant je rovný* ± 1.
Celočíselná matica A sa nazýva totálne unimodulárna, *ak determinant každej jej štvorcovej podmatice je rovný +1, -1, alebo 0.*

Poznámka. Matica A nemusí byť nutne štvorcová aby bola totálne unimodulárna.

Tvrdenie 4. *Wolsey (1998, str. 40, Proposition 3.3)*
Úloha (LP) $max\ \{c^T x : Ax \leq b, x \in \mathbb{R}^n, x \geq 0\}$ *má celočíselné optimálne riešenie pre každý celočíselný vektor b taký, pre ktorý je optimálna hodnota účelovej funkcie konečná práve vtedy keď matica ohraničení A je totálne unimodulárna.*

Zistili sme, že totálna unimodularita hrá dôležitú rolu v celočíselnom programovaní. Ako môžeme overiť, či matica má danú vlastnosť?
Overovanie totálnej unimodularity z definície je veľmi pracné. Tento postup je v praxi pre väčšie matice nepoužiteľný. Postačujúcu podmienku pre totálnu unimodularitu uvádza nasledujúce tvrdenie.

Tvrdenie 5. *Papadimitriou a Steiglitz (1998, str. 317, Theorem 13.3)*
Celočíselná matica $A = (a_{i,j})$ je totálne unimodulárna, ak súčasne platia nasledujúce podmienky:

(a) $a_{i,j} \in \{-1, +1, 0\}$,

(b) každý stĺpec obsahuje najviac dva nenulové prvky,

(c) riadky matice A sa dajú rozdeliť do dvoch množín I_1 a I_2 tak, že

- *(i) ak stĺpec má dva nenulové prvky s rovnakým znamienkom, príslušné riadky sú v rôznych množinách,*
- *(ii) ak stĺpec má dva nenulové prvky s rôznym znamienkom, príslušné riadky sú v rovnakých množinách.*

Dôkaz. Dôkaz tvrdenia je uvedený napríklad v Papadimitriou a Steiglitz (1998, str. 317, Theorem 13.3).

□

To, že tvrdenie nám dáva len postačujúcu podmienku ilustruje príklad prevziaty z Wolsey (1998, str. 39, Table 3.2).

Príklad. Nasledujúca matica je totálne unimodulárna, aj keď nespĺňa podmienku (b) v tvrdení 5.

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

1.6 Branch and bound algoritmus

Branch and bound algoritmus môžeme preložiť ako algoritmus vetvenia a hraníc. Patrí k hlavným algoritmom, ktoré sa používajú pri riešení celočíselných úloh. Sú na ňom postavené rôzne ďalšie algoritmy. V tejto časti vysvetlíme na akom princípe funguje. Ďalej čerpáme z Pelikán (2001, kapitola 5.6) a Wolsey (1998, kapitola 7).

Základná myšlienka algoritmu je metóda rozdeľ a panuj. Na začiatku máme komplikovanú úlohu, ktorú chceme riešiť. Úlohu rozdelíme na menšie podúlohy. Tie postupne jednu po druhej vyriešime a spätným zložením získame riešenie pôvodnej komplikovanej úlohy. Ak novo vzniknuté podúlohy sú stále pre nás zložité, postupujeme opätovným delením na menšie a menšie podúlohy, až kým ich nie sme schopný riešiť.

Algoritmus vetvenia a hraníc je iteratívny, v každom kroku porovnáva aktuálne riešenie s doteraz najlepším riešením a pri tom zohľadňuje dolnú a hornú hranicu ktorú priebežne aktualizuje.

Riešime úlohu (IP) max $\{c^T x : x \in S\}$, kde $S = \{x \in \mathbb{Z}^n : Ax \leq b, x \geq 0\}$. Metóda vetvenia a hraníc spočíva v postupnom delení množiny S na jednotlivé menšie podmnožiny S_α, $\alpha \in M$. Na každej z týchto podmnožín hľadáme optimálne riešenie. Ak sa na niektorej podmnožine S_α nepodarí nájsť optimálne riešenie, podmnožinu ďalej rozdelíme na menšie a proces opakujeme. Postupne vytvárame pomyselný strom, ktorého vrcholy sú jednotlivé množiny S_α. Pre každý vrchol stromu riešime pôvodnú úlohu s novou množinou prípustných riešení, tj.

$$(IP)_\alpha \text{ max } \{c^T x : x \in S_\alpha\},\ \alpha \in M.$$

V každom kroku namiesto celočíselnej úlohy riešime lineárne relaxovanú úlohu, tzn. upustíme od podmienky na celočíselnosť. Ak je získané riešenie celočíselné, úlohu ďalej nevetvíme a optimálne riešenie porovnáme s doposiaľ najlepším prípustným riešením, ktoré údáva dolnú hranicu. Na začiatku dolnú hranicu položíme mínus nekonečno.

Ak je získané riešenie neceločíselné, získali sme horný odhad pre danú vetvu stromu. V prípade, že je horší ako aktuálna dolná hranica, vetvu spolu s vrcholom „odrežeme“. Ak je šanca, že v danej vetve stromu by sa mohlo nachádzať optimálne riešenie, úlohu vetvíme nasledovne. Našli sme neceločíselné riešenie $\hat{x} = (\hat{x}_1, ..., \hat{x}_n)$. Existuje teda neceločíselná zložka $\hat{x}_i$, $i \in 1, ..., n$. Vytvoríme dve nové podúlohy úpravou množiny prípustných riešení:

- $S_{\alpha_1} = S_\alpha \cap \{x_i \leq \lfloor \hat{x}_i \rfloor\}$
- $S_{\alpha_2} = S_\alpha \cap \{x_i \geq \lceil \hat{x}_i \rceil\}$

V každom vetvení pridáme do aktuálnej množiny prípustných riešení ďalšie ohraničenie. Z toho je zrejmé, že postupným vetvením hodnota účelovej funkcie zostane nezmenená alebo klesne.

Zapíšeme algoritmus po krokoch. K postupnému spracovávaniu jednotlivých podúloh budeme potrebovať zoznam, do ktorého si budeme ukladať aktívne podúlohy. Najčastejšie sa používa fronta alebo zásobník v závislosti na tom či chceme strom prehľadávať do hĺbky alebo do šírky. Náš zoznam označme $\mathcal{Z}$. Do neho budeme ukladať aktívne podmnožiny S_α. Označme $x^* \in \mathbb{Z}_n$ doposiaľ najlepšie známe prípustné riešenie. Symbolom $z^* \in \mathbb{Z}$ označme hodnotu účelovej funkcie riešenia x^*. Potom z^* je dolná hranica úlohy.

Algoritmus branch and bound

Krok 0: Položme $\mathcal{Z} := \{S\}$, $z^* := -\infty$ a $x^* := \emptyset$.

Krok 1: Ak $\mathcal{Z} = \emptyset$, KONIEC. Optimálne riešenie je x^* a hodnota účelovej funkcie je z^*.
Ak $\mathcal{Z} \neq \emptyset$, vyberieme množinu S_α, $\alpha \in M$, teda $\mathcal{Z} := \mathcal{Z} \setminus \{S_\alpha\}$.

Krok 2: Riešime lineárnu relaxáciu úlohy $(IP)_\alpha$ max $\{c^T x : x \in S_\alpha\}$. Riešenie označme x^0 a hodnotu účelovej funkcie z^0. Ak relaxácia nemá riešenie, pokračujeme *Krokom 1.*

Krok 3: Ak $z^0 \leq z^*$, na množinu S_α zabudneme a pokračujeme *Krokom 1.*

Krok 4: Ak $z^0 > z^*$ a riešenie x^0 je celočíselné, položíme $z^* := z^0$ a $x^* := x^0$. Pokračujeme *Krokom 1.*

Krok 5: Ak $z^0 > z^*$ a riešenie x^0 nie je celočíselné (BÚNO $x_i^0 \notin \mathbb{Z}$), vytvoríme dve nové podmnožiny S_{α_1} a S_{α_2}:

- $S_{\alpha_1} = S_\alpha \cap \{x_i \leq \lfloor x_i^0 \rfloor\}$
- $S_{\alpha_2} = S_\alpha \cap \{x_i \geq \lceil x_i^0 \rceil\}$

Obe podmnožiny S_{α_1} a S_{α_2} pridáme do zoznamu $\mathcal{Z}$, teda $\mathcal{Z} := \mathcal{Z} \cup \{S_{\alpha_1}, S_{\alpha_2}\}$. Pokračujeme *Krokom 1.*

Značenie. Symbolom $\lfloor y \rfloor$ značíme dolnú celú časť čísla y, tzn. najväčšie celé číslo, ktoré je menšie alebo rovné ako y.
Podobne symbolom $\lceil w \rceil$ značíme hornú celú časť čísla w, tzn. najmenšie celé číslo, ktoré je väčšie alebo rovné ako w

Poznámka. Výber množiny zo zoznamu $\mathcal{Z}$ v Kroku 1 sme bližšie nešpecifikovali. Už pred samotným algoritmom sme poznamenali, že ako zoznam môžeme použiť frontu prípadne zásobník. Sú ale aj ďalšie možnosti. V praxi sa často používa mix viacerých prístupov.
Na začiatku sa odporúča, aby sme čo najrýchlejšie našli prvé prípustné riešenie. Teda vyberáme množinu S_α, ktorá bola do zoznamu pridaná ako prvá. Táto metóda sa anglicky nazýva „Depth-First Search“, prípadne „last in, last out“. S prípustným riešením máme dolnú hranicu, na základe ktorej môžeme niektoré vetvy stromu odrezať.
Ďalej sa odporúča spracovávať prioritne uzly, ktoré majú najvyššiu hornú hranicu. Tým zabezpečíme znižovanie globálnej hornej hranice úlohy. Táto metóda nesie názov „Best-Node First“.
Detailnejšia diskusia a porovnanie rôznych prístupov je k nájdeniu v literatúre Pelikán (2001, kapitola 5.6.1) a Wolsey (1998, kapitola 7.4).

Poznámka. Podobne existujú vylepšenia pre výber vetviacej premennej Kroku 5. Opäť ale čitateľa odkážeme na literatúru Pelikán (2001, kapitola 5.6.1).

Algoritmus branch and bound obvykle v praxi pre veľké úlohy nekončí vyprázdnením fronty. Zvyčajne skončí po vyčerpaní predom stanoveného časového limitu, alebo dosiahnutím dostatočne malého rozdielu medzi hornou a dolnou hranicou, viď kapitola 1.4.

1.7 Cutting plane algoritmus

Druhým základným algoritmom na riešenie celočíselných úloh je cutting plane algoritmus, do slovenčiny prekladaný ako algoritmus rezných nadrovín. Tento

názov označuje skupinu viacerých algoritmov. V následujúcej časti predstavíme základný algoritmus, založený na Gomoryho rezoch. Čerpáme z Papadimitriou a Steiglitz (1998, kapitola 14) a Pelikán (2001, kapitola 5.1).

Pripomeňme, že riešime úlohu (IP) max $\{c^T x : x \in S\}$, kde $S = \{x \in \mathbb{Z}^n : Ax \leq b, x \geq 0\}$. Označme $P = \{x \in \mathbb{R}^n : Ax \leq b, x \geq 0\}$. Lineárne relaxovanou úlohou v príslušnom značení rozumieme úlohu (LP) max $\{c^T x : x \in P\}$.

Princíp algoritmu je jednoduchý. V prvom kroku vyriešime relaxovanú úlohu, tzn. hľadáme optimálne riešenie na množine P. Ak je nájdené riešenie celočíselné, výpočet končí, úloha je vyriešená.
Predpokladajme, že sme našli neceločíselné riešenie. Ak by sme do úlohy pridali nové ohraničenia, ktoré by zmenšili množinu P a zároveň by nevylúčili žiadny bod z množiny S, opakovaným riešením relaxovanej úlohy s novou množinou $\tilde{P}$ sa zvyšuje šanca, že nájdené riešenie bude celočíselné. Teda v každom kroku algoritmu pridáme nové lineárne ohraničenie s ohľadom nato, aby množina S ostala nezmenená. Toto ohraničenie sa nazýva Gomoryho rez. Postup opakujeme, kým riešenie relaxovanej úlohy nie je celočíselné.

Máme zadanú úlohu (IP). Riešime relaxovanú úlohu (LP) simplexovým algoritmom. S-tý riadok z výslednej simplexovej tabuľky môžeme zapísať nasledovne:

$$\sum_{j=1}^{n} a_{sj} x_j = b_s. \tag{1.1}$$

Premenné x_j sú nezáporné, preto zrejme platí nerovnosť

$$\sum_{j=1}^{n} \lfloor a_{sj} \rfloor x_j \leq \sum_{j=1}^{n} a_{sj} x_j, \tag{1.2}$$

kde $\lfloor a_{sj} \rfloor$ značí dolnú celú časť čísla a_{sj}.
V úlohe (IP) vyžadujeme, aby riešenie $x = (x_1, \ldots, x_n)$ bolo celočíselné. Preto pre úlohu (IP) je ľavá strana nerovnosti 1.2 celé číslo. Pravú stranu 1.1 môžeme podobne nahradiť dolnou celou časťou

$$\sum_{j=1}^{n} \lfloor a_{sj} \rfloor x_j \leq \lfloor b_s \rfloor. \tag{1.3}$$

Rozdielom 1.1 a 1.3 máme

$$\sum_{j=1}^{n} (a_{sj} - \lfloor a_{sj} \rfloor) x_j \geq (b_s - \lfloor b_s \rfloor).$$

Označme

$$\begin{aligned} r_{sj} &:= a_{sj} - \lfloor a_{sj} \rfloor, \\ r_s^0 &:= b_s - \lfloor b_s \rfloor. \end{aligned}$$

Po preznačení dostávame tzv. Gomoryho nerovnosť

$$\sum_{j=1}^{n} r_{sj} x_j \geq r_s^0. \tag{1.4}$$

Pridaním kĺzavej premennej $\hat{x} \geq 0$ prevedieme nerovnosť 1.4 na rovnosť

$$\sum_{j=1}^{n} r_{sj} x_j - \hat{x} = r_s^0.$$

Prenásobením -1 dostaneme výslednú rovnosť

$$-\sum_{j=1}^{n} r_{sj} x_j + \hat{x} = -r_s^0$$

nazývanú Gomoryho rez.

Poznámka. Číslo r definované ako $r := a - \lfloor a \rfloor$ sa nazýva celočíselný zvyšok (zlomková časť) čísla a. Platí, že $0 \leq r < 1$.

Naším cieľom je pridať Gomoryho rez do pôvodnej úlohy, ako nové ohraničenie. Nasledujúce tvrdenie hovorí o tom, že pridaním Gomoryho rezu sa riešenie úlohy (IP) nezmení.

Tvrdenie 6. *Papadimitriou a Steiglitz (1998, str 329, Lemma 14.1)*
Pridaním Gomoryho rezu do finálnej tabuľky simplexovho algoritmu úlohy (LP) sa žiadne prípustné celočíselné riešenie nevylúči, nová tabuľka je bázická, primárne neprípustná ak $b_s \notin \mathbb{Z}$ a duálne prípustná.

Dôkaz. K nájdeniu v knihe Papadimitriou a Steiglitz (1998, str 329-330). □

V tejto chvíli môžeme zapísať Gomoryho algoritmus po jednotlivých krokoch. V algoritme sa vo všeobecnosti pracuje so zlomkami, zaokrúhľovacie chyby môžu mať fatálne dôsledky. Z tohto dôvodu je zaužívaný aj názov „zlomkový duálny algoritmus“.

Gomoryho algoritmus

Krok 1: Riešime relaxovanú úlohu (LP) simplexovým algoritmom.

Krok 2: Ak je získané optimálne riešenie celočíselné, KONIEC, našli sme optimálne riešenie pôvodnej úlohy (IP).

Krok 3: Existuje zložka optimálneho riešenia úlohy (LP), ktorá je neceločíselná. Predpokladajme, že jej odpovedá s-tý riadok vo výslednej simplexovej tabuľke, ktorý má tvar $\sum_{j=1}^{n} a_{sj} x_j = b_s$. K tomuto riadku vytvoríme Gomoryho nerovnosť, tzn. $\sum_{j=1}^{n} r_{sj} x_j \geq r_s^0$.

Krok 4: Gomoryho nerovnosť prevedieme na rovnosť pridaním kĺzavej premennej $\hat{x} \geq 0$ a prenásobíme -1. Získanú rovnicu $\sum_{j=1}^{n} -r_{sj} x_j + \hat{x} = -r_s^0$ pridáme do simplexovej tabuľky. Rozšírenú tabuľku riešime pomocou duálneho simplexovho algoritmu. Po získaní optimálneho riešenia pokračujeme *Krokom 2.*

Podobne ako branch and bound algoritmus, ani cutting plane algoritmus v praxi pri veľkých úlohách nekončí nájdením optimálneho riešenia, ale nájdením riešenia, ktoré je optimálnemu riešeniu veľmi blízko. Ako sme už spomínali, cutting plane algoritmus pracuje so zlomkami. V praxi sa však zlomky udržujú vo forme čísla, s konečným počtom desatinných miest. Zaokrúhľovacie chyby sa kumulujú, a po čase sa môže stať, že nie sme schopný rozlíšiť, či nájdené riešenie je alebo nie je celočíselné.

2. Priraďovací problém

Priraďovací problém (angl. assignment problem) vo všeobecnosti označuje typy úloh, v ktorých chceme nájsť najlepšie vzájomné priradenie medzi prvkami jednej alebo viacerých množín. Priraďovací problém je špeciálnym prípadom dopravného problému.

Formuláciu lineárneho priraďovacieho problému budeme ilustrovať na praktickom príklade.

2.1 Formulácia problému

Uvažujme dve množiny prvkov, medzi ktorými hľadáme optimálne párovanie. V zásade poznáme dva typy priraďovacieho problému:

- vyvážený,
- nevyvážený.

Pri vyváženom priraďovacom probléme je veľkosť jednej aj druhej množiny prvkov rovnaká. Naopak, nevyvážený priraďovací problém má dve rôzne veľké množiny prvkov. Formuláciu a metódy na hľadanie optimálneho riešenia uvedieme pre vyvážený priraďovací problém. Nevyvážený problém sa prevedie na vyvážený pridaním fiktívnych prvkov do menšej množiny. Vychádzame z Berežný a Kravecová (2012, kapitola 2.2.5) a Pelikán (2001, kapitola 3.3).

Uvažujme n rôznych pracovníkov a n rôznych činností. Každý pracovník môže vykonávať ľubovolnú činnosť avšak s rôznou efektivitou práce. Efektivita každého pracovníka pre danú činnosť je vyjadrená nezáporným koeficientom c_{ij}, kde $i = 1, 2, ..., n$ a $j = 1, 2, ..., n$. Koeficienty c_{ij} sa nazývajú sadzby a tvoria štvorcovú maticu $n \times n$. Úlohou je prideliť pracovníkov k jednotlivým činnostiam tak, aby celková efektivita práce bola maximálna.

Zavedieme bivalentné premenné x_{ij}, tzn. x_{ij} nadobúdajú len hodnôt 0 alebo 1, kde $i = 1, 2, ..., n$ a $j = 1, 2, ..., n$.
Označme

$$x_{ij} = \begin{cases} 1, & \text{ak } i\text{-ty pracovník vykonáva } j\text{-tu činnosť,} \\ 0, & \text{ak } i\text{-ty pracovník nevykonáva } j\text{-tu činnosť.} \end{cases}$$

Podobne premenné x_{ij} tvoria maticu $n \times n$. Každý pracovník vykonáva práve jednu činnosť. Každá činnosť je vykonávaná práve jedným pracovníkom. To má za následok, že v matici je v každom riadku a v každom stĺpci práve jeden prvok rovný 1, zvyšné prvky sú rovné 0.

Poznámka. Matica ohraničení je zrejme totálne unimodulárna. Viac v kapitole 1.5.

Matematická formulácia je nasledovná:

$$max \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \quad i = 1, 2, ..., n,$$

$$\sum_{i=1}^{n} x_{ij} = 1, \quad j = 1, 2, ..., n,$$

$$x_{ij} \in \{0, 1\} \quad pre \quad i, j = 1, 2, ..., n.$$

Priraďovací problém môžeme tiež formulovať ako úlohu optimálneho párovania v bipartitnom grafe. Bipartitným grafom rozumieme graf, v ktorom množina vrcholov V môže byť rozdelená do dvoch skupín V_1 a V_2 tak, že každá hrana grafu má jeden vrchol z množiny V_1 a druhý z množiny V_2. Ak pridelíme jednotlivým hranám „ceny“, optimálne párovanie bude hľadanie maximálnej celkovej ceny.

Ako možný postup pre hľadanie optimálneho riešenia je systematické prechádzanie všetkých možností. V ilustrovanom príklade sme uvažovali n pracovníkov a n činností. Existuje teda $n!$ možných priradení. Počet všetkých možných priradení pre vybrané n je uvedený v tabuľke 2.1. Z nej je zrejmé, že prostým prechádzaním jednotlivých možností sme schopný riešiť len úlohy malého rozsahu.

n	**10**	**100**	**1000**
n!	$3{,}6 \cdot 10^{6}$	$9{,}33 \cdot 10^{157}$	$4{,}02 \cdot 10^{2567}$

Tabulka 2.1: Vybrané hodnoty funkcie n!

Pre hľadanie optimálneho riešenia by sme mohli použiť algoritmy predstavené v prvej kapitole. Pomocou nich by sme síce riešenie získali, mohlo by sa ale jednať len o suboptimálne riešenie.

Podobne ako pre iné špeciálne prípady celočíselného programovania, aj pre priraďovací problém existujú metódy, ktoré hľadajú optimálne riešenie pomerne elegantným spôsobom. Jednu z nich si predstavíme v ďalšej časti.

2.2 Maďarská metóda

Maďarská metóda je formulovaná pre hľadanie optimálneho riešenia minimalizačnej úlohy. Riešenie maximalizačnej úlohy, tzn. prevod na minimalizačnú úlohu a následné aplikovanie maďarskej metódy je bližšie popísané v poznámke za samotným algoritmom. Čerpáme z Hrablik-Chovancová a Sakál (2011, kapitola 4.2).

Uvažujme priraďovací problém predstavený v časti 2.1 s tým rozdielom, že účelovú funkciu chceme minimalizovať.

Na tomto mieste predstavíme jednotlivé kroky algoritmu. Prvý krok algoritmu sa prevedie len raz. Druhý a tretí krok sa môže podľa potreby zopakovať viac krát. Za algoritmom nasleduje krátka diskusia ohľadom jeho správnosti.

Maďarská metóda

Krok 1

(a) V každom riadku matice sadzieb nájdeme minimálny prvok. Hodnotu minimálneho prvku odčítame od každého prvku v danom riadku.

(b) V každom stĺpci matice sadzieb nájdeme minimálny prvok. Hodnotu minimálneho prvku odčítame od každého prvku v danom stĺpci.

Krok 2

(a)
- Počnúc prvým riadkom matice postupne skontrolujeme pre každý riadok, či sa v ňom nachádza práve jeden neprekrytý nulový prvok.
  Ak áno, nulový prvok vyznačíme a stĺpec kolmý na príslušný riadok obsahujúci práve vyznačený nulový prvok prekryjeme čiarou.
  Inak prejeme k ďalšiemu riadku.
- Počnúc prvým stĺpcom matice postupne skontrolujeme pre každý stĺpec, či sa v ňom nachádza práve jeden neprekrytý nulový prvok.
  Ak áno, nulový prvok vyznačíme a riadok kolmý na príslušný stĺpec obsahujúci práve vyznačený nulový prvok prekryjeme čiarou.
  Inak prejdeme k ďalšiemu stĺpcu.

Ak sme neprekryli všetky nulové prvky čiarou, postup opakujeme. Vynechávame riadky (stĺpce), ktoré sú už prekryté.

(b) Ak je počet vyznačených nulových prvkov rovný počtu riadkov matice, KONIEC algoritmu, našli sme optimálne riešenie. Optimálne priradenie pozostáva z vyznačených pozícii nulových prvkov. Optimálna hodnota účelovej funkcie sa určí z pôvodnej matice sadzieb.
Ak je počet vyznačených prvkov menší ako počet riadkov matice, pokračujeme *Krokom 3*.

Krok 3

(a) Nájdeme minimum z neprekrytých prvkov. Túto hodnotu odčítame od všetkých neprekrytých prvkov a pričítame ju k prvkom, ktoré sú prekryté zároveň horizontálnou aj vertikálnou čiarou. Pokračujeme *Krokom 2*.

Matica, ktorý vznikne po prvom kroku algoritmu sa nazýva redukovaná matica sadzieb. Je dôležité si uvedomiť, že optimálne riešenie redukovanej matice sadzieb je zhodné s optimálnym riešením pôvodnej matice sadzieb. Zmena nastala len v hodnote účelovej funkcie.

Poznámka. Riešenie maximalizačnej úlohy prebieha nasledovne. V matici sadzieb nájdeme maximálny prvok, označme ho c^{max}. Vytvoríme novú maticu sadzieb:

$$c^*_{ij} = c^{max} - c_{ij}.$$

Úlohu s novou maticou sadzieb riešime ako minimalizačnú úlohu pomocou maďarskej metódy. Získané riešenie je optimálnym riešením pôvodnej úlohy. Hodnota účelovej funkcie sa určí z pôvodnej matice sadzieb.

Nasleduje príklad formulovaný ako slovná úloha, kde budeme hľadať optimálne párovanie medzi dvoma rovnako veľkými množinami.

Príklad. Taxislužba má k dispozícii 4 voľné taxíky. Dispečer na základe telefonických objednávok má poslať taxíky k 4 rôznym zákazníkom. Jeho cieľom je priradiť taxíky k zákazníkom tak, aby celková doba čakania zákazníkov bola minimálna. Čas potrebný na presun jednotlivých taxíkov k daným zákazníkom je uvedený v tabuľke 2.2.

	A	**B**	**C**	**D**
Taxi 1	2	10	9	7
Taxi 2	15	4	14	8
Taxi 3	13	14	16	11
Taxi 4	4	15	13	9

Tabulka 2.2: Doby presunu taxíkov k zákazníkom

Riešenie. Hľadáme maticu priradení 4×4. Úlohu budeme riešiť pomocou maďarskej metódy. Namiesto matice sadzieb budeme pracovať s tabuľkou:

2	10	9	7
15	4	14	8
13	14	16	11
4	15	13	9

Po riadkovej a stĺpcovej redukcii (prvý krok algoritmu) máme

0	8	2	5
11	0	5	4
2	3	0	0
0	11	4	5

Pokračujeme druhým krokom algoritmu. Po aplikovaní bodu (a) dostaneme

0	8	2	5
11	**0**	5	4
2	3	**0**	0
0	11	4	5

Prevedieme test optimality (Krok 2, bod (b)). Počet vyznačených nulových prvkov je menší ako počet riadkov matice. Optimálne riešenie zatiaľ nemáme. Prejdeme na Krok 3. Po jeho aplikácii máme

$$\begin{matrix} 0 & 8 & 0 & 3 \\ 11 & 0 & 3 & 2 \\ 4 & 5 & 0 & 0 \\ 0 & 11 & 2 & 3 \end{matrix}$$

Opakujeme druhý krok algoritmu. Po prevedení bodu (a) dostaneme

$$\begin{matrix} 0 & 8 & \mathbf{0} & 3 \\ 11 & \mathbf{0} & 3 & 2 \\ 4 & 5 & 0 & \mathbf{0} \\ \mathbf{0} & 11 & 2 & 3 \end{matrix}$$

Počet vyznačených nulových prvkov je 4. Test optimality je úspešný, algoritmus končí. Hľadaná matica priradení má tvar

$$\begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}.$$

Dispečer pošle *Taxi 1* k zákazníkovi C, *Taxi 2* k zákazníkovi B, *Taxi 3* k zákazníkovi D a nakoniec *Taxi 4* k zákazníkovi A. Celková doba čakania zákazníkov je 28.

V našom príklade hodnota účelovej funkcie (celková doba čakania) nie je veľmi dôležitá a sama o sebe nemá žiadny význam. Dôležité však je, že je minimálna.

Uvedieme ešte jeden príklad, podobný predošlému.

Príklad. Taxislužba má k dispozícii 4 voľné taxíky. Dispečer na základe telefonických objednávok má poslať taxíky k 4 rôznym zákazníkom. Tentokrát je jeho cieľom priradiť taxíky k zákazníkom tak, aby celková prejdená vzdialenosť bola minimálna. Vzdialenosti jednotlivých taxíkov a zákazníkov sú uvedené v tabuľke 2.3.

	A	**B**	**C**	**D**
Taxi 1	3	1	1	4
Taxi 2	4	2	2	5
Taxi 3	5	3	4	8
Taxi 4	4	2	5	9

Tabulka 2.3: Vzdialenosti taxíkov a zákazníkov

Riešenie. Opäť budeme hľadať maticu priradení 4×4 pomocou maďarskej metódy. Sadzby si uložíme do tabuľky:

$$\begin{matrix} 3 & 1 & 1 & 4 \\ 4 & 2 & 2 & 5 \\ 5 & 3 & 4 & 8 \\ 4 & 2 & 5 & 9 \end{matrix}$$

Po riadkovej a stĺpcovej redukcii (prvý krok algoritmu) máme

$$\begin{matrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 3 & 4 \end{matrix}$$

Pokračovaním Kroku 2 zistíme, že náš algoritmus zostane zacyklený hneď v prvom bode a teda nenájde optimálne riešenie. Pri pohľade na tabuľku je však zrejmé, že optimálne riešenie existuje, dokonca ich existuje viacero. Jedno z nich je napríklad

$$\begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}.$$

Náš algoritmus, tak ako sme ho uviedli, si s takýmto príkladom neporadí. K vyriešeniu takýchto problémov sú potrebné sofistikovanejšie metódy.

3. Praktická časť

V tejto kapitole využijeme získané teoretické výsledky z prvých dvoch kapitol na riešenie praktickej úlohy.

3.1 Vstupné dáta

Vstupné dáta pochádzajú zo Školy sv. Augustína, Praha 4. Skladajú sa zo študijného plánu jednotlivých tried spolu s úväzkami učiteľov, viď tabuľky 3.1 a 3.2. Ďalej obsahujú požiadavky vybraných učiteľov, ktorí nemôžu učiť každý deň v týždni, viď tabuľka 3.3. Učitelia bez požiadavkov môžu učiť ľubovolný deň.

Súčasťou zadania sú ďalej tieto podmienky:

- žiadne voľné hodiny uprostred vyučovania
- 1. až 4. trieda sa môže učiť 1. až 5. vyučovaciu hodinu
- 5. až 7. trieda sa môže učiť 1. až 6. vyučovaciu hodinu

Rozvrh má byť vyrovnaný tak, aby sa v daný deň neopakoval viac krát rovnaký predmet. Výnimku tvoria predmety český jazyk a výtvarná výchova. Český jazyk môže byť vyučovaný najviac dva krát denne. U výtvarnej výchovy pre triedy s dvojhodinovou týždennou dotáciu (4. až 7. trieda) sa navyše požaduje, aby bola vyučovaná v dvojhodinovom bloku, tzn. dve po sebe idúce hodiny v jeden deň.

Našim cieľom je zhotoviť rozvrh pre každú triedu tak, aby boli splnené všetky vyššie uvedené podmienky a požiadavky.

Predmet	**1. trieda**	**2. trieda**	**3. trieda**	**4. trieda**
Český jazyk	8 LM	7 KA	7 VR	7 MG
Anglický jazyk	1 HO	2 HO	3 AG	3 AG
Španielský jazyk	1 SP	1 SP	1 SP	1 SP
Matematika	4 LM	4 KA	5 VR	4 MG
Náboženstvo	1 PJ	1 PJ	1 PJ	1 PJ
Človek a svet	2 LM	2 KA	2 VR	3 MG
Hudobná výchova	1 LM	1 KL	1 VR	1 KR
Výtvarná výchova	1 LM	1 KA	1 VR	2 MG
Telesná výchova	2 PA	2 PA	2 PA	2 PA
Práca a financie	1 LM	1 KA	1 VR	1 MG
Spolu	22	22	24	25

Pozn: 8 LM znamená, že učiteľ LM učí 8 hodín český jazyk 1. triedu.

Tabulka 3.1: Študijný plán 1. až 4. ročník

Predmet	5. trieda	6. trieda	7. trieda
Český jazyk	7 KL	5 MA	5 MA
Anglický jazyk	3 HO	4 AG	3 AG
Španielský jazyk	1 SP	2 SP	2 SP
Matematika	4 KL	5 SK	5 SK
Informatika a média	1 MC	1 MC	
Náboženstvo	1 PJ	1 PJ	1 PJ
Človek a svet	3 KL		
Dejepis		2 AN	2 AN
Občianska výchova		1 MA	1 MA
Fyzika			2 ZK
Prírodopis		2 AN	1 AN
Zemepis		2 MA	1 MA
Hudobná výchova	1 KL	1 KR	1 KR
Výtvarná výchova	2 VR	2 SM	2 SM
Telesná výchova	2 PA	2 PA	2 PA
Výchova k zdraviu a rodine			1 EV
Práca a financie	1 KA		1 FS
Spolu	26	30	30

Tabulka 3.2: Študijný plán 5. až 7. ročník

	HO	PJ	KA	KL	AG	KR	MC	AN	SM
Po	x			x	x			x	x
Ut	x		x		x		x		
Str		x	x	x					x
Št	x	x	x	x	x	x		x	
Pia			x	x	x				

Pozn: Učiteľ HO môže učiť v pondelok, utorok a štvrtok.

Tabulka 3.3: Požiadavky vybraných učiteľov

3.2 Model

Po spracovaní zadania sme zostavili nasledujúci model. Kto, kedy a aký predmet sa učí je vyjadrené pomocou bivalentných premenných x_{thp}. Prvý index t značí triedu a nadobúda hodnôt 1 až 7. Druhý index h značí deň a hodinu v týždni. Nadobúda hodnôt 1 až 30, viď tabuľka 3.4. Posledný index p značí o aký predmet sa jedná.

Je dôležité poznamenať, že vôbec nemusíme uvažovať špeciálny index pre učiteľov. To je spôsobené tým, že existuje vzájomne jednoznačné priradenie medzi učiteľom a predmetom spolu s triedou. Tento poznatok je významný predovšetkým kvôli výpočetnej zložitosti problému. Momentálne máme počet neznámych 3 570 (počet tried × počet hodín × počet predmetov). Pridaním indexu pre učiteľov nám vzrastie počet neznámych na 67 830 (učiteľov v zadaní máme 19).

	1.	**2.**	**3.**	**4.**	**5.**	**6.**
Po	1	2	3	4	5	6
Ut	7	8	9	10	11	12
Str	13	14	15	16	17	18
Št	19	20	21	22	23	24
Pia	25	26	27	28	29	30

Tabulka 3.4: Hodnoty indexu h

Mohla by nastať otázka, ako chceme vyjadriť požiadavky učiteľov, keď učiteľov v skutočnosti nebudeme uvažovať. Otázka je už zodpovedaná vyššie. Napríklad učiteľ PJ učí náboženstvo v každej triede a môže učiť len v stredu a vo štvrtok. Túto jeho požiadavku premietneme do modelu tak, že každá trieda môže mať náboženstvo len v stredu a vo štvrtok.

Každý rozvrh splňujúci všetky ohraničujúce podmienky je prípustným riešením. Medzi všetkými prípustnými riešeniami sa snažíme nájsť v istom zmysle optimálne riešenie. Aké kritérium optimality zvolíme, je na nás.
V našom prípade sme zvolili kritérium, ktoré síce v zadaní nie je explicitne spomenuté, ale v praxi je považované za akúsi samozrejmosť. Predmety náročnejšie na koncentráciu, ako napríklad jazyky a matematika, musia byť v každý deň vyučované prednostne, tzn. pred predmetmi ako je napríklad hudobná výchova.

Predmety rozdelíme do skupín podľa priority, kde 1. skupina obsahuje predmety s najvyššou prioritou:

1. český jazyk, anglický jazyk, matematika
2. španielský jazyk, človek a svet, dejepis, fyzika, prírodopis, zemepis
3. informatika a média, náboženstvo, práca a financie, občianska výchova, výchova k zdraviu a rodine
4. hudobná výchova, výtvarná výchova, telesná výchova

Prioritu jednotlivých predmetov zachytáva koeficient d_{thp}. Predmety prvej skupiny majú najvyšší koeficient, predmety poslednej skupiny majú najnižší koeficient, viď tabuľka 3.5.

	1. skupina	**2. skupina**	**3. skupina**	**4. skupina**
d_{thp}	100	60	20	1

Tabulka 3.5: Priorita predmetov - hodnoty koeficientu d_{thp}

Prioritu vyučovania zachytáva koeficient c_{thp}. Každá vyučovacia hodina bez ohľadu na deň má priradený koeficient c_{thp}. Ohodnotenie jednotlivých vyučovacích hodín je uvedené v tabuľke 3.6.

	1.	2.	3.	4.	5.	6.
Po	100	80	60	40	20	-1
Ut	100	80	60	40	20	-1
Str	100	80	60	40	20	-1
Št	100	80	60	40	20	-1
Pia	100	80	60	40	20	-1

Tabulka 3.6: Priorita vyučovania - hodnoty koeficientu c_{thp}

Účelová funkcia má následujúci tvar:

$$max \sum_{t} \sum_{h} \sum_{p} c_{thp} d_{thp} x_{thp}.$$

Poznámka. Iný vhodný tvar účelovej funkcie by mohol byť napríklad:

$$max \sum_{t} \sum_{h} \sum_{p} (c_{thp} + d_{thp}) x_{thp}.$$

Ohraničujúce podmienky úlohy pozostávajú z týchto bodov:

- Počty hodín jednotlivých predmetov odpovedajú študijnemu plánu príslušnej triedy:

  $$\sum_{h} x_{thp} = plan(p,t), \quad \forall\, t, p,$$

  kde $plan(p,t)$ je tabuľka hodinových dotácii pre každý predmet a každú triedu v súlade s 3.1 a 3.2.

- Najviac jeden predmet rozvrhnutý v daný čas pre danú triedu:

  $$\sum_{p} x_{thp} \leq 1, \quad \forall\, t, h.$$

- Žiadne voľné hodiny uprostred vyučovania:

  $$\sum_{h=1}^{6} \sum_{p} c_{thp} x_{thp} \geq 280, \quad \forall\, t.$$

  Uvedená podmienka vyjadruje ohraničenie pre „pondelok“. Zvyšné dni analogicky.

- 1. až 4. trieda sa neučí 6. hodinu:

  $$\sum_{t=1}^{4} \sum_{h_6} \sum_{p} x_{th_6p} = 0, \quad kde\ h_6 = 6, 12, 18, 24, 30.$$

- Krytie učiteľov, tzn. daný učiteľ nesmie učiť vo viacerých triedach v rovnaký čas:

  $$x_{1hAJ} + x_{2hAJ} + x_{5hAJ} \leq 1, \quad \forall\, h,$$

kde AJ označuje predmet anglický jazyk. Uvedená podmienka zakazuje, aby učiteľ HO učil vo viacerých triedach v rovnaký čas.
Podmienky pre zvyšných učiteľov s výnimkou LM, MG, ZK, EV, FS sú analogické. Učiteľov LM, MG, ZK, EV a FS ošetrovať nie je potrebné, pretože učia len v jednej triede a každá trieda má podmienku, aby sa učila najviac jeden predmet v daný čas.

- Požiadavky jednotlivých učiteľov v súlade s tabuľkou 3.3:

$$\sum_{h=13}^{18} x_{t_{HO}hp_{HO}} + \sum_{h=25}^{30} x_{t_{HO}hp_{HO}} = 0, \quad \forall t_{HO}, p_{HO},$$

kde t_{HO} označuje triedy a p_{HO} označuje predmety, ktoré učí učiteľ HO. Uvedená podmienka vyjadruje požiadavku, že učiteľ HO nemôže učiť v stredu a v piatok.
Podmienky pre učiteľov PJ, AG, KR, MC, AN a SM sú analogické. Učitelia KA a KL majú zvlášť podmienky:

$$\sum_{h=1}^{6} x_{2hp_{KA}} + \sum_{h=1}^{6} x_{5hPF} = 0, \quad \forall p_{KA},$$

$$\sum_{h=7}^{12} x_{5hp_{KL}} + \sum_{h=7}^{12} x_{2hHV} = 0, \quad \forall p_{KL},$$

kde p_{KA} označuje predmety učiteľa KA, p_{KL} označuje predmety učiteľa KL, PF označuje predmet práca a financie a HV označuje predmet hudobná výchova.

- Najviac jedna hodina denne z každého predmetu pre každú triedu, s výnimkou českého jazyku a výtvarnej výchovy:

$$\sum_{h=1}^{6} x_{th\hat{p}} \leq 1, \quad \forall t, \hat{p},$$

kde $\hat{p}$ označuje všetky predmety okrem českého jazyka a výtvarnej výchovy. Uvedená podmienka vyjadruje ohraničenie pre „pondelok“. Zvyšné dni analogicky.

- Najviac dve hodiny českého jazyku denne pre 1. až 5. triedu:

$$\sum_{h=1}^{6} x_{thCJ} \leq 2, \quad pre\ t = 1, ..., 5.$$

Uvedená podmienka vyjadruje ohraničenie pre „pondelok“. Zvyšné dni analogicky.

- Najviac jedna hodina českého jazyku denne pre 6. a 7. triedu:

$$\sum_{h=1}^{6} x_{thCJ} \leq 1, \quad pre\ t = 6, 7.$$

Uvedená podmienka vyjadruje ohraničenie pre „pondelok“. Zvyšné dni analogicky.

- Dvojhodinový blok výtvarnej výchovy pre 4. až 7. triedu:

$$\sum_{h_p} x_{t_{VV} h_p VV} = 1, \quad \forall t_{VV}, \tag{3.1}$$

kde $h_p = 2, 4, 6, ..., 28, 30$ (len párne hodiny), t_{VV} označuje triedy s dvojhodinovou dotáciou výtvarnej výchovy (tj. triedy 4, 5, 6, 7) a VV označuje predmet výtvarná výchova. Podmienka 3.1 zaručí, že jedna hodina výtvarnej výchovy bude rozvrhnutá v párnu hodinu a druhá hodina výtvarnej výchovy bude rozvrhnutá v nepárnu hodinu. Ďalej

$$\sum_{h_1} x_{t_{VV} h_1 VV} + \sum_{h_4} x_{t_{VV} h_4 VV} \leq 1, \quad \forall t_{VV}, \tag{3.2}$$

kde $h_1 = 1, 7, 13, 19, 25$ (prvé hodiny každý deň), $h_4 = 4, 10, 16, 22, 28$ (štvrté hodiny každý deň). Podmienka 3.2 zaručí bez garancie rovnakého dňa, že v prvú a štvrtú hodinu sa môže vyučovať najviac jedna výtvarná výchova, pre každú triedu s dvojhodinovou dotáciou.
Analogicky sa zapíšu podmienky pre h_1 a h_6, h_2 a h_5, h_3 a h_6. Podmienky pre dvojice h_1 a h_3, h_1 a h_5, h_2 a h_4, h_2 a h_6, h_3 a h_5, h_4 a h_6 neuvažujeme, pretože takéto kombinácie už zakazuje podmienka 3.1. Na záver

$$\sum_{sk_1} x_{t_{VV} sk_1 VV} \leq 1, \quad \forall t_{VV}, \tag{3.3}$$

kde $sk_1 = 1, 3, 5, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30$. Analogicky sa zapíšu ďalšie štyri podmienky pre zvyšné štyri skupiny hodín. Podmienka 3.3 zaručí, že obe hodiny výtvarnej výchovy budú rozvrhnuté v rovnaký deň. Pre lepšiu prehľadnosť skupina hodín sk_1 je vyznačená tučným písmom v tabuľke 3.7.

	1.	**2.**	**3.**	**4.**	**5.**	**6.**
Po	**1**	2	**3**	4	**5**	6
Ut	7	**8**	9	**10**	11	**12**
Str	13	**14**	15	**16**	17	**18**
Št	19	**20**	21	**22**	23	**24**
Pia	25	**26**	27	**28**	29	**30**

Tabulka 3.7: Skupina hodín sk_1

Krátka rekapitulácia. Podmienka 3.1 zabezpečí rozvrhnutie výtvarnej výchovy v jednu párnu a v jednu nepárnu hodinu. Pridaním podmienky 3.2 a k nej analogickým podmienkam zabezpečíme rozvrhnutie výtvarnej výchovy do dvoch susedných hodín bez záruky rovnakého dňa (napr. pondelok 4. hodina a streda 5. hodina). Až podmienka 3.3 spolu s analogickými verziami zaručí rozvrhovanie výtvarnej výchovy v rovnaký deň. Spojením všetkých podmienok získame stanovenú podmienku: dvojhodinový blok výtvarnej výchovy pre štvrtú až siedmu triedu.

- Predmety s dvojhodinovou týždennou dotáciou musia byť rozvrhnuté tak, aby bol medzi nimi aspoň jeden „voľný deň“, s výnimkou výtvarnej výchovy:

$$\sum_{h=1}^{6} x_{thp_2} + \sum_{h=7}^{12} x_{thp_2} \leq 1, \quad \forall t, p_2,$$

  kde p_2 označuje všetky predmety s dvojhodinovou týždennou dotáciou s výnimkou výtvarnej výchovy.
  Uvedená podmienka vyjadruje ohraničenie pre dvojicu„pondelok“ a „utorok“. Podmienky pre zvyšné dvojice „utorok“ a „streda“, „streda“ a „štvrtok“, „štvrtok“ a „piatok“ sú analogické.

- Predmety s trojhodinovou týždennou dotáciou môžu byť rozvrhnuté v troch po sebe idúcich dňoch najviac dva krát:

$$\sum_{h=1}^{6} x_{thp_3} + \sum_{h=7}^{12} x_{thp_3} + \sum_{h=13}^{18} x_{thp_3} \leq 2, \quad \forall t, p_3,$$

  kde p_3 označuje všetky predmety s trojhodinovou týždennou dotáciou.
  Uvedená podmienka vyjadruje ohraničenie pre trojicu „pondelok“, „utorok“, „streda“. Podmienky pre zvyšné trojice „utorok“, „streda“, „štvrtok“ a „streda“, „štvrtok“, „piatok“ sú analogické.

3.3 Riešenie

Na riešenie modelu sme použili modelovací softvér GAMS (General Algebraic Modeling System) s využitím univerzitnej licencie. Bez použitia licencie je sprístupnená len obmedzená verzia softvéru, ktorá je na riešenie príslušného modelu nepostačujúca. K výpočtu bol ďalej použitý notebook Lenovo ThinkPad Edge E531 so 64-bitovým procesorom Intel Core i5-3230M CPU @ 2,60 GHz a operačnou pamäťou 6,00 GB.

Pri výpočte sme použili solver CPLEX. Samotný výpočet trval menej ako jednu sekundu. Hodnota účelovej funckie je 872 912, počet iterácii je 1 697.
Sú splnené všetky požiadavky a ohraničenia, ktoré sme na úlohu kládli. Rozvrhy jednotlivých tried sú k nájdeniu v *Prílohe A* na konci práce. Zdrojový kód je uložený v textovom dokumente „GAMS A“ na priloženom CD.

Po analýze jednotlivých rozvrhov musíme konštatovať, že získané výsledky nie sú úplne uspokojivé. Na prvý pohľad je zarážajúci rozvrh piatej triedy. Pred výpočtom sme očakávali, že piata trieda bude mať šesť vyučovacích hodín len jeden deň v týždni. Opak sa stal pravdou a počet dní v týždni, kedy má piata trieda šesť vyučovacích hodín je maximálny. Získaný rozvrh pôsobí nevyrovnane. Ak by nám takýto rozvrh z pedagogického alebo praktického hľadiska nevyhovoval a požadovali by sme rozvrh s minimálnym počtom dní, kedy sa piata trieda učí šesť hodín, mohli by sme do modelu pridať novú ohraničujúcu podmienku:

- Piata trieda má každý deň aspoň päť vyučovacích hodín:

$$\sum_{h=1}^{6} \sum_{p} x_{5hp} \geq 5.$$

Uvedená podmienka vyjadruje ohraničenie pre „pondelok“. Zvyšné dni analogicky.

Ďalej po preskúmaní rozvrhu šiestej triedy vidíme, že v pondelok sa učí šesť náročnejších predmetov. Na druhú stranu, v stredu sa učí tri ľahšie predmety. Podobných situácii je v riešení viacero.
Pri tvorbe modelu sme predmety rozdelili podľa priority do štyroch skupín. Predmety prvej a druhej skupiny sú považované za náročnejšie predmety, vyžadujúce zvýšenú aktivitu na hodine a väčšiu domácu prípravu. Naopak predmety tretej a štvrtej skupiny nevyžadujú tak veľkú mieru sústredenia na hodine a malú alebo žiadnu domácu prípravu. Pridaním podmienky, aby sa každý deň vyučoval aspoň jeden predmet z tretej alebo štvrtej skupiny docielime väčšiu vyváženosť celého rozvrhu. Do modelu môžeme pridať túto ohraničujúcu podmienku:

- Každá trieda sa učí každý deň aspoň jeden predmet z tretej alebo štvrtej prioritnej skupiny:
$$\sum_{h=1}^{6} \sum_{\hat{p}} x_{th\hat{p}} \geq 1, \quad \forall t,$$
kde $\hat{p}$ označuje predmety tretej a štvrtej prioritnej skupiny.
Uvedená podmienka vyjadruje ohraničenie pre „pondelok“. Zvyšné dni analogicky.

Každá nová ohraničujúca podmienka nesie so sebou riziko neprípustnosti úlohy. Iný možný prístup by viedol cez úpravu koeficientov c_{thp} a d_{thp}. Výsledok by bol podobný.

Pre zaujímavosť sme vyriešili aj model s dodatočne formulovanými podmienkami. Jednotlivé rozvrhy sú k nájdeniu v *Prílohe B* na konci práce. Zdrojový kód je uložený v textovom dokumente „GAMS B“ na priloženom CD.
Pri výpočte sme použili opäť solver CPLEX. Výpočet trval menej ako jednu sekundu. Hodnota účelovej funkcie klesla na hodnotu 872 213, počet iterácii stúpol na 1 738.

Po preskúmaní nových výsledkov môžeme konštatovať, že sa nám podarilo odstrániť zistené nedokonalosti. Novo získané rozvrhy sú celkovo vyváženejšie. Zvyšná časť je venovaná možným vylepšeniam.

Ďalšia požiadavka zo strany školy by mohla byť napríklad delenie tried na jazyky, prípadne telesnú výchovu (chlapci a dievčatá). Naše dáta takúto požiadavku neobsahovali, preto sme ju neuvažovali.

Iná, celkom logická požiadavka by mohla byť, že škola disponuje len jednou telocvičňou, kde prebieha výuka telesnej výchovy. Túto požiadavku sme však splnili bez toho, aby sme ju explicitne formulovali. Všetky triedy učí telesnú výchovu rovnaký učiteľ. Ten však môže učiť v daný čas najviac jednu triedu. Každá trieda sa môže učiť v daný čas najviac jeden predmet. Spolu máme požiadavku na obsadenie telocvične.
Podobne by škola mohla mať aj iné špeciálne triedy, napríklad triedu na výuku jazykov, triedu s dataprojektorom, počítačovú triedu, atď. Výuku predmetu informatika a média, kde by sa dala predpokladať počítačová trieda, ošetrenú máme

podobne ako telesnú výchovu. Ďalším prípadom sme nevenovali pozornosť, nakoľko nemáme informáciu o tom, či škola takýmito triedami disponuje. V prípade, ak by takáto požiadavka vznikla, nebol by žiadny problém príslušné ohraničenie do modelu pridať.

Záver

V tejto práci sme sa zaoberali priraďovacím problémom, čo je špeciálna podskupina úloh celočíselného programovania. Teoretické poznatky z prvých dvoch kapitol sme využili v praktickej časti, ktorá predstavuje len zlomok z celkovej využiteľnosti celočíselného programovania v praxi. Potenciál a škála využitia na riešenie veľkých úloh je obrovská.

V praktickej časti sme sa sústredili na riešenie problému tvorby rozvrhu konkrétnej základnej školy s danými požiadavkami. Na základe vstupných dát sme vypracovali model. Pri jeho tvorbe sme formulovali všetky potrebné ohraničujúce podmienky a zvolili kritérium optimality. Model sme riešili pomocou modelovacieho softvéru GAMS. Pri analýze získaných výsledkov sme objavili nevyváženosť rozvrhu piatej triedy. Rovnako sme zistili že pondelok v šiestej triede je výrazne „náročnejší“ ako streda. Navrhli sme dve možné opatrenia, jedno formou dodatočných ohraničujúcich podmienok, druhé úpravou koeficientov c_{thp} a d_{thp}. Zvolili sme opatrenie formou dodatočných podmienok. Nový model s dodatočnými podmienkami sme opäť riešili pomocou softvéru GAMS a získané výsledky porovnali s predošlými. Nové rozvrhy sú celkovo vyrovnanejšie a z pedagogického hľadiska prijateľnejšie. Na záver sme uviedli možné vylepšenia modelu.

Zoznam použitej literatúry

Berežný, S. a Kravecová, D. (2012). *Lineárne programovanie*. Prvé vydanie. Katedra matematiky a teoretickej informatiky Fakulta elektrotechniky a informatiky Technická univerzita v Košiciach, Košice. ISBN 978-80-553-0910-1.

Dupačová, J. (1982). *Lineární programování*. 1. vyd. Státní pedagogické nakladatelství, Praha.

Hrablik-Chovancová, H. a Sakál, P. (2011). *Operačná analýza časť I.* Prvé vydanie. AlumniPress, Trnava. ISBN 978-80-8096-151-0.

Papadimitriou, C. H. a Steiglitz, K. (1998). *Combinatorial Optimization: Algorithms and Complexity*. II. Title. Dover Publications, Inc., New York. ISBN 0-486-40258-4.

Pelikán, J. (2001). *Diskrétní modely v operačním výzkumu*. První vydání. Professional Publishing, Praha. ISBN 80-86419-17-7.

Wolsey, L. A. (1998). *Integer Programming*. I. Title. Wiley, New York. ISBN 0-471-28366-5.

Príloha A

	1.	2.	3.	4.	5.
Po	CJ	AJ	CJ	PF	HV
Ut	M	CJ	CJ	CS	
Str	M	CJ	CJ	TV	
Št	CJ	M	N	SJ	VV
Pia	CJ	M	CS	TV	

Tabulka 3.8: Rozvrh 1. trieda

	1.	2.	3.	4.	5.
Po	AJ	SJ	HV	TV	
Ut	CJ	CJ	M	CS	
Str	CJ	M	CJ	PF	VV
Št	AJ	CJ	M	TV	N
Pia	CJ	CJ	M	CS	

Tabulka 3.9: Rozvrh 2. trieda

	1.	2.	3.	4.	5.
Po	AJ	M	CJ	CS	TV
Ut	CJ	M	CJ	SJ	
Str	CJ	M	CJ	CS	TV
Št	AJ	CJ	M	N	VV
Pia	CJ	M	AJ	PF	HV

Tabulka 3.10: Rozvrh 3. trieda

	1.	2.	3.	4.	5.
Po	CJ	CJ	AJ	VV	VV
Ut	AJ	CJ	M	CS	TV
Str	M	CJ	CJ	CS	N
Št	M	CJ	AJ	HV	TV
Pia	M	CJ	SJ	CS	PF

Tabulka 3.11: Rozvrh 4. trieda

	1.	2.	3.	4.	5.	6.
Po	M	CJ	AJ	CS	VV	VV
Ut	AJ	SJ	ICT	TV		
Str	M	CJ	CJ	CS	HV	N
Št	CJ	CJ	AJ	M	PF	TV
Pia	M	CJ	CJ	CS		

Tabulka 3.12: Rozvrh 5. trieda

	1.	2.	3.	4.	5.	6.
Po	CJ	AJ	M	SJ	PR	D
Ut	M	AJ	CJ	Z	ICT	TV
Str	CJ	M	SJ	N	VV	VV
Št	M	AJ	CJ	PR	D	HV
Pia	M	AJ	CJ	Z	TV	OV

Tabulka 3.13: Rozvrh 6. trieda

	1.	2.	3.	4.	5.	6.
Po	M	CJ	D	PR	VV	VV
Ut	CJ	M	AJ	F	OV	PF
Str	M	CJ	Z	SJ	VZR	TV
Št	CJ	M	D	AJ	HV	N
Pia	AJ	CJ	M	F	SJ	TV

Tabulka 3.14: Rozvrh 7. trieda

Príloha B

	1.	2.	3.	4.	5.
Po	M	AJ	CS	PF	HV
Ut	CJ	CJ	SJ	TV	
Str	CJ	M	CJ	N	
Št	CJ	M	CJ	TV	
Pia	CJ	M	CJ	CS	VV

Tabulka 3.15: Rozvrh 1. trieda

	1.	2.	3.	4.	5.
Po	AJ	SJ	HV	TV	
Ut	CJ	CJ	M	CS	VV
Str	CJ	M	CJ	TV	
Št	AJ	M	CJ	CS	N
Pia	CJ	M	CJ	PF	

Tabulka 3.16: Rozvrh 2. trieda

	1.	2.	3.	4.	5.
Po	CJ	M	AJ	CS	VV
Ut	M	CJ	AJ	SJ	TV
Str	CJ	CJ	M	CS	PF
Št	AJ	CJ	M	N	
Pia	CJ	M	CJ	TV	HV

Tabulka 3.17: Rozvrh 3. trieda

	1.	2.	3.	4.	5.
Po	AJ	M	CJ	CS	TV
Ut	CJ	M	CJ	VV	VV
Str	CJ	M	CJ	CS	N
Št	M	AJ	CJ	HV	TV
Pia	CJ	AJ	SJ	CS	PF

Tabulka 3.18: Rozvrh 4. trieda

	1.	2.	3.	4.	5.	6.
Po	CJ	M	AJ	CS	HV	
Ut	AJ	SJ	ICT	VV	VV	
Str	CJ	M	CJ	CS	TV	
Št	CJ	CJ	AJ	M	PF	N
Pia	CJ	M	CJ	CS	TV	

Tabulka 3.19: Rozvrh 5. trieda

	1.	2.	3.	4.	5.	6.
Po	M	AJ	CJ	D	PR	TV
Ut	AJ	M	CJ	Z	ICT	OV
Str	CJ	M	N	SJ	VV	VV
Št	M	CJ	D	AJ	PR	HV
Pia	M	CJ	AJ	Z	SJ	TV

Tabulka 3.20: Rozvrh 6. trieda

	1.	2.	3.	4.	5.	6.
Po	CJ	PR	D	M	VV	VV
Ut	CJ	AJ	M	F	OV	TV
Str	M	CJ	SJ	Z	VZR	N
Št	CJ	M	AJ	D	HV	TV
Pia	AJ	M	CJ	SJ	F	PF

Tabulka 3.21: Rozvrh 7. trieda