Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra matematiky

Bakalářská práce

Rozvrhovací úlohy

Plzeň 2024 **Eliška Schwarzová**

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta aplikovaných věd
Akademický rok: 2023/2024

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Eliška SCHWARZOVÁ**
Osobní číslo:	**A21B0046P**
Studijní program:	**B0541A170006 Matematika a finanční studia**
Téma práce:	**Rozvrhovací úlohy**
Zadávající katedra:	**Katedra matematiky**

Zásady pro vypracování

1. Seznamte se s metodami celočíselné optimalizace a přístupy k modelování rozvrhovacích úloh.
2. Proveďte rešerši metod a modelů užívaných v řešení rozvrhovacích úloh a zpracujte přehledně danou problematiku.
3. Vytvořte a implementujte vhodné matematické modely dané úlohy.
4. Proveďte analýzu efektivity jednotlivých modelů a softwarových řešičů na testovacích nebo reálných datech.

Rozsah bakalářské práce: **20-50 stran**
Rozsah grafických prací: **dle potřeby**
Forma zpracování bakalářské práce: **tištěná**

Seznam doporučené literatury:

- Laurence A. Wolsey, George L. Nemhauser: Integer and Combinatorial Optimization, Wiley 1988.
- Michele Conforti, Gérard Cornuéjols, Giacomo Zambelli: Integer Programming, Springer 2014.
- D.de Werra: An Introduction to Timetabling, European Journal of Operational Research Volume 19, Issue 2, February 1985, Pages 151-162.
- Další literatura bude upřesňována průběžně.

Vedoucí bakalářské práce: **Doc. Ing. Roman Čada, Ph.D.**
Katedra matematiky

Datum zadání bakalářské práce: **2. října 2023**
Termín odevzdání bakalářské práce: **22. května 2024**

Západočeská univerzita v Plzni
Fakulta aplikovaných věd
L.S.

Doc. Ing. Miloš Železný, Ph.D.
děkan

Doc. Ing. Marek Brandner, Ph.D.
vedoucí katedry

V Plzni dne 2. října 2023

Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracovala samostatně a výhradně s použitím uvedených zdrojů.

V Plzni dne 22. 5. 2024 .....

Eliška Schwarzová

Poděkování

Ráda bych prostřednictvím této bakalářské práce věnovala poděkování svému vedoucímu panu doc. Ing. Romanu Čadovi, Ph.D., za nápomocný, trpělivý a povzbudivý přístup při vypracování této práce i v průběhu celého studia a za pravidelné konzultace k teoretické i praktické části.

Další poděkování patří zástupci druhého stupně 31. základní školy v Plzni panu Mgr. Vladimíru Hrubému za poskytnutá data k praktické části a jeho rady a nápomocný přístup při zkoumání daného problému.

V neposlední řadě patří mé poděkování spolužákům ze studijního oboru Matematika a finanční studia, přátelům a mé rodině, za jejich podporu, kterou mi dodávali v průběhu celého studia.

Abstrakt

Tato bakalářská práce je věnována teoretické i praktické části rozvrhovacích úloh. V úvodu práce jsou nadefinovány pojmy potřebné při vytváření rozvrhů. Konkrétně se jedná o pojmy z teorie grafů a lineárního programování. Blíže specifikovaný je konkrétní typ celočíselného lineárního programování. Následně se zabýváme přímo úlohou na vytváření rozvrhů. V rámci této problematiky jsou nadefinovány proměnné vstupující do modelu a podmínky, které jsou obecně pro tyto modely stanoveny. V poslední části práce je praktické zpracování rozvrhů za pomoci řešičů v prostředí *AMPL*, k čemuž nám byla poskytnuta reálná data z 31. základní školy v Plzni. Tato škola na zpracování svých rozvrhů využívá aplikaci *aSc Rozvrhy*, kterou si v této části také představíme.

Klíčová slova: teorie grafů, lineární programování, celočíselné lineární programování, rozvrhovací úlohy, *AMPL*, *aSc Rozvrhy*

Abstract

This bachelor's thesis is dedicated to both theoretical and practical parts of timetabling problems. Concepts needed for the construction of timetables are defined in the introductory part of the thesis. These are mainly concepts related to graph theory and linear programming. One particular type of linear programming is specified more closely. Up next, a specific timetabling problem is considered. In relation to this problem, variables and conditions which are typically used in such models are defined. In the last part of the thesis, a practical solution to a timetabling problem using real data, which were provided by 31st elementary school in Pilsen, can be found. This solution is made with the help of *AMPL* environment. Said school uses the *aSc Rozvrhy* application for solving their timetabling problems. This application is briefly introduced in the last part of the thesis as well.

Keywords: graph theory, linear programming, integer linear programming, timetabling progblems, *AMPL*, *aSc Rozvrhy*

Obsah

1 Úvod

Rozvrhovací úlohy jsou obecně známým problémem. Lze si pod tímto pojmem představit klasické vytváření rozvrhových akcí na základních, středních i vysokých školách. To však neznamená, že je tato úloha snadno řešitelná. Právě naopak, jedná se o NP-těžký problém. Vysoká výpočetní složitost je způsobena především tím, že při vytváření je potřeba dodržet řadu omezení a podmínek, které jsou pro rozvrhy stanoveny. Řešení problému je na školách často zjednodušeno již vytvořenými programy, které byly vyvinuty k tomuto účelu.

Pokud bychom se zaměřili na samotné vytváření rozvrhů, můžeme k němu přistupovat z několika možných pohledů. Většina známých metod funguje na principu heuristických a přiřazovacích strategií, kdy se vybírá jedna rozvrhová akce za druhou a přiřazuje se na daný čas do cílového rozvrhu. Samotné algoritmy, které tento rozvrh vytvářejí, fungují několika možnými způsoby. V práci jsou představeny metody přes obarvení grafu z oblasti diskrétní matematiky nebo matematické programování.

V první části práce jsou tyto matematické disciplíny nadefinovány. Zjistíme nejen co je diskrétní graf a jeho vrcholové a hranové obarvení, ale také další specifika této matematické disciplíny. Druhá část bude věnována konkrétnímu typu matematického programování, který je využíván v rozvrhovacích úlohách, lineárnímu programování a jeho celočíselné modifikaci. Od třetí části se zaměříme na samotný rozvrhovací problém.

Prvně si charakterizujeme rozvrhovací úlohu prostřednictvím proměnných, které do tohoto modelu vstupují, a podmínek, které musí daný rozvrh splnit. S těmito podmínkami a proměnnými si následně přiblížíme modely, které využívají předchozích dvou zmíněných strategií.

V poslední části si ukážeme praktickou část vytváření rozvrhů, kdy využijeme reálná data, která nám byla poskytnuta 31. základní školou v Plzni. Pokusíme se sestavit model pro dané vstupní parametry a pomocí řešiče *AMPL* nalézt optimální řešení tohoto rozvrhu. Také si přiblížíme práci s programem *aSc Rozvrhy*, který se v praxi využívá pro vytváření rozvrhů na základních školách. S tímto programem jsme se seznámili v rámci vypracování této práce na zmíněné základní škole.

2 Teorie grafů

Teorie grafů je podstatná část diskrétní matematiky, která má využití v řadě praktických úloh, se kterými se setkáváme v běžném životě. Její výhodou je snadná a intuitivní interpretace [15].

Jedním příkladem praktického využití teorie grafů je úloha obchodního cestujícího (tedy plánování optimální trasy). Vrcholové barvení grafů se například uplatňuje při obarvování politických map (jak už Evropy, tak celého světa). Můžou také znázorňovat elektrické obvody, organické sítě molekul nebo i abstraktní jevy, jako sociální vztahy. Dalším možným příkladem využití v praxi jsou právě rozvrhovací úlohy [9, 15, 24].

Pomocí grafu můžeme znázornit vztah mezi učitelem a třídou (zda daný učitel v této třídě učí či nikoliv), pokud mají společnou výuku, v jaký čas se odehrává, respektive, zda má v danou dobu učitel čas vyučovat nebo je již zaneprázdněn, a další parametry rozvrhů [24]. Abychom však se všemi funkcemi grafů mohli pracovat, nejprve si je nadefinujeme v následujících kapitolách.

2.1 Základní pojmy a definice z teorie grafů

V první řadě si nadefinujeme, co je vůbec graf v kontextu diskrétní matematiky. Nebude se jednat o grafické znázornění průběhu funkce, ale o algebraickou strukturu, která popisuje vztah a vlastnosti určitých subjektů [15].

Definice 2.1 (Graf)**.** *[15] Graf G je uspořádaná dvojice $G = (V, E)$, kde V je neprázdná množina vrcholů a E je množina hran, kde hranou myslíme dvouprvkovou podmnožinu množiny V.*

Jedná se tedy o množinu vrcholů (graficky je můžeme znázornit například jako tečky nebo body), které jsou vzájemně podle některé vlastnosti spojeny hranami (graficky znázorněno jako úsečky). Toto grafické znázornění nazýváme nakreslení grafu [14, 15].

V našem případě budeme uvažovat především konečné grafy, také jinak grafy s konečnou množinou vrcholů, existují však i grafy nekonečné. Počet vrcholů v grafu G budeme značit $|V(G)|$ a počet hran $|E(G)|$ [14, 15].

Vrcholy grafu G obvykle značíme malými písmeny, nebo případně i číslicemi. Hrana, jako dvouprvková podmnožina V, je zadána dvojicí vrcholů, ze kterých vychází. Pokud máme vrcholy u a v, které jsou spojeny hranou, tuto hranu můžeme nazvat $\{u, v\}$ nebo $\{v, u\}$ (mohou být však také pojmenovány malými písmeny nebo číslicemi). Pokud se jedná o neorientovaný graf (tento druh grafu jsme uvažovali doteď), tak na pořadí nezáleží. Jinak tomu bude v případě orientovaného grafu [15].

Definice 2.2 (Orientovaný graf)**.** *[15] Orientovaným grafem rozumíme uspořádanou dvojici $G = (V, E)$, kde V je množina vrcholů a $E \subseteq V \times V$ je množina orientovaných hran.*

Zásadní rozdíl je tedy v tom, že orientovaná hrana má jasně daný začátek i konec, pořadí vrcholů tedy při zápisu hrany nemůžeme zaměnit. Nejedná se již o dvouprvkovou podmnožinu, ale o uspořádanou dvojici vrcholu (u, v) [15].

Další termín, který si zavedeme, a který využijeme v následujících kapitolách, je podgraf.

Definice 2.3 (Podgraf)**.** *[15] Graf H nazveme podgrafem grafu G, jestliže platí $V(H) \subseteq V(G)$ a $E(H) \subseteq E(G)$. Píšeme $H \subseteq G$.*

Je to tedy určitá část grafu. Mohli bychom říct, že podgraf H je jistá podmnožina vrcholů původního grafu G, kterou doplníme o hrany, které mezi těmito vrcholy vedly [15].

Stejně tak budeme potřebovat definici párování v grafu G. Tento pojem využijeme pro zavedení Hallovy věty v následujících kapitolách.

Definice 2.4 (Párování)**.** *[15] Párování M v grafu G je podmnožina nezávislých hran $M \subseteq E(G)$, kde žádné dvě hrany z M nemají společný koncový vrchol. Potom řekneme, že párování M pokrývá vrchol v, jestliže v je koncovým vrcholem nějaké hrany v párování M.*

Z této definice vyplývá, že párování v grafu G je znázorněno tak, že žádné dvě hrany nepřísluší stejnému vrcholu. Vrcholy tedy přiřazujeme do párů, které jsou spojeny jednou hranou a jiné hrany z těchto vrcholů nevedou [15].

Tímto jsme shrnuli jen ty nejzákladnější vlastnosti a definice grafů. V dalších kapitolách se budeme zabývat konkrétními typy grafů a pojmy, které se nám budou hodit při rozboru rozvrhovací úlohy.

2.2 Barvení grafů

Přístup, který se v souvislosti s grafy v úlohách o rozvrhování využívá, je barvení grafů [24]. Máme dva základní typy barvení grafů, vrcholové a hranové [9].

Definice 2.5 (Vrcholové barvení grafů)**.** *[15] Obarvení grafu G pomocí k barev je takové zobrazení*

$$c : V(G) \to \{1, 2, \ldots, k\},$$

ve kterém každé dva vrcholy, které jsou spojeny hranou, budou mít různou barvu, tj. $c(u) \neq c(v)$ pro každou hranu $uv \in E(G)$. Uvedenému obarvení se říká dobré vrcholové barvení grafu.

Přiřazujeme tedy vrcholům barvy, které jsou značeny čísly, podle určitých vlastností. Druhá varianta je, že takto barvy přiřazujeme hranám. Definice je velmi podobná, jen obarvujeme hrany tak, že žádnému vrcholu nesmí příslušet dvě hrany, které by měly stejnou barvu [9, 14, 15].

Dalším důležitým údajem je, kolika barvami jsme schopni tento graf obarvit.

Definice 2.6 (Chromatické číslo)**.** *[14] Řekneme, že graf G je vrcholově k-obarvitelný, jestliže existuje jeho dobré vrcholové obarvení k barvami. Nejmenší číslo k takové, že graf G je vrcholově k-obarvitelný, se nazývá chromatické číslo grafu G a značí se $\chi(G)$. Graf G se nazývá vrcholově k-chromatický, jsou-li jeho vrcholy dobře obarveny k barvami a platí $k = \chi(G)$.*

Takové číslo můžeme mít i v případě hranového obarvení. Udává počet barev, který potřebujeme na dobré hranové obarvení, a nazveme jej chromatický index [14].

Cílem úlohy barvení grafu je použít co nejméně barev. V úloze rozvrhování připadá na každou časovou jednotku (každou vyučovací hodinu) nějaká barva, kterou hraně náležící učiteli a třídě přiřadíme, pokud se potkají v danou vyučovací hodinu. Pokud tak učiníme, nemůžeme již barvu učiteli přiřadit s jinou třídou (případně třídě s jiným učitelem). Máme tedy podchycenou

podmínku, že učitel nemůže být u dvou tříd najednou. V případě rozvrhování tedy využijeme více hranové barvení grafů [14, 15, 24].

2.3 Speciální typy grafů a jejich zobecnění

V teorii rozvrhování se zpravidla využívá některých specifických typů grafů (případně jejich zobecnění), které si v této kapitole nadefinujeme. Do těchto grafů zahrneme bipartitní graf, multigraf, hypergraf a intervalový graf [24].

2.3.1 Bipartitní graf

Bipartitní graf je speciální typ grafu, který rozkládá množinu vrcholů určitým způsobem. Tento způsob bude pro naše rozvrhovací úlohy velice užitečný [15].

Definice 2.7 (Bipartitní graf)**.** *[9] V bipartitním grafu G lze rozdělit množinu vrcholů V na dvě podmnožiny U a W tak, že každá hrana z množiny E má jeden koncový vrchol v podmnožině U a druhý koncový vrchol v podmnožině W. Dvojice U, W se nazývá (vrcholová) bipartita G.*

V případě rozvrhového plánování se nám tento druh grafu velmi hodí. Množina vrcholů U může představovat množinu učitelů na dané škole a množina W může znázorňovat množinu tříd, které se na škole vyskytují. Hrany, které vedou od vrcholů náležícím učitelům, k těm připadajícím třídám, nám znázorňují vztah mezi nimi, tedy také, který učitel v dané třídě učí, a který naopak ne [24].

V souvislosti s bipartitním grafem si ještě zmíníme větu, kterou budeme potřebovat v rozvrhovacích úlohách.

Věta 2.1 (Hallova pro bipartitní graf)**.** *[14] Nechť $G = (V, E)$ je bipartitní graf s partitami X a Y. Pak má G párování obsahující všechny vrcholy partity X právě tehdy, když pro každou množinu $M \subseteq X$ platí, že z ní vedou hrany alespoň k $|M|$ různým vrcholům Y.*

Co tato věta znamená v řeči rozvrhů si uvedeme přímo při jejím využití.

2.3.2 Multigraf

Multigraf je dalším typem grafu, který má praktické využití v rozvrhovacích úlohách. Jeho specifická vlastnost je přípustnost vícenásobných hran. Je možné v něm také vytvářet smyčky, tedy hrany, které vedou z daného vrcholu do něho samého (začátek a konec je v tom samém vrcholu). Formálně graf G zapisujeme $G = (V, E, R)$, kde V je množina vrcholů, E je množina hran a R je matice udávající násobnost hran [6, 14, 15].

Praktické uplatnění při vytváření rozvrhů můžeme nalézt v případě, kdy například jeden učitel vyučuje v dané třídě více předmětů a potřebujeme tuto skutečnost zanést do grafu. Případně pokud učitel v dané třídě má více vyučovacích hodin v jednom časovém období, například dni [24].

2.3.3 Hypergraf

U základního nadefinování grafu jsme často omezeni tím, že hrana může vést jen mezi právě dvěma vrcholy. V přiřazení občas máme učitele, který učí danou třídu v nějaké učebně, nebo učitele, který vyučuje dvě spojené třídy dohromady. V takovém případě by se nám hodilo mít strukturu, která dokáže pojmout více vrcholů, náležící jedné hraně. K tomu slouží hypergraf [24].

Struktura hypergrafu je obecnější než základní definice grafu. Připouští hrany, které nejsou jen dvouprvkové, ale obecné podmnožiny V. Označíme si jej $H = (V, E)$, kde V je opět konečná množina vrcholů. Změní se nám množina hran E. Hranu budeme uvažovat jako podmnožinu některé množiny vrcholů. Obsahuje-li každá hrana E_i právě 2 vrcholy, jedná se o graf, který jsme popsali v definici 2.1 [14, 24].

Toto zobecnění grafu také využijeme v dalších kapitolách této práce.

2.3.4 Intervalové grafy

Intervalové grafy jsou specifické tím, že každý vrchol reprezentuje určitý uzavřený interval na reálné ose (to lze brát i ve smyslu času, kdy každý vrchol reprezentuje nějakou vymezenou dobu). Hrany reprezentují překrytí těchto intervalů, takže mezi vrcholy V a U vede hrana, pokud se intervaly reprezentující tyto vrcholy překrývají [9, 20].

Tím jsme si představili klasické grafické znázornění a zobrazení intervalového grafu. Tuto strukturu je však možné znázornit i v geometrické reprezentaci, kdy na osu reálných čísel $\mathbb{R}$ znázorníme dané intervaly, které reprezentují vrcholy grafu. V případě, že mají tyto intervaly společný průnik, který není prázdná množina, v grafickém znázornění by mezi vrcholy, náležícím těmto intervalům, vedla hrana [9, 20].

U tohoto typu grafů nás také bude zajímat obarvení, konkrétně vrcholové obarvení, našeho intervalového grafu. Pro samotné obarvení platí stejné pravidlo jako v definici 2.5. Budeme se spíše zabývat tím, kolik barev na takové vhodné obarvení budeme potřebovat, tedy jaké bude chromatické číslo intervalových grafů a zda ho můžeme nějak obecněji určit. Než si toto číslo určíme, zadefinujeme si další dva pojmy a to úplný graf a největší klika v grafu.

Definice 2.8 (Úplný graf)**.** *[15] Graf na n vrcholech, kde $n \in \mathbb{N}$, který obsahuje všech $\binom{n}{2}$ hran, se nazývá úplný nebo také kompletní graf a značí se K_n.*

Pokud máme takový graf například na pěti vrcholech, z každého vrcholu vedou čtyři hrany ke zbylým čtyřem vrcholům. Chceme-li takový graf obarvit, budeme tedy pro graf na n vrcholech potřebovat právě n barev k jeho vrcholovému obarvení, jelikož každý vrchol bude potřebovat svou vlastní barvu [15].

Úplný graf se může vyskytnout také jen v určité části grafu, také jinak jako podgraf. Největší takový úplný podgraf, se nazývá největší klika grafu.

Definice 2.9 (Klikovost)**.** *[14] Klikové číslo (nebo také klikovost) grafu G budeme značit $\omega(G)$. Jedná se o počet vrcholů největšího kompletního grafu (kliky), který je podgrafem G.*

Tyto pojmy nám pomohou určit, kolik barev potřebujeme k obarvení intervalového grafu. Ukážeme si to na příkladu z rozvrhování. Budeme uvažovat, že naše intervaly, které reprezentují vrcholy, jsou probíhající vyučovací hodiny v daných třídách. Těmto hodinách chceme přiřadit

učitele (nebudeme uvažovat, které předměty vyučují). Pokud probíhá šest vyučovacích hodin, v šesti třídách současně, potřebujeme minimálně šest učitelů, kteří budou schopni v daný čas vyučovat. Chceme-li tento moment zakreslit do grafové podoby, projeví se jako úplný podgraf na šesti vrcholech. Počet učitelů, které potřebujeme na celý týden, je pak nejvyšší počet vyučovaných hodin, které probíhají ve stejný čas, tedy největší klika v grafu. Pro intervalový graf G tedy platí, že chromatické číslo je rovno klikovosti daného grafu, či-li $\chi(G) = \omega(G)$ [20, 24].

Tímto stylem bychom byli schopni určit, kolik učitelů musí ve škole pracovat, aby byli schopni obstarat výuku pro dané třídy (ve zjednodušeném modelu).

3 Lineární programování a celočíselné lineární programování

Lineární programování, které je jedním z typů matematického programování, se vyvíjelo již od 30. let 20. století [23]. Jeho cílem je nalézt optimální řešení daného problému, proto je to často využívaná metoda [17]. Jinak tomu není ani v rozvrhování, máme více možností jak rozvrh sestavit, a lineární programování je jednou z nich [24]. Nelze však na rozvrhovací úlohy použít obecné lineární programování, ale je zapotřebí k tomu použít speciální případ celočíselného lineárního programování [10], které je sloučením spojité a kombinatorické optimalizace [11]. Tento typ se od standardní metody lineárního programování liší tím, že výsledné proměnné musí nabývat celočíselných hodnot [17]. V následujících kapitolách si nejprve nadefinujeme základní úlohu lineárního programování a metody jejího řešení a poté úlohu celočíselné optimalizace a její řešení. Než však k této problematice přistoupíme, nadefinujeme si dva pojmy, které k tomu budeme potřebovat.

3.1 Dolní a horní celočíselná část

Dolní a horní celočíselná část čísla x jsou důležité matematické pojmy, které budeme v následující práci využívat, proto je zde zmíníme.

Dolní celočíselná část reálného čísla x je definována jako největší celé číslo, které je menší nebo rovno číslu x. Tuto charakteristiku můžeme zapsat následujícím vztahem

$$\lfloor x \rfloor = \max\{m \in \mathbb{Z} : m \leq x\},$$

kde $\lfloor x \rfloor$ je dolní celočíselná část čísla x [13].

Obdobně nadefinujeme horní celočíselnou část čísla x, kterou označíme $\lceil x \rceil$. Tato proměnná znázorňuje nejmenší celé číslo, které je větší nebo rovno proměnné x. Matematický zápis je opět podobný

$$\lceil x \rceil = \min\{m \in \mathbb{Z} : m \geq x\} \text{ [13].}$$

Teď když známe tyto dva pojmy, můžeme se přesunout na úlohu lineárního programování.

3.2 Základní úloha lineárního programování LP

Úloha lineárního programování je významná v úlohách diskrétní optimalizace [19]. Není to však klasické počítačové programování. Zde je slovo programování převzato z amerického slangového výrazu, který vyjadřuje rozvrh či plánování nějaké činnosti. Tento název se uchytil z toho důvodu, že cílem úlohy lineárního programování je nalezení, či naplánování optimálního cíle, či plánu. Slovo lineární již naznačuje, že daná omezení a podmínky úlohy LP jsou zadány formou lineárních funkcí, které zahrnují dané optimalizované veličiny [17]. Tyto podmínky mohou být ve tvaru rovností i nerovností. Je několik typů úlohy LP a lze je řešit řadou způsobů, které si představíme. Rozlišujeme celkem tři základní formulace úlohy LP a to obecnou úlohu, kanonický tvar úlohy a standardní tvar úlohy [19].

3.2.1 Obecný tvar

V obecné úloze LP budeme uvažovat reálnou matici $\mathbf{A}$ o rozměrech $m \times n$. Dále si také potřebujeme zadefinovat $\mathbf{N}$ jako podmnožinu množiny sloupcových indexu $\{1, \ldots, n\}$ matice $\mathbf{A}$ a $\mathbf{M}$ jako podmnožinu množiny řádkových indexů $\{1, \ldots, m\}$ matice $\mathbf{A}$. Množiny $\tilde{\mathbf{N}}$ a $\tilde{\mathbf{M}}$ budou pak značit doplňky těchto podmnožin. Pro formalizování obecného tvaru budeme také potřebovat $\mathbf{a}_i^T$ řádky matice $\mathbf{A}$, kde $(i = 1, \ldots, m)$, reálný vektor $\mathbf{b}$ délky m a reálný vektor $\mathbf{c}$ délky n. Cílem a úkolem obecné úlohy LP je pak nalézt vektor výsledné proměnné $\mathbf{x} \in \mathbb{R}^n$, který maximalizuje výraz $\mathbf{c}^T\mathbf{x}$ (dá se převést i na tvar minimalizace) za podmínek

$$\begin{aligned}
&\max \mathbf{c}^T\mathbf{x}, \\
&\mathbf{a}_i^T\mathbf{x} = b_i \quad \text{pro } i \in \mathbf{M}, \\
&\mathbf{a}_i^T\mathbf{x} \leq b_i \quad \text{pro } i \in \tilde{\mathbf{M}}, \\
&x_j \geq 0 \quad \text{pro } j \in \mathbf{N}, \\
&x_j \in \mathbb{R} \quad \text{pro } j \in \tilde{\mathbf{N}}
\end{aligned}$$ [18, 19, 22].

Tyto podmínky nám stanovují obecný tvar úlohy lineárního programování. Tento tvar je možné převést na kanonický, případně na standardní tvar úlohy. Ve zbylých dvou tvarech jsou podmínky obecné úlohy specifikovány o něco striktněji [25].

Některé zdroje uvažují jako počáteční úlohu minimalizaci tvaru $\mathbf{c}^T\mathbf{x}$ [18], pro definování pojmů to však nemá zásadní rozdíl.

3.2.2 Kanonický tvar úlohy

Jak bylo již zmíněno v kapitole 3.2.1, kanonický tvar úlohy LP lze získat pomocí úprav z obecného tvaru této úlohy a je nadefinován velice podobně [25]. Liší se pouze cíl tohoto tvaru úlohy. Cílem je nalézt vektor $\mathbf{x} \in \mathbb{R}^{\mathbf{n}}$, který maximalizuje výraz $\mathbf{c}^T\mathbf{x}$ za podmínek

$$\begin{aligned}
&\mathbf{Ax} \leq \mathbf{b}, \\
&\mathbf{x} \geq \mathbf{0}.
\end{aligned}$$

Liší se tedy od obecného tvaru tím, že uvažuje jen podmínky typu nerovností, a všechny výsledné hodnoty x_j jsou nezáporné pro každé j [18, 19].

3.2.3 Standardní tvar úlohy

O standardním tvaru úlohy LP můžeme říct na úvod podobné informace jako o kanonickém tvaru. Lze jej získat z obecného tvaru určitou změnou v podmínkách této úlohy [25]. Nadefinování počátečních proměnných je tedy opět stejné, liší se svým cílem. Úkolem je tentokrát nalézt vektor $\mathbf{x} \in \mathbb{R}^{\mathbf{n}}$, který maximalizuje výraz $\mathbf{c}^T\mathbf{x}$ za podmínek

$$\begin{aligned}
&\mathbf{Ax} = \mathbf{b}, \\
&\mathbf{x} \geq \mathbf{0}.
\end{aligned}$$

Tento tvar tedy obsahuje jen podmínky rovností a opět požadujeme, aby všechny výsledné proměnné x_j byly nezáporné pro každé j [19].

3.3 Simplexové metody

V této kapitole se seznámíme s algoritmy, které slouží k řešení úlohy lineárního programování. Tyto metody obecně pracují se standardním tvarem úlohy LP, úloha je tedy ve tvaru

$$\max \{\mathbf{c}^T\mathbf{x} : \mathbf{A}\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0},\ \mathbf{x} \in \mathbb{R}^n\},$$

kde $\mathbf{c} \in \mathbb{R}^n$, $\mathbf{A} \in \mathbb{R}^{m\times n}$ a $\mathbf{b} \in \mathbb{R}^m$ [16, 22]. Pro proveditelnost simplexového algoritmu musí mít matice $\mathbf{A}$ hodnost rovnou m [17].

Dalším krokem je získání bazického řešení.

Definice 3.1 (Bazické řešení)**.** *[19] Je-li $\beta = \{A_{j_1}, A_{j_2}, \dots, A_{j_m}\}$ množina m lineárně nezávislých sloupců matice $\mathbf{A}$, pak vektor $\mathbf{x} \in \mathbb{R}^n$, splňující podmínky*

- $x_j = 0$ pro $A_j \notin \beta$,

- x_k je k-tá složka vektoru $\mathbf{B}^{-1}\mathbf{b}$, kde $\mathbf{B} = [A_{j_i}]_{i=1}^m$ pro $k = 1, \dots, m$,

se nazývá bazické řešení příslušné množině β.

Pokud dokážeme najít bazické řešení, pro které platí, že $x_j \geq 0$, pro každé $j = 1, \dots, n$, nazveme jej přípustné bazické řešení. Některé z těchto přípustných bazických řešení poté může být optimálním řešením úlohy lineárního programování. Stejně tak existuje pro každé bazické řešení vektor $\mathbf{c}$, který maximalizuje výraz $\mathbf{c}^T\mathbf{x}$ [16].

Takto bychom tedy mohli vzít všechny m-prvkové podmnožiny sloupců matice $\mathbf{A}$, nalézt všechna bazická řešení a mezi nimi vybrat to optimální. To je však zdlouhavý postup, proto se zavedl přístup, který upřesňuje přechod od jednoho přípustného bazického řešení ke druhému. Tato metoda se nazývá simplexový algoritmus [19].

3.3.1 Simplexový algoritmus

Opět budeme uvažovat úlohu ve standardním tvaru

$$\begin{aligned} &\max \mathbf{c}^T\mathbf{x}, \\ &\mathbf{A}\mathbf{x} = \mathbf{b}, \\ &x_i \geq 0, \qquad i = 1, \dots, n. \end{aligned}$$

Také budeme předpokládat, že jsme nalezli výchozí přípustné bazické řešení, to však nemusí být rovnou optimální. Nahrazením některé bazické proměnné za jinou přejdeme k dalšímu přípustnému bazickému řešení. Musíme však nejprve rozhodnout, kterou proměnnou z báze odstraníme, a kterou na její místo dosadíme [17, 19].

Uvažujme s-tou nebazickou proměnnou, které přiřadíme cenu c_s. Ta by nahradila hodnotu $c_1a_{1s} + \dots + c_ma_{ms}$. Pokud by tato suma byla vyšší, než cena proměnné c_s, tak se tuto nebazickou proměnnou vyplatí zahrnout do báze [19]. To nám tedy řeší, kterou proměnnou do báze zahrnout, ještě musíme vyřešit otázku vyjmutí složky z báze.

Pokud v dané bázi uplatníme x_s jednotek s-té proměnné, tak v dané směsi proměnných budeme mít $x_1 - a_{1s}x_s$ jednotek první bazické proměnné a tak dále, až $x_m - a_{ms}x_s$ jednotek m-té bazické proměnné, kde jsou všechny výrazy nezáporné. Platí tedy

$$x_s \leq \frac{x_i}{a_{is}} \quad \forall(i = 1, \dots, m).$$

Chceme do báze použít takovou proměnnou, která se v ní vyskytne v co největším množství, proto vyjmeme z báze tu proměnnou, která má výraz $\frac{x_i}{a_{is}}$ minimální kladný [19].

Přípustné bazické řešení je optimální, pokud již nemáme žádnou další proměnnou, kterou bychom chtěli do báze zahrnout [19].

3.3.2 Dvoufázová metoda

Vysvětlili jsme si, jak získat pomocí simplexového algoritmu optimální bazické řešení. Předpokladem této metody však bylo, že jsme již nalezli nějaké výchozí přípustné bazické řešení. To je součástí dvoufázové metody.

Budeme k tomu potřebovat zavést novou pomocnou veličinu $y = [y_1, \dots, y_m]^T$, dále si zadefinujeme nulový vektor $\mathbf{o} \in \mathbb{R}^n$ a vektor $\mathbf{v} = [1, 1, \dots, 1]^T \in \mathbb{R}^m$. S těmito nově zavedenými proměnnými vyřešíme pomocnou úlohy

$$\begin{aligned} &\max [\mathbf{o}^T | \mathbf{v}^T] \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix}, \\ &[\mathbf{A}|\mathbf{I}] \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} = \mathbf{b}, \\ &\begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} \geq \mathbf{0}. \end{aligned}$$

Při nalezení optimálního řešení $\begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix}$ je veličina $\mathbf{x}$ výchozím přípustným bazickým řešením pro řešení simplexového algoritmu. Nalezení tohoto bazického řešení se obvykle nazývá první fází a řešení simplexového algoritmu s touto výchozí proměnnou je druhou fází. Z tohoto důvodu se daná metoda nazývá dvoufázová. [19].

3.4 Dualita

Řešení úlohy lineárního programování v obecném tvaru (ale také v kanonickém a standardním tvaru) maximalizuje výraz $\mathbf{c}^T\mathbf{x}$. Je však možné řešit úlohu LP pro minimalizaci. Vztah mezi minimalizační a maximalizační úlohou se nazývá dualita [23]. Základní myšlenka duality spočívá v tom, že ke každé takzvané primární úloze, lze sestavit úlohu duální. Tyto dvě úlohy mají velkou spojitost a v momentě vyřešení jedné máme zpravidla řešení i druhé úlohy [21].

V různé literatuře se liší, zda primární úloha je právě maximalizační, nebo minimalizační, v této práci budeme uvažovat jako primární úlohu tu, ve které výraz $\mathbf{c}^T\mathbf{x}$ maximalizujeme [18].

3.4.1 Primární úloha LP

Pro zavedení pojmu primární úlohy LP využijeme kanonický tvar. Ten v případě primární úlohy vypadá následovně

$$\begin{aligned} &\max \mathbf{c}^T\mathbf{x}, \\ &\mathbf{A}\mathbf{x} \leq \mathbf{b}, \\ &\mathbf{x} \geq \mathbf{0}. \end{aligned}$$

Opět platí, že matice $\mathbf{A}$ je typu $m \times n$, vektor $\mathbf{b} \in \mathbb{R}^m$ a vektory $\mathbf{x}$, $\mathbf{c} \in \mathbb{R}^n$. Pokud chceme tuto úlohu vyřešit, převedeme ji na tvar duální úlohy. Někdy je minimalizační úloha snazší na vyřešení, a proto se jeví jako výhodnější, vyřešit v první řadě duální úlohu a pomocí ní určit řešení úlohy primární [17, 18, 21, 22].

3.4.2 Duální úloha LP

Duální úlohu si, stejně jako předchozí primární úlohu, zavedeme přes kanonický tvar úlohy

$$\begin{aligned} &\min \mathbf{b}^T\mathbf{y}, \\ &\mathbf{A}^T\mathbf{y} \geq \mathbf{c}, \\ &\mathbf{y} \geq \mathbf{0}. \end{aligned}$$

Navíc se nám přidal vektor neznámých $\mathbf{y} \in \mathbb{R}^m$ [18, 21].

Tato duální úloha omezuje původní primární úlohu shora, tedy každé řešení $(y_1, \ldots, y_m)$, které je přípustné, nám poskytuje horní odhad pro maximum účelové funkce primární úlohy, pomocí kterého ji můžeme snáze vyřešit [17, 22].

3.4.3 Duální simplexový algoritmus

Duální simplexový algoritmus, podobně jako ten klasický, hledá přípustná bazická řešení, která jsou výsledkem úlohy lineárního programování. Toto výsledné řešení však neoptimalizuje zadanou primární úlohu, ale vytvořenou duální úlohy k této primární. Výsledek je však zpravidla přípustný jak pro duální, tak i primární úlohu [25].
Stejně jako u simplexového algoritmu převedeme duální úlohu na kanonický tvar. Poté nejprve postupujeme stejně jako dříve, rozdíl nastává, pokud hledáme, kterou složku báze vyjmout, a kterou do ní přidat [25].

Duální simplexovou metodu využíváme v případě, kdy je některá složka vektoru $\mathbf{b}$ záporná (tj. $b_i < 0$). V případě, kdy jsou všechny složky nezáporné, můžeme využít klasický simplex. Jestliže pro některou složku vektoru $\mathbf{y}$ platí $y_j < 0$, vybereme tento index j. Příslušná proměnná s tímto indexem pak bude odstraněna z původní báze. Máme-li více takových záporných hodnot y_j, vybereme tu nejmenší. Jsou-li všechny hodnoty y_j nezáporné, nemáme již žádnou proměnnou, kterou bychom chtěli z báze odstranit a algoritmus v tomto bodě končí a nalezené bazické řešení je optimální [25].

Při výběru proměnné, kterou přidáme do báze, se opět zaměříme na veličinu $\frac{y_i}{a_{is}}$. V případě, že je některá proměnná $y_i < 0$, vybereme tu veličinu, kde je zlomek $\frac{y_i}{a_{is}}$ nejmenší. V případě, že žádná proměnná vektoru y_i není záporná, primární úloha není přípustná a algoritmus končí.

Takto úlohu řešíme, dokud nedojdeme k jednomu ze závěrů, kterým daný algoritmus skončí [25].

3.5 Celočíselné lineární programování ILP

Celočíselné lineární programování je speciálním případem úlohy LP, kdy výsledné proměnné musí nabývat celočíselných hodnot. Je sice těžší na řešení, protože podmínka celočíselnosti je poměrně striktní, avšak je hojně využíváno v řadě praktických problémů [11], například

u rozvrhovacích úloh, na které se tato práce zaměřuje [10]. Nemusí se však jednat jen o rozvrhování ve školách, ale také binární proměnné, které přiřazují úlohy ke strojům. Dalším možným uplatněním je optimalizace portfolia (celočíselné počty akcií, modelování fixních transakčních nákladů, atd.) nebo také úlohy rozvozu, kde se plánuje cesta vozidla [4].

V následujících kapitolách si představíme nadefinování a přístup k celočíselnému lineárnímu programování a také jeho metody řešení.

3.5.1 Formulace obecné úlohy ILP

Úlohu obecného celočíselného lineárního programování můžeme naformulovat následujícím způsobem

$$\begin{aligned} &\max \mathbf{c}^T\mathbf{x}, \\ &\mathbf{A}\mathbf{x} \leq \mathbf{b}, \\ &\mathbf{x} \geq \mathbf{0} \qquad \mathbf{x} \in \mathbb{Z}^n, \end{aligned}$$

kde vektor $\mathbf{b}$ je řádkový vektor délky m, $\mathbf{A}$ je matice o rozměrech $m \times n$ a $\mathbf{c}$ a $\mathbf{x}$ jsou sloupcové vektory délky n. Vektor $\mathbf{x}$ obsahuje proměnné hledaného řešení. Potom množina řešení $S := \{\mathbf{x} \in \mathbb{Z}_+^n : \mathbf{A}\mathbf{x} \leq \mathbf{b}\}$ je hledaným výsledkem obecné úlohy celočíselného lineárního programování [5, 11, 18].

Tato úloha uvažuje za přípustné řešení jen ta, která mají všechny složky celočíselné, tedy $x_j \in \mathbb{Z},\ \forall(j = 1, \ldots, n)$ [5, 11, 18].

3.5.2 Formulace smíšené úlohy MILP

V případě obecné úlohy ILP jsme předpokládali celočíselné složky přípustného řešení. Další možnou variantou je připustit výskyt některých reálných složek. Takovou úlohu bychom mohli naformulovat následujícím způsobem

$$\begin{aligned} &\max \mathbf{c}^T\mathbf{x} + \mathbf{h}^T\mathbf{y}, \\ &\mathbf{A}\mathbf{x} + \mathbf{G}\mathbf{y} \leq \mathbf{b}, \\ &\mathbf{x} \geq \mathbf{0} \qquad\qquad \mathbf{x} \in \mathbb{Z}^n, \\ &\mathbf{y} \geq \mathbf{0} \qquad\qquad \mathbf{y} \in \mathbb{R}^p, \end{aligned}$$

kde $\mathbf{c}$ je vektor délky n, $\mathbf{h}$ vektor délky p, $\mathbf{A}$ je matice s rozměry $m \times n$, $\mathbf{G}$ je matice $m \times p$ a $\mathbf{b}$ je vektor délky m. Obvykle předpokládáme, že složky $\mathbf{c}$, $\mathbf{h}$, $\mathbf{A}$, $\mathbf{G}$ a $\mathbf{b}$ jsou racionální. Vektory proměnných $\mathbf{x}$ o délce n a $\mathbf{y}$ o délce p obsahují proměnné hledaného řešení [4, 5].

U proměnných x_j uvažujeme jen nezáporné celočíselné hodnoty, zatímco u proměnných y_j všechny nezáporné reálné hodnoty. Připouštíme tedy přítomnost neceločíselné složky řešení. Vždy budeme předpokládat, že existuje alespoň jedna celočíselná proměnná čili že $n \geq 1$. Obecné ILP je speciálním případem smíšeného celočíselného lineárního programování, kde $p = 0$. Množina řešení $S := \{(\mathbf{x}, \mathbf{y}) \in \mathbb{Z}_+^n \text{ x } \mathbb{R}_+^p : \mathbf{A}\mathbf{x} + \mathbf{G}\mathbf{y} \leq \mathbf{b}\}$ se nazývá množina smíšeného celočíselného lineárního programování [4, 5].

3.5.3 Smíšené binární LP

Množina smíšeného 0, 1 lineárního programování představuje takovou formulaci množiny, ve které nabývají celočíselné proměnné jen hodnot 0 nebo 1:

$$S := \{(\mathbf{x}, \mathbf{y}) \in \{0,1\}^n \times \mathbb{R}^p_+ : \mathbf{A}\mathbf{x} + \mathbf{G}\mathbf{y} \leq \mathbf{b}\}.$$

Tato množina představuje přijatelná řešení úlohy smíšeného 0, 1 lineárního programování [4, 5].

3.5.4 Relaxace

Nalezení řešení úlohy celočíselného lineárního programování je obecně velmi těžké. Jedním možným přístupem, který se využívá, je nalezení relaxace této dané úlohy. To se řeší numericky snadněji než původní úloha a poskytuje nám dobrou aproximaci. Uvažujme množinu smíšených celočíselných řešení $S \subseteq \mathbb{Z}^n \times \mathbb{R}^p$, jejíž lineární relaxací je množina

$$P' := \{(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^n \times \mathbb{R}^p : \mathbf{A}'\mathbf{x} + \mathbf{G}'\mathbf{y} \leq \mathbf{b}'\}.$$

Formulace relaxace lineárního programování je úloha $\max \{\mathbf{c}^T\mathbf{x} + \mathbf{h}^T\mathbf{y} : (\mathbf{x}, \mathbf{y}) \in P'\}$ [5].

Pro smíšenou celočíselnou lineární množinu řešení S existuje přirozená lineární relaxace, která je definována:

$$P_0 := \{(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^n_+ \times \mathbb{R}^p_+ : \mathbf{A}\mathbf{x} + \mathbf{G}\mathbf{y} \leq \mathbf{b}\}[5].$$

Tuto formulace množiny získáme vyřazením podmínky celočíselnosti složky $\mathbf{x}$ ve formulaci smíšené úlohy ILP. Přirozená relaxace lineárního programování je dána předpisem:

$$\max \{\mathbf{c}^T\mathbf{x} + \mathbf{h}^T\mathbf{y} : (\mathbf{x}, \mathbf{y}) \in P_0\} \text{ [5]}.$$

3.6 Metody pro řešení celočíselného lineárního programování

V této kapitole si přiblížíme dvě základní metody pro řešení úloh celočíselného lineárního programování a jejich následnou modifikaci. Tyto základní metody jsou metoda mezí a větví a metoda řezné roviny. Jsou založeny na jednoduchých myšlenkách, ale jsou velice významné při řešení. Jsou také základním vybavením softwarů pro řešení celočíselného programování. Dále si představíme společnou modifikaci těchto dvou metod a to metodu mezí a řezů. [5]

V následujících kapitolách budeme formulaci smíšené úlohy z kapitoly 3.5.2 značit zkratkou MILP. Pro připomenutí:

$$\text{MILP} : \ \max \{\mathbf{c}^T\mathbf{x} + \mathbf{h}^T\mathbf{y} : (\mathbf{x}, \mathbf{y}) \in S\},$$

kde $S := \{(\mathbf{x}, \mathbf{y}) \in \mathbb{Z}^n_+ \times \mathbb{R}^p_+ : \mathbf{A}\mathbf{x} + \mathbf{G}\mathbf{y} \leq \mathbf{b}\}$. Pro usnadnění budeme však uvažovat, že řešení úlohy MILP je konečné. Takové optimální řešení si označíme $(\mathbf{x}^*, \mathbf{y}^*)$ a optimální hodnotu úlohy MILP z^*. Tyto tři veličiny jsou neznámé, které hledáme [5].

Nechť $(\mathbf{x}^0, \mathbf{y}^0)$ a z^0 jsou optimální řešení a optimální hodnota přirozené relaxace lineárního programování

$$\max \{\mathbf{c}^T\mathbf{x} + \mathbf{h}^T\mathbf{y} : (\mathbf{x}, \mathbf{y}) \in P_0\},$$

kde P_0 je přirozená lineární relaxace S. Budeme předpokládat, že jsme schopni pomocí řešiče vypočítat hledané hodnoty $(\mathbf{x}^0, \mathbf{y}^0)$ a z^0. Jelikož $S \subseteq P_0$, víme také, že $z^* \leq z_0$. Je-li $\mathbf{x}^0$ celočíselný vektor, potom $(\mathbf{x}^0, \mathbf{y}^0) \in S$ a také platí $z^* = z_0$. V tomto případě máme hledané řešení MILP [5].

Teď se již zaměříme na samotné metody řešení úlohy celočíselného lineárního programování.

3.6.1 Metoda větví a mezí

Metoda větví a mezí je jednou ze základních metod, které umí řešit úlohy s celočíselnými výsledky a nemusí to být zpravidla jen ty lineární. Principem této úlohy, je rozdělit si daný problém na menší podúkoly, které se snáze řeší. V těchto jednotlivých úkolech hledáme maximum účelové funkce [4, 11].

V prvním kroku vyřešíme relaxaci původní úlohy bez podmínky celočíselnosti. Pokud nám vyjdou celočíselné výsledné proměnné, jedná se rovnou o optimální přijatelné řešení. Pokud ne, vybereme takový index j, který splňuje $1 \leq j \leq n$ a zároveň x_j, tedy prvek výsledného vektoru z relaxační úlohy s indexem j, není celočíselný. Tuto proměnou si pro další úlohu upravíme dvojím způsobem

$$S_1 := S \cap \{(\mathbf{x}, \mathbf{y} : x_j \leq \lfloor x_j \rfloor\}, \quad S_2 := S \cap \{(\mathbf{x}, \mathbf{y}) : x_j \geq \lceil x_j \rceil\}.$$

Také jinak, rozdělíme si úlohu na nové dvě, kde v jedné x_j zaokrouhlíme dolu a v druhé nahoru. Budeme tedy zkoumat, která z celočíselných veličin, které jsou reálnému výsledku z relaxace nejblíže, vyhovuje více. Tuto hodnotu přidáme k počátečním podmínkám původní úlohy a spočítáme znovu relaxaci pro dvě nové úlohy

$$\text{MILP}_1 : \max \{\mathbf{c}^T\mathbf{x} + \mathbf{h}^T\mathbf{y} : (\mathbf{x}, \mathbf{y}) \in S_1\}, \quad \text{MILP}_2 : \max \{\mathbf{c}^T\mathbf{x} + \mathbf{h}^T\mathbf{y} : (\mathbf{x}, \mathbf{y}) \in S_2\}.$$

Tím dospějeme k rozvětvení původní úlohy na dvě [4, 5, 11].

Bylo by však neoptimální pracovat neustále se všemi větvemi. Proto si vždy určíme, která větev má lepší nalezené celočíselné řešení, tedy to s největší hodnotou účelové (objektivní) funkce, a s touto větví budeme pracovat. Tím nastavíme určitá omezení na větve [4, 5].

Graficky se tato metoda dá přehledně zpracovat do stromu, kde rozvětvujeme výsledky pomocí horních a dolních celočíselných mezí a naopak ukončujeme větve danými omezeními [4].

Tato metoda obvykle na větších úlohách nepracuje příliš dobře. Dochází buď k vyčerpání zadaného časového limitu, nebo výsledek vystoupí z přípustné množiny řešení [4].

3.6.2 Metoda řezné roviny

Opět budeme uvažovat smíšenou celočíselnou úlohu lineárního programování

$$\text{MILP} : \max + \{\mathbf{c}^T\mathbf{x} + \mathbf{h}^T\mathbf{y} : (\mathbf{x}, \mathbf{y}) \in S\},$$

kde $S := \{(\mathbf{x}, \mathbf{y}) \in \mathbb{Z}_+^n \times \mathbb{R}_+^p : \mathbf{A}\mathbf{X} + \mathbf{G}\mathbf{y} \leq \mathbf{b}\}$ a nechť P_0 je přirozená lineární relaxace S, která byla nadefinována v kapitole 3.5.4. Poté z_0 je ideální hodnota a $(\mathbf{x}^0, \mathbf{y}^0)$ je optimální řešení této

relaxace [5].

Principem metody řezné roviny je nalézt nerovnost $\alpha x_i + \gamma y_j \leq \beta$, která je splněna v každém bodě oblasti S (tedy pro každé i a j), avšak není splněna pro $(\mathbf{x}_0, \mathbf{y}_0)$, které není celočíselné. Pokud bychom našli výsledek z množiny celých čísel, metoda rovnou končí. Pokud tomu tak není, utvoříme z $(\mathbf{x}_0, \mathbf{y_0})$ řeznou rovinu, která bude oddělena od množiny přípustných řešení S [5, 23].

Výsledek dostaneme za pomoci rekurzivního postupu. V i-tém kroku vyřešíme relaxaci dané úlohy, pro začátek uvažujeme $i = 0$

$$\max \{\mathbf{c}^T\mathbf{x} + \mathbf{h}^T\mathbf{y} : (\mathbf{x}, \mathbf{y} \in P_i\}.$$

Pokud výsledné řešení $(\mathbf{x}^i, \mathbf{y}^i)$ připadá do množiny celočíselných řešení S, nemusíme úlohu dále řešit. Pokud však ne, musíme nalézt řeznou roviny $\alpha\mathbf{x} + \gamma\mathbf{y} \leq \beta$, která odděluje $(\mathbf{x}^i, \mathbf{y}^i)$ od množiny S. Pokud ji nalezneme, nastavíme novou relaxační množinu jako

$$P_{i+1} := P_i \cap \{(\mathbf{x}, \mathbf{y}) : \alpha\mathbf{x} + \gamma\mathbf{y} \leq \beta\}$$

a opakujeme znovu rekurzi s $i = i+1$. V novém kroku si tedy upravíme množinu pro relaxaci tak, že neobsahuje námi dříve nalezené reálné optimum, máme tedy možnost najít jiné, to celočíselné. Tyto řezy, kterými se dostáváme k optimálnímu řešení, nazýváme Gomoryho řezy [2, 5, 23].

3.6.3 Metoda mezí a řezů

Metoda mezí a řezů, jak už název vypovídá, je kombinací předchozích dvou zmíněných metod. Ze začátku postupujeme stejně jako při metodě větví a mezí. Rozdíl nastává v jednotlivých větvích výčtového stromu. Tento rozvětvený strom můžeme omezit Gomoryho řezy, které jsme prováděli v metodě řezné roviny [5, 11].

Tato metoda byla v roce 2014 nejúspěšnější metodou řešení celočíselných programů, protože spojuje výhody předchozích dvou metod [2, 5].

4 Rozvrhovací úlohy — základní proměnné a podmínky

Do této chvíle jsme si přiblížili matematické nástroje, které budeme na vytváření rozvrhových akcí a na definování samotného problému potřebovat, teď se již zaměříme přímo na rozvrhy.

V následujících kapitolách si řekneme, s jakými proměnnými musíme rozvrhy sestavovat a jaké podmínky musí být pro tyto proměnné splněny. Také jinak s jakými daty při sestavování rozvrhů pracujeme.

Rozvrhovací úlohy se mohou týkat jak základních, středních, tak vysokých škol. V našem modelu budeme však obecněji předpokládat základní a střední školy, jelikož u vysokých škol se již požadavky na rozvrhy mírně liší (tyto odlišnosti si uvedeme později při výčtu podmínek rozvrhů). V praktické části se budeme později snažit narozvrhovat data, která nám byla poskytnuta ze základní školy v Plzni, proto se zaměříme přímo na jejich specifika.

4.1 Základní proměnné

Nejzákladnějšími proměnnými našeho rozvrhovacího modelu, které uvažujeme pro libovolnou školu (základní nebo střední), budou množina jednotlivých žákovských tříd a množina učitelů na škole. Množinu tříd si označíme $\mathbf{C} = \{c_1, \ldots, c_m\}$, kde m je počet tříd a množinu učitelů si zadefinujeme jako $\mathbf{T} = \{t_1, \ldots, t_n\}$, kde n bude značit počet učitelů na škole. Každý prvek těchto množin má svůj vlastí popis a charakteristiku (u tříd jsou to zpravidla požadavky na výuku, zatímco i učitelů jsou to pracovní povinnosti) [1, 8, 24].

Mezi základní informace, které si nese každý učitel množiny **T**, jsou aprobace (předměty, které vyučuje) a velikosti svého úvazku (počet hodin, které týdně odučí). Prvky množiny **C** obsahují již více informací [24].

Mezi základní vlastnosti prvků z množiny **C** patří počet vyučovacích hodin daného předmětu, který má být ve třídě odučen. Případně může být již uvedeno, který učitel z množiny **T** vyučuje daný předmět. V našem modelu budeme uvažovat, že již víme, které hodiny mají společné třída c_i a učitel t_j. Pro uchování těchto informací nám bude sloužit matice $\mathbf{R} = (r_{ij})$ o velikosti $m \times n$, kde prvek r_{ij} značí již zmíněnou vlastnost (počet hodin, které mají společné třída c_i a učitel t_j) [24].

Další proměnnou, kterou si zavedeme, je množina vyučovacích hodin p, která nese informaci o počtu vyučovacích hodin, které mohou být odučeny v rámci jednoho dne a jak dlouho trvá jedna vyučovací hodina [8, 24].

Mezi základní proměnné patří také seznam učeben, ve kterých může probíhat výuka. Náš rozvrhovací model budeme však stavět bez této proměnné. V reálném problému se s učebnami ve většině případů nerozvrhuje. Škola disponuje dostatkem prostor pro výuku a učitelé si volí učebny sami až po sestavení rozvrhu [24].

Poslední proměnná, kterou si v tuto chvíli zavedeme, je x_{ijk}, která nabývá dvou hodnot (0 nebo 1). Hodnoty 1 nabývá, pokud se třída c_i a učitel t_j potkají při vyučovací hodině v čase k. Hodnoty 0 nabývá v ostatních případech [24].

Na těchto základních proměnných bude stát celý model rozvrhovacích úloh [24].

4.2 Základní podmínky

Pokud máme základní data s charakteristikou daných proměnných, které jsou potřebné pro sestavení rozvrhů, můžeme se již zabývat samotnými podmínkami, které tyto rozvrhy musí splnit. Těchto podmínek je řada.

Pro počáteční nadefinování rozvrhovacích úloh, a podmínek které splňují, si problém rozdělíme na denní a týdenní plánování.

4.2.1 Denní plánování

Pro počáteční zjednodušení problému budeme uvažovat, že potřebujeme vytvořit rozvrh jen v rámci jednoho dne. Aby bylo možné daná data narozvrhovat, musí proměnné x_{ijk} splňovat následující podmínky

$$\sum_{k=1}^{p} x_{ijk} = r_{ij} \qquad \forall(i = 1, \dots, m;\ j = 1, \dots, n), \tag{1}$$

$$\sum_{j=1}^{n} x_{ijk} \leq 1 \qquad \forall(i = 1, \dots, m;\ k = 1, \dots, p), \tag{2}$$

$$\sum_{i=1}^{m} x_{ijk} \leq 1 \qquad \forall(j = 1, \dots, n;\ k = 1, \dots, p), \tag{3}$$

$$x_{ijk} = 0 \text{ nebo } x_{ijk} = 1 \qquad \forall(i = 1, \dots, m;\ j = 1, \dots, n;\ k = 1, \dots, p) \text{ [24]}. \tag{4}$$

Rovnice 1 udává správnost dat. Počet hodin, které má učitel t_j ve třídě c_i v různých časech ($k = 1, \dots, p$) je roven prvku r_{ij} v matici R. Podmínky 2 a 3 zaručují, že učitel t_j nebude v jednu vyučovací hodinu učit ve více třídách najednou (nebo naopak, že jedna třída nebude mít více než jednu rozvrhovou akci během dané vyučovací hodiny). Podmínku 4 jsme již zmiňovali [24].

Při známosti výše uvedených proměnných lze sestavit bipartitní multigraf $G = (\mathbf{C}, \mathbf{T}, \mathbf{R})$. Uzly tohoto grafu tvoří jednotlivé prvky množiny $\mathbf{C}$ a množiny $\mathbf{T}$, tedy třídy a učitelé. Vrcholy c_i a t_j jsou spojeny r_{ij} rovnoběžnými hranami (tedy takovým počtem hran, kolik mají společných hodin). Pokud každé hodině odpovídá jedna barva, hledáme takové přiřazení jedné z p barev ke každé hraně z G, že žádné dvě sousední hrany nemají stejnou barvu (tím zaručíme, že jeden učitel nevyučuje dvě hodiny v ten samý čas). Platí tedy, že proměnná x_{ijk} bude nabývat hodnoty 1, pokud nějaká hrana $[c_i, t_j]$ má barvu k. Řešení rovnic 1—4 tedy existuje, pokud jsou splněny následující podmínky

$$\sum_{i=1}^{m} r_{ij} \leq p \qquad \forall(j = 1, \dots, n),$$

$$\sum_{j=1}^{n} r_{ij} \leq p \qquad \forall(i = 1, \dots, m).$$

To nám říká, že se žádný učitel (ani žádná třída) nesmí účastnit více než p vyučovacích hodin [24].

4.2.2 Týdenní plánování

To byl však zjednodušený model denního plánování, který se v praxi příliš nevyužívá. Z toho důvodu si přiblížíme problém týdenního plánování.

Pokud řešíme rozvrh jen v rámci jednoho dne, zadáváme vyučovací hodinu přímo do něj a naše matice R představuje všechny vyučovací hodiny, které musí být v tento den naplánovány. Ve školství se však více řeší týdenní plánování, kdy musí být výuka přiřazena k některému dni v týdnu. Matice R pak představuje všechny takové hodiny, které se musí konat během týdne. Pro týdenní plánování si musíme zadefinovat některé doplňující proměnné [24].

Pro každou třídu c_i máme kladné celé číslo a_i, které představuje maximální počet vyučovacích hodin, kterých se může třída c_i zúčastnit během každého z p dnů. Podobně si zavedeme číslo b_j pro učitele t_j (tedy kolik vyučovacích hodin je schopný odpracovat během každého z p dnů). Význam proměnné x_{ijk} se příliš nemění, v týdenním plánování bude představovat počet vyučovacích hodin zahrnujících třídu c_i a učitele t_j, které jsou přiřazeny ke dni k (nenabývá tedy už jen hodnot 0 a 1, ale maximálně hodnoty a_i, případně b_j). Tyto podmínky si zapíšeme v následujících nerovnicích

$$\sum_{j=1}^{n} x_{ijk} \leq a_i \qquad \forall(i = 1, \ldots, m;\ k = 1, \ldots, p), \tag{5}$$

$$\sum_{i=1}^{m} x_{ijk} \leq b_j \qquad \forall(j = 1, \ldots, n;\ k = 1, \ldots, p), \tag{6}$$

$$x_{ijk} \geq 0 \qquad \forall(i = 1, \ldots, m;\ j = 1, \ldots, n;\ k = 1, \ldots, p). \tag{7}$$

Řešení těchto podmínek 5—7 existuje, pokud jsou splněny následující nerovnice

$$\sum_{i=1}^{m} r_{ij} \leq p \cdot b_j \qquad \forall(j = 1, \ldots, n),$$

$$\sum_{j=1}^{n} r_{ij} \leq p \cdot a_i \qquad \forall(i = 1, \ldots, m),$$

také jinak, pokud se učitel (respektive třída) vyskytuje jen na maximálně tolika hodinách, kolik je mu naplánováno na týden [24].

Týdenní plánování rozvrhů se dá také vyjádřit bipartitním multigrafem. V tomto grafu musíme přiřadit jednu z p barev ke každé hraně takovým způsobem, že maximálně a_i sousedních hran, které vedou z uzlu c_i, mají stejnou barvu (respektive maximálně b_j hran z uzlu t_j). Jelikož tyto hodiny se stejnou barvou můžou probíhat v různých dnech [24].

Pokud máme zadány hodnoty a_i a b_j, jsme schopni určit minimální počet dní p, které potřebujeme k narozvrhování daného počtu vyučovacích hodin. Tuto hodnotu p spočteme podle následujícího vzorce

$$p = max(\max_j \lceil \sum_{i=1}^{m} r_{ij}/b_j \rceil, \max_i \lceil \sum_{j=1}^{n} r_{ij}/a_i \rceil),$$

kde $\lceil t \rceil$ je horní celočíselná část čísla t z kapitoly 3.1 [24].

To však není příliš obvyklé. Zpravidla máme daný počet dní v týdnu, ve kterých probíhá výuka, a počet vyučovacích hodin jednoho dne je rovnoměrně upraven do p dní. Hledáme tedy celé číslo x_{ijk}, které bude splňovat podmínky

$$\lfloor \sum_{j=1}^{n} r_{ij}/p \rfloor \leq \sum_{j=1}^{n} x_{ijk} \leq \lceil \sum_{j=1}^{n} r_{ij}/p \rceil \qquad \forall (i = 1, \ldots, m;\ k = 1, \ldots, p), \tag{8}$$

$$\lfloor \sum_{i=1}^{m} r_{ij}/p \rfloor \leq \sum_{i=1}^{m} x_{ijk} \leq \lceil \sum_{i=1}^{m} r_{ij}/p \rceil \qquad \forall (j = 1, \ldots, n;\ k = 1, \ldots, p), \tag{9}$$

$$\lfloor r_{ij}/p \rfloor \leq \quad x_{ijk} \quad \leq \lceil r_{ij}/p \rceil \qquad \forall (i = 1, \ldots, m;\ j = 1, \ldots, n;\ k = 1, \ldots, p), \tag{10}$$

kde $\lfloor t \rfloor$ je dolní celočíselná část čísla t, kterou jsme definovali v kapitole 3.1. Omezení 8 a 9 udává denní zátěž všech tříd a všech učitelů, která je dokonale vyvážena během p dní. Omezení 10 vyjadřuje skutečnost, že hodiny zahrnující třídu c_i a učitele t_j jsou rozprostřeny do p dní [24].

Pokud si tuto podmínku představíme v multigrafové interpretaci, tak mezi rovnoběžnými hranami, které vedou mezi vrcholy c_i a t_j existuje alespoň $\lfloor r_{ij}/p \rfloor$ a nejvíce $\lceil r_{ij}/p \rceil$ hran každé barvy [24].

4.2.3 Předběžné přiřazení a nedostupnost

Můžeme mít i další omezení či podmínky, které chceme do rozvrhu zanést. Pro jejich nadefinování se na chvíli vrátíme opět k dennímu plánování, kde pro proměnnou x_{ijk} uvažujeme jen hodnoty 0 a 1. V reálném problému máme spoustu požadavků, které musíme vzít v úvahu. Jeden z nich je předběžné přiřazení [24].

Může se stát, že některé dvojice c_i a t_j mají mít hodiny v pevně stanovených časech k, přiřadíme je tedy předběžně, ještě než začneme vytvářet zbytek rozvrhu. Zavedeme si také novou podmínku s novou proměnnou

$$x_{ijk} \geq p_{ijk}, \tag{11}$$

kde p_{ijk} bude 0, pokud není předem určeno, kdy se setkají třídy c_i a učitelé t_j v obdobích k [24].

Další omezení, které může nastat, je nedostupnost učitelů nebo tříd. V některých časech k nemusí být učitel t_j (případně třída c_i) k dipozici a tudíž se nemůžou zúčastnit žádné vyučovací hodiny. To se zpravidla děje spíše u učitelů, můžeme však takovou pauzu uvažovat pro třídu v případě oběda [24].

Nastavit nedostupnost učitele můžeme vícero způsoby. My budeme uvažovat ten, kde použijeme předběžné přiřazení. Pokud není učitel t_j k dispozici v čase k, přiřadíme ho k fiktivní třídě c_i^*. Podobně pokud je třída c_i zaneprázdněna, přiřadíme ji k fiktivnímu učiteli t_j^* [24].

Místo toho, abychom požadovali dodržení předchozí podmínky, zavedeme si jinou formulaci pro každodenní problém s předem přidělenými hodinami a nedostupností. Definujeme proměnnou $\bar{x}_{ijk} = 1$, pokud se třída c_i a učitel t_j setkají v čase k při hodině, která není zadaná předem, jinak je $\bar{x}_{ijk} = 0$. Tyto vlastnosti jsou shrnuty v následujících rovnicích a nerovnicích

$$\sum_{k=1}^{p} \bar{x}_{ijk} = \bar{r}_{ij} \qquad \forall(i = 1, \ldots, m;\ j = 1, \ldots, n), \tag{12}$$

$$\sum_{j=1}^{n} \bar{x}_{ijk} \leq \bar{b}_{ik} \qquad \forall(i = 1, \ldots, m;\ k = 1, \ldots, p), \tag{13}$$

$$\sum_{i=1}^{m} \bar{x}_{ijk} \leq \bar{c}_{jk} \qquad \forall(j = 1, \ldots, n;\ k = 1, \ldots, p), \tag{14}$$

$$\bar{x}_{ijk} = 0 \text{ nebo } \bar{x}_{ijk} = 1 \qquad \forall(i = 1, \ldots, m;\ j = 1, \ldots, n;\ k = 1, \ldots, p), \tag{15}$$

kde $\bar{r}_{ij} = r_{ij} - \sum_{k=1}^{p} p_{ijk}$,

$\bar{b}_{ik} = 1,$ pokud je třída c_i k dispozici a není přiřazena na období $k,$ jinak je rovno 0,

$\bar{c}_{jk} = 1,$ pokud je učitel t_j k dispozici a není předem přiřazen na dobu $k,$ jinak je rovno 0.

Tato omezení nám definují rozvrhovací problém s předřazenými úkoly. Obecně jsou tyto podmínky 12—15 nezbytné pro existenci řešení. To víme na základě Hallových podmínek pro systémy s odlišnými zástupci (Hallovu větu pro bipartitní grafy zmiňujeme v kapitole 2.3.1). Tato věta v řeči rozvrhů říká, že pro každou periodu k musí pro každou podmnožinu q tříd existovat alespoň q učitelů, kde se každý učitel musí setkat minimálně s jednou z q tříd v čase k. Takové podmínky bohužel nestačí k existenci řešení [24].

4.2.4 Soft a hard podmínky

Podmínek a omezení, které se skrývají za úlohou rozvrhování, je mnoho. Řadu z nich jsme si již vyjmenovali, ne všechny však musíme napevno dodržet (ne vždy se to požaduje a ne vždy je to možné). Z tohoto důvodu lze rozvrhovací podmínky rozdělit na ty nutné (hard podmínky) a ty méně potřebné (soft podmínky) [1, 7, 8].

- **Soft podmínky**

  Podmínek, které nemusí být nutně splněny, máme řadu. I v případě jejich nesplnění lze sestavit akceptovatelné řešení rozvrhu, které bude fungovat [1].

  Mezi tyto podmínky patří například požadavek, aby učitel neměl zbytečné pauzy ve výuce (tedy aby probíhala co nejvíce souvisle) a nemusel zbytečně čekat na své hodiny. Toto je asi nejčastěji optimalizovaná podmínka, jelikož její splnění není až tak snadné, ale je významné a zpříjemní učitelům práci [24].

  Další podmínky nejsou tak složité na splnění, ale také jejich význam je mnohem menší. Například požadavek, aby výuka jednotlivých předmětů probíhala v učebnách, které jsou k tomu specializované. To je dobré splnit alespoň jednou týdně. Jak bylo zmíněno, v praxi se často přiřazení učeben řeší tak, že učitelé si do již vytvořeného rozvrhu sami přiřadí učebnu, ve které povedou danou vyučovací hodinu [24].

  Nakonec zmíníme dvě podmínky, které spolu souvisí. Některé předměty bychom preferovali vyučovat v ranních hodinách (jako například matematika, fyzika, atd.), jelikož

jsou tyto předměty pro děti náročnější. Naopak některé předměty nevyžadují tolik pozornosti, a proto preferujeme jejich výuku v pozdějším čase (například výtvarná výchova, hudební výchova, atd.) [24].

Takových podmínek bychom jistě našli více, nám však pro příklad budou v této práci stačit ty výše zmíněné.

- **Hard podmínky**

  Splnitelnost těchto podmínek již nebývá dobrovolná, ba naopak nutná. Pokud bychom je porušili, docházelo by k vytvoření nerealizovatelných nebo případně nevhodných rozvrhů [1].

  Na rozdíl od učitelů, kteří mohou mít ve výuce pauzy (avšak snažíme se tyto pauzy minimalizovat), děti by měly mít výuku v kuse (s výjimkou pauzy na oběd). Tato podmínka je striktně dodržována na základních školách, aby děti neměly zbytečné záminky opouštět budovu školy [24].

  Co se týče samotné výuky, vždy musíme dodržet již dříve zmíněná pravidla. Učitel ani třída nemohou být na dvou vyučovacích hodinách zároveň a v jedné učebně nesmí probíhat dvě hodiny současně. Nedochází tedy k žádné kolizi [1]. Další podmínka, která bývá zpravidla striktně dodržována je, že třídní učitel vyučuje ve své třídě (to však můžeme zařídit přímo před vytvářením rozvrhů při rozdělování výuky). Preferujeme totiž, aby měl třídní učitel kontakt s dětmi, se kterými řeší třídnické záležitosti.

  Nakonec zmíníme ještě jednu podmínku. Na rozdíl od hodin biologie, kdy je vyhovující, aby výuka probíhala ve specializované učebně biologie (není to však nutné), výuka informatiky musí nutně probíhat v učebně s počítači. Stejně tak tělesná výchova musí probíhat v tělocvičnách a na hřištích k tomu určených [1].

  Opět bychom jistě našli jiné podmínky, které by se daly do této kategorie zařadit, my tu však další nebudeme uvádět.

4.2.5 Složitost úlohy

Zmínili jsme řadu podmínek, které při rozvrhovacích úlohách bereme v potaz a rozhodně to nebyly všechny, které se v dané problematice vyskytují. Přidáním podmínek se zvyšuje složitost řešeného problému rozvrhování [24].

Pokud budeme opět uvažovat denní plánování s nedostupností a předem přidělenými hodinami, jsme schopni o něco přesněji určit výpočetní složitost plánování rozvrhu. Problém rozhodnutí, zda vůbec řešení existuje, případně jeho nalezení, je NP–úplný problém. Existuje však zvláštní případ omezení 12—15, u kterého jsme schopni snadno rozhodnout, zda rozvrh existuje, či nikoliv [8, 24].

Věta 4.1. *[24] Pokud je v omezeních 12—15 každý učitel k dispozici během maximálně dvou vyučovacích hodin potom je výpočetní složitost algoritmu pro nalezení řešení (nebo pro ukázání, zda vůbec řešení existuje)* $O(n^2)$.

Máme-li tedy tento požadovaný tvar, výpočetní složitost se značně zlepší [24].

5 Rozvrhovací úlohy — metody řešení

Při sestavování rozvrhovacích úloh existuje několik možností přístupů, jak tento problém řešit. Avšak většina nám dostupných metod je založena na heuristických postupech, kdy se řešení vytváří krok za krokem. Tyto metody hledají proveditelná řešení bez záruky jejich existence, často kombinují kombinatorické modely (které využívá k některým dílčím problémům) s více či méně intuitivními rozhodovacími pravidly. Základní dva možné přístupy, které si tu představíme, jsou přes teorii grafů a celočíselné lineární programování [1, 24].

Mohli bychom říct, že všechny navržené metody spočívají v postupném přiřazování vyučovacích hodin do určitého časového období a zároveň se snaží dodržet všechny stanovené nutné podmínky existence řešení zbývajícího problému. Tento proces pokračuje do té doby, než je přidělena veškerá výuka nebo dokud již nelze vyučovací hodiny přiřazovat. V takovém případě se některé modely zastaví, jiné zruší některé z posledních úkolů, které zadaly, a začnou znovu v této fázi [1, 24].

Většina algoritmů pracuje nejprve s obtížně plánovatelnými akcemi, jako jsou například půlené hodiny, nebo naopak sdílené hodiny mezi více třídami. Lze je přiřadit předem a následně použít heuristický model s již zadanými předúkoly. Jsou-li však tyto případy v rozvrhu častější, je dobré k tomu mít vyvinutý odpovídající model [24].

Významným měřítkem modelu je také míra volnosti. Pod tímto pojmem si představujeme počet zbývajících vyučovacích hodin, kdy může daný učitel pracovat, od něhož odečteme počet vyučovacích hodin, které je pro něj ještě zapotřebí naplánovat [24].

K dalšímu možnému problému často dochází při plánování rozvrhových akcí na vysokých školách. Ta totiž nabízí sbírku kurzů, který se každý skládá z určitého počtu přednášek. Výběr kurzů může být však z určité části libovolný, nemáme tedy vždy pevně daný učební plán. To se však u základních škol, na které se v této práci především zaměřujeme, nestává a nebudeme tedy tento problém do našeho modelu uvažovat [24].

Teď se již zaměříme na samotné modely rozvrhovacích úloh, tedy alespoň na ty dva základní.

5.1 Rozvrh a teorie grafů

V předchozích kapitolách jsme si již představili řadu možných znázornění rozvrhovacích úloh v grafové struktuře. Především jsme k tomu používali bipartitní grafy, případně bipartitní multigrafy, kde třída c_i je spojena s učitelem t_j hranou (v případě multigrafů může být takových hran více), pokud mají společnou výuku. K přiřazení období, kdy tyto vyučovací hodiny probíhají, můžeme využít hranové obarvení tohoto grafu. Také jsme stanovili pomocí Hallovy věty podmínku řešitelnosti rozvrhu třídy c_i (případně učitele t_j) [24].

Nemusíme se však omezit jen na hranové barvení grafů, daná problematika se dá popsat i pomocí toho vrcholového. Metody, které jsou zaměřeny na vrcholové barvení grafů, obecně vyjmenují všechna možná obarvení s p barvami, proto nejsou přizpůsobeny pro práci s rozsáhlými grafy. V současné době jsou algoritmy tohoto typu schopny pracovat maximálně se stovkami vrcholů grafu. To je však silné omezení, většina rozvrhových grafů jich má více. Z tohoto důvodu probíhaly experimenty na náhodně generovaných grafech až s tisíci uzly, pomocí kterých docházelo k odhadům minimálního počtu potřebných barev na obarvení [24].

Kromě heuristických algoritmů je možné využít takzvaných exaktních algoritmů, které

hledají obarvení vrcholů pomocí p barev vždy, pokud nějaké takové řešení existuje. Heuristické metody většinou obarvují uzel po uzlu, například jednu vyučovací hodinu za druhou. To nazýváme přiřazovací strategií, kterou můžeme rozdělit na dvě skupiny. V prvním případě vybíráme časové období nezávisle na vyučovaném předmětu, v druhém případě je oboje vybíráno současně [7, 24].

- **První přístup**

  Jak bylo již zmíněno, při první strategii přiřazujeme vyučovací hodinu k časovému období nehledě na to, o jaký předmět se jedná. Na řešení tohoto problému zpravidla využíváme sekvenční metody barvení [24].

  Nejprve si určíme pořadí vrcholů, ve kterém je následně obarvíme. Snažíme se vrcholu přiřadit co nejmenší barvu. Seřazení vrcholů probíhá podle různých kritérií, zpravidla to je podle počtu sousedních hodin (nejprve vybíráme ty vrcholy, které mají nejvíce sousedů) [24].

  Z heuristických metod se jako nejefektivnější osvědčila metoda *DSATUR*. Při ní se nejprve vybere vrchol s největším počtem sousedů a ten se obarví nejmenší možnou barvou. Pro každý vrchol x si určíme množinu zakázaných barev $F(x)$ (například barvy, které mají již jeho sousedi). V každém kroku pak vybereme vrchol, který má největší počet zakázaných barev, tedy takový vrchol y, kde je největší $|F(y)|$. Tento vrchol obarvíme nejmenší možnou barvou [24].

  Toto je heuristický postup pro obecné grafy, který je odvozen z přesného postupu, jenž je uplatňován pro barvení bipartitních grafů. V oblasti sekvenčních metod došlo k mnohým experimentům, kdy se hledalo optimální barvení grafů se stovkami uzlů. Tento postup byl původně vytvořen pro náhodně generované grafy, které mohou být obarveny přesně p barvami a ne méně než p barvami [24].

- **Druhý přístup**

  Pokud volíme druhou skupinu strategií, kde vybíráme časové období a vyučovanou hodinu současně, máme v postupu jisté odlišnosti od předchozí varianty. Současně vybíráme uzel, který má být obarven, i barvu, kterou mu přiřadíme [24].

Doteď jsme uvažovali jen strukturu bipartitních grafů. Pokud bychom však chtěli do grafu zanést také seznam učeben, nebude nám vyhovovat. Rozvrh učeben jsme sice do této chvíle neuvažovali, můžeme se však setkat i s omezeními danými počtem současně dostupných učeben. Chceme-li plánovat současně přiřazení učitele k třídě a s tím umístění této vyučovací hodiny do učebny, bude se nám hodit struktura hypergrafu [24].

Uvažujme hypergraf $H = (X, E)$, kde X je konečná množina vrcholů a E je konečná množina hran. Zavedeme pro každou hranu E_i přirozené číslo e_i. Obarvení hypergrafu H je vrcholové, pokud v každé hraně E_i nemůže mít stejnou barvu více než e_i vrcholů. Je-li H graf a pro všechny hrany E_i platí $e_i = 1$, jedná se o obvyklé vrcholové barvení. Takto nadefinovaný graf nám může sloužit k propojení učitele a třídy, které se setkají na vyučovací hodině v čase k, i učebny, ve které výuka proběhne [24].

Obarvení hypergrafu je ovšem obtížnější. V praxi se proto nerozvrhuje rovnou s přiřazením učeben, ale učitelé si volí učebny sami až po sestavení rozvrhu [24].

5.2 Přístup matematického programování k rozvrhování

Dalším možným přístupem k řešení rozvrhovacích úloh je matematické programování. Pod tímto pojmem si můžeme představit více metod. Může se jednat o lineární programování, nelineární programování, nebo také kvadratické programování [16]. Pro rozvrhovací úlohy jsou však nejvhodnější metody lineárního programování a to konkrétně celočíselná optimalizace. Metody celočíselného lineárního programování jsem si také již podrobněji představili v předchozích kapitolách [8].

V případě rozvrhování bylo navrženo několik modelů plánování kurzů, které byly založeny na matematickém programování. Zde si představíme jednu z těchto formulací [24].

Budeme uvažovat soubor vyučovaných předmětů $K_1, \ldots, K_q$, přičemž každý předmět K_i se skládá z k_t vyučovacích hodin po jedné periodě (zpravidla jednom týdnu). Na základních školách se tento počet s každým ročníkem u předmětu mění. Celkový počet období budeme opět uvažovat p. Žáky si rozdělíme do r skupin (nebo také tříd) $S_1, \ldots, S_r$ tak, že v každé S_i navštěvují daní žáci stejné předměty ve stejný čas. Zavedeme si také další konstantu l_k, která bude udávat maximální počet vyučovacích hodin, které lze naplánovat na období k, a tomu musí také samozřejmě odpovídat počet dostupných učeben. Dále si nadefinujeme proměnnou y_{ik}, která je rovna 1, pokud je na čas k naplánovaná vyučovací hodina předmětu K_i, jinak je rovna 0. Pro tyto veličiny si zavedeme určitá omezení

$$\max \sum_{i=1}^{q} \sum_{k=1}^{p} C_{ik} y_{ik}, \tag{16}$$

$$\sum_{k=1}^{p} y_{ik} = k_i \qquad (i = 1, \ldots, q), \tag{17}$$

$$\sum_{i=1}^{q} y_{ik} \leq l_k \qquad (k = 1, \ldots, p), \tag{18}$$

$$\sum_{i \in S_l} y_{ik} \leq 1 \qquad (l = 1, \ldots, r;\ k = 1, \ldots, p), \tag{19}$$

$$y_{ik} = 0 \text{ nebo } y_{ik} = 1. \tag{20}$$

Proměnnou C_{ik} jsme si zatím nezavedli. Jedná se o náklady, které se vyskytují v maximalizované funkci 16. Vyjadřují nám, jak moc žádoucí je, aby předmět K_i probíhal v čase k (tj. aby byla veličina y_{ik} rovna jedné). Pokud pro daný předmět K_i preferujeme, aby se vyučoval v čase k, nastavíme hodnotu C_{ik} vysokou. V opačném případě, není-li to vůbec žádoucí, určíme tuto proměnnou jako nulu [8, 24].

Na vyřešení těchto rovnic je vhodné použít Lagrangeovu relaxační techniku. Před samotným řešením úlohy si však upravíme objektivní funkci následujícím způsobem

$$\max \sum_{i=1}^{q} \sum_{k=1}^{p} C_{ik} y_{ik} + \sum_{l} \sum_{k} \lambda_{kl} (1 - \sum_{i \in S_l} y_{ik}). \tag{21}$$

Tento tvar získáme přenásobením podmínky 19 koeficienty $\lambda_{kl} \geq 0$. Tím se nám daná úloha zjednoduší, jelikož můžeme podmínku 19 vynechat a snížit tím počet omezení [24].

Samotný princip Lagrangeovy relaxace zde nebudeme uvádět, kompletní problematika je podrobněji zpracována ve zdrojích [3] a [18].

5.3 Proveditelnost řešení a jeho optimalita

Do této chvíle jsme u modelů uvažovali jen libovolnou realizaci řešení dané rozvrhovací úlohy, ne však její optimalitu. V reálném problému jsou však modely hledající proveditelná řešení a optimální řešení podobná. Jakékoliv řešení minimalizuje určitou vzdálenost k přijatelnému výsledku. Nelze jednoznačně určit jednu objektivní funkci, která by zahrnovala optimalizaci všech proměnných (například ideální rozvrh pro učitele i třídy). Máme příliš požadavků a kritérií na vytváření rozvrhů, z jejichž pohledu lze rozvrh optimalizovat. To nám však nebrání v použití heuristické metody generování rozvrhů. Během tohoto postupu plníme různá omezení vyskytující se v problému a tím můžeme některé omezení prioritizovat [24].

Pokud se rozhodneme rozvrh optimalizovat ve smyslu některého omezení, většinou se jedná o podmínky typu soft z kapitoly 4.2.4. Tento typ podmínek nemusí být vždy kompletně splněn, proto můžeme hledat řešení, která je co možná nejvíce splňují. Nemusí to však být nutné, můžeme se soustředit pouze na řešitelnost a existenci rozvrhu (jakéhokoliv) [24].

5.4 Skutečné případy v praxi

Do této chvíle jsme se zaměřili na teoretický popis modelů. Ty dodržují různé typy omezení s danými proměnnými a tím řeší danou problematiku. Zmínili jsme řadu složek modelů, které ho ovlivňují, ale stále jich mnoho existuje. Příkladem dalšího omezení je co největší kompaktnost rozvrhu (tedy co nejméně přestávek v práci), určitá požadovaná posloupnost předmětů nebo vyučovací hodiny různých délek. Ačkoliv nejsou tyto podmínky zahrnuty v předchozích zmíněných modelech, heuristické metody (které sestavují rozvrh krok za krokem) jsou schopny zvládnout všechny druhy požadavků [24].

V praxi je vyvinuta řada úspěšných aplikací, které jsou využívány pro sestavování rozvrhů. O tomto počítačovém zpracování rozvrhů na školách si teď něco blíže řekneme.

První počítačové kódy vytvářely rozvrhy takzvaně od nuly, tedy od úplného začátku. Omezení, požadavky či přání na rozvrh byly zadávány se souborem priorit jako vstupní data. Tento kód přiřadil vyučovací hodinu k nějakému časovému období tak, aby byl počet konfliktů co nejmenší (ideálně nulový) nebo nechal vyučovací hodinu nepřiřazenou. Takový kód zvládl vytvořit proveditelný rozvrh s relativně malými ručními úpravami, avšak stejně nebyl učiteli příliš přijat. Hodiny jim rozvrhoval stroj, což se nesetkávalo s přílišným přijetím, proto jsou dnešní aplikace vysoce interaktivní a zásadní rozhodnutí může během procesu učinit rozvrhovatel [24].

Počítačové zpracování rozvrhů má časovou úsporu pro rozvrhovatele, ale to není jediná výhoda. Vytvořené rozvrhy zpravidla neporušují žádná zadaná omezení, což se při ručním sestavování často nepodaří dostatečně podchytit. Z toho důvodu jsou tyto aplikace zajímavé pro velké školy. Ty malé, které mají jen pár tříd a učitelů, takové aplikace nepotřebují, protože se rozvrhy dají sestavit snadno i ručně. U velkých škol s řadou tříd je to však nereálné. Počítačové aplikace mohou také často sloužit k tisku rozvrhů a k převodu rozvrhů do školních aplikací pro každodenní používání [24].

Žádné dvě školy nemají na rozvrhy stejné požadavky, všude jsou nějaká specifická kritéria

ria, která školy požadují. Z toho důvodu by nebylo rozumné, aby byly rozvrhové aplikace čistě univerzální programy. Heuristické metody musí být z toho důvodu pro efektivnost přizpůsobeny povaze omezení a požadavků (a jejich hustotě), které daná škola vyžaduje [24].

Posledním kritériem aplikací, které se používají v praxi, a které v této práci zmíníme, je jejich snadná použitelnost. Školy zpravidla nemají IT specialistu, který by se rozvrhovými problémy zabýval. Z toho důvodu je zapotřebí, aby byly aplikace (k tomuto účelu vytvořené) snadno ovladatelné pro každého. Pochopit princip těchto aplikací může zpravidla pomoct rozvrhovateli dosáhnout dobrého řešení v rozumném počtu běhů programu. Pro dlouhodobější přežití aplikace je také důležité, aby měl uživatel přímý přístup k programu. Pokud bude muset dojíždět pro využití dané aplikace, příliš se to nevyplatí, proto se zpravidla umožňuje přístup k aplikaci přímo z počítačů dané školy [24].

6 Rozvrhovací úlohy — implementace

Již jsme se seznámili s veškerými parametry, které musí rozvrhy obsahovat a dodržovat, a také jednotlivými přístupy, které k řešení rozvrhovacích úloh můžeme využít. V této kapitole se tedy podíváme již na přímou implementaci tohoto problému a sestavíme si model pro návrh rozvrhů. Pro vytvoření našeho vlastního rozvrhu využijeme modelovací jazyk pro matematické programování *AMPL*, ve kterém využijeme tří řešičů pro sestavení vyhovujícího rozvrhu. Náš výpočet modelu bude tedy založen na maximalizační úloze lineárního programování, podobně jako v kapitole 5.2. Na závěr zmíníme řešení, které bylo sestaveno za pomoci programu *aSc Rozvrhy*, který je využíván na 31. základní škole v Plzni. Tato škola nám také poskytla reálná data, která byla využita při sestavení našeho modelu, jak bylo již dříve zmíněno.

6.1 Implementace v modelovacím jazyce *AMPL*

Zkratka *AMPL* vychází z anglického "*A Mathematical Programming Language*". Je to tedy programovací jazyk, který se využívá k modelování a řešení různých typů rozsáhlých optimalizačních problémů. V našem případě se jedná o problém celočíselného lineárního programování, na jehož principech je založena rozvrhovací úloha. Matematický model rozvrhovací úlohy, tedy množina proměnných a výčet podmínek, na kterých je rozvrh založen, jsou do prostředí *AMPLu* vkládány pomocí textového souboru. Ten následně využívá vestavěné řešiče k vyčíslený splnitelného rozvrhu. My si vyzkoušíme tvorbu rozvrhů za pomocí tří řešičů — *CPLEX*, *Gurobi* a *CBC*. Nejprve si však představíme obecný model dané úlohy, který je pro všechny řešiče stejný. Model našeho rozvrhu je sepsán v přiloženém souboru *Rozvrh.txt*, který si zde popíšeme a vysvětlíme [10, 12].

6.1.1 Vlastní model v *AMPLu*

V první části modelu jsou nadefinovány množiny proměnných, které budeme potřebovat. Konkrétně se jedná o dvaatřicet tříd, z čehož šestnáct tříd je reálných a šestnáct fiktivních. Mezi reálné třídy patří vždy čtyři třídy z každého ročníku (čtyři šesté třídy, čtyři sedmé, čtyři osmé a čtyři deváté). Fiktivní třídy jsou zavedeny z důvodu půlených hodin, kdy každá polovina třídy má výuku s jiným učitelem. Třídě VI.A (v množině tříd je tato třída označena jako číslo 1) přísluší fiktivní třída číslo 17, třídě VI.B (s označením 2) náleží fiktivní třída číslo 18, atd. Dalším parametrem vstupujícím do modelu jsou učitelé, kterých máme šestapadesát, mezi nimiž je čtyřicet reálných a šestnáct fiktivních učitelů. Zde jsou fiktivní učitelé zavedeni z důvodu pauzy na oběd, kterou v modelu vytváříme mechanicky pro každou třídu jako rozvrhovou akci, ke které tudíž musíme přiřadit učitele. Následně jsme nadefinovali množinu osmnácti vyučovaných předmětů, mezi nimiž je zahrnuta i obědová pauza. V neposlední řadě jsme pak do modelu zavedli počet dní, ve kterých se koná výuka (tedy pondělí až pátek), a počet vyučovacích hodin, které se mohou odehrát v jednom dni, to jest celkem osm vyučovacích hodin.

V druhé části rozvrhovacího modelu již definujeme matici $\boldsymbol{R}$ o velikosti $m \times n$, kde m je počet tříd a n je počet učitelů. Hodnota r_{ij} pak udává počet vyučovacích hodin v týdnu, které má společné třída i a učitel j, jak bylo již dříve zadefinováno. Dále využijeme matici E, kde

koeficient $e_{h,l}$ udává, jak žádoucí je, aby se předmět h konal ve vyučovací hodině l. Například u vyučovacího předmětu 3, tedy matematiky, preferujeme ranní výuku. Hodnota $e_{3,l}$ pro $(l = 1, .., 3)$ bude tedy vysoká. Poslední proměnnou, kterou si pro náš model nadefinujeme, je $x_{h,i,j,k,l}$. Ta nabývá hodnoty 1, pokud se koná vyučovací hodina ve třídě i s učitelem j daného předmětu h v den k a vyučovací hodinu l. V jiném případě se rovná 0. To bude také proměnná, kterou hledáme.

Tímto jsme nadefinovali proměnné a jejich specifika, které do modelu vstupují, také neznámou proměnnou, která bude zahrnuta do objektivní funkce, a zaměříme se na samotnou funkci, kterou budeme optimalizovat. Tvar objektivní funkce je vyjádřen následujícím výrazem

$$\max \sum_{h=1}^{q} \sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{k=1}^{p} \sum_{l=1}^{s} e_{h,l}\, x_{h,i,j,k,l},$$

kde q je počet předmětů, m je počet tříd, n je počet učitelů, p je počet vyučovacích dní a l je počet vyučovacích hodin v jednom dni. Tato objektivní funkce se snaží přiřadit dané rozvrhové akce mezi učitelem j a třídou i ke každému dni k, aby daný předmět h byl v co nejvhodnější dobu l.

Následně jsou již v modelu definovány konkrétní podmínky, které musí rozvrhy splňovat. Tyto implikované podmínky zahrnují 26 globálních podmínek, které musí být splněny pro všechny rozvrhy, a následně kritéria pro konkrétní třídy. Prvně si zde přiblížíme obecné podmínky pro všechny třídy. První podmínka udává, že počet hodin, které mají v jednom týdnu společné třída i a učitel j, se rovná příslušné hodnotě r_{ij} v matici $\boldsymbol{R}$. Toto pravidlo máme znázorněno v následující rovnici

$$\sum_{h=1}^{q} \sum_{k=1}^{p} \sum_{l=1}^{s} x_{h,i,j,k,l} = r_{i,j} \qquad \forall (i = 1, \ldots, m;\ j = 1, \ldots, n).$$

Podmínky 2 a 3 zaručují, že všechny třídy a všichni učitelé můžou být jen v jedné vyučovací hodině v daný čas. Výuka se tedy nesmí krýt. Zde si znázorníme nerovnici, která zakazuje kolizi pro učitele

$$\sum_{h=1}^{q} \sum_{i=1}^{m} x_{h,i,j,k,l} \leq 1 \qquad \forall (j = 1, \ldots, n;\ k = 1, \ldots, p;\ l = 1, \ldots, s).$$

Podmínky 4—7 jen kapacitně udávají, kolik maximálně hodin se smí odehrát v jednom dni a jednom týdnu (tedy pět vyučovacích dní a v každém dni může probíhat až osm vyučovacích hodin).

Podmínky 8—10 a 25—26 omezují počet hodin daného předmětu, které se mohou konat v rámci jednoho dne. Většina je omezena na jednu vyučovací hodinu, aby byl rozvrh pestrý. Pro předměty výtvarná výchova, pracovní činnosti a tělesná výchova povolujeme dvě vyučovací hodiny za den, jelikož tyto předměty nejsou pro děti příliš náročné a dvouhodinové vyučování daného předmětu často i preferujeme. Následující nerovnicí lze například vyjádřit, že se v rámci jednoho dne má konat v dané třídě nejvýše jedna hodina matematiky

$$\sum_{j=1}^{n} \sum_{l=1}^{s} x_{3,i,j,k,l} \leq 1 \qquad \forall (i = 1, \ldots, m;\ k = 1, \ldots, p).$$

Podmínky 11—20 zakazují některé předměty v odpoledním vyučováním a podmínkami 21 a 22 specifikujeme, kdy může probíhat obědová pauza, respektive kdy probíhat nesmí. Opět si

zde uvedeme jeden příklad, kdy nedovolujeme výuku českého jazyka v odpoledním vyučování

$$\sum_{j=1}^{n}\sum_{l=7}^{8} x_{1,i,j,k,l} = 0 \qquad \forall(i = 1,\ldots,m;\ k = 1,\ldots,p).$$

V poslední části modelu jsou již podmínky specifikovány pro jednotlivé třídy. Řada z nich udává počty hodin daného předmětu, které se v jednom týdnu u třídy konají, a který učitel tento předmět v dané třídě vyučuje. Například následující rovnice nám říká, že třída VI.A (tedy třída 1) má s učitelem 22 pět hodin českého jazyka (předmětu číslo 1) v jednom týdnu

$$\sum_{k=1}^{p}\sum_{l=1}^{s} x_{1,1,22,k,l} = 5.$$

Dalším typem podmínky, kterou si pro třídy zavedeme, je přiřazení půlených skupin k jejich vyučujícím. Současně s tím implementujeme, že se půlené hodiny v dané třídě odehrávají v jeden čas. Opět si zde uvedeme příklad, kdy požadujeme, aby výuka anglického jazyka (tedy předmětu 2) probíhala ve třídě 1 a její druhé půlce (fiktivní třídě 17) s učiteli 37 a 38 v ty samé časy. Také jinak příslušné proměnné $x_{h,i,j,k,l}$ jsou si pro danou třídu a dané učitele rovny

$$x_{2,1,38,k,l} = x_{2,17,37,k,l} \qquad \forall(k = 1,\ldots,p;\ l = 1,\ldots,s).$$

V některých případech bylo zapotřebí využít předběžné přiřazení vyučovací hodiny. Těchto hodin není příliš, zpravidla se jedná o hodiny plavání, které jsou omezeny časových harmonogramem bazénu a povinně volitelného předmětu, který si vybírají žáci sedmého ročníku.

Pokud máme sepsané tyto podmínky, lze již daný model vyřešit pomocí různých řešičů. V našem případě jsem využili tři již dříve uvedené, které teď podrobněji popíšeme na třech různých typech rozvrhů — běžné osmé třídy, jednoho učitele s plným úvazkem (tedy 22 hodin výuky týdně) a jednoho učitele s částečným úvazkem (konkrétně 6 hodin týdně). Pomocí našeho modelu a daných řešičů však byly sestaveny rozvrhové akce pro celý druhý stupeň zmíněné základní školy. Všechny výsledky jsou dostupné v přiložených souborech, které budeme konkrétně zmiňovat v následujících kapitolách této práce.

6.1.2 *Gurobi*

Gurobi je jeden z nejmodernějších řešičů matematického programování, který je schopen řešit úlohy lineárního i kvadratického programování. Je vhodný také u problému s celočíselným lineárním programováním, na kterém jsou postaveny modely rozvrhovacích úloh. Tento řešič je navržen pro využití moderních procesorů s nejpokročilejší implementací nejnovějších algoritmů [12].

Výpočet našeho modelu trval v případě tohoto řešiče průměrně 10 sekund a řešení objektivní funkce dosáhlo hodnoty 398. Jelikož náš model je optimalizovaný především z pohledu třídy, tento typ rozvrhu pro ni vychází velmi přijatelný. Zde si ukážeme příklad rozvrhu třídy VIII.B. Obecně námi vytvořený model preferuje výuku bez odpoledního vyučování. Tomu se však nedá vyhnout u osmých a devátých ročníků. U třídy VIII.B máme dvě odpolední vyučování. Náš program však volí pro odpolední výuku vhodné předměty, které nejsou pro děti tak náročné na soustředění. Zde jsou to konkrétně předměty výchova k občanství a tělesná výchova. Na obrázku 1 máme graficky znázorněn kompletní rozvrh pro tuto třídu.

VIII.B	1.	2.	3.	4.	5.	6.	7.	8.
	8:00-8:45	8:55-9:40	10:00-10:45	10:55-11:40	11:50-12:35	12:45-13:30	13:35-14:20	14:25-15:10
Po	Čj učitel 19	M učitel 17	Př učitel 2	Ch učitel 16	D učitel 8	Pč učitel 3 Pč učitel 24	O B Ě D	
Út	Aj učitel 15 Aj učitel 38	Čj učitel 19	M učitel 17	Inf učitel 7 Inf učitel 20	Vv učitel 7	VKZ učitel 17	O B Ě D	VKO učitel 25
St	M učitel 17	F učitel 23	Aj učitel 15 Aj učitel 38	D učitel 8	Nj učitel 18	Tv učitel 1	O B Ě D	Tv učitel 1
Čt	M učitel 17	Aj učitel 15 Aj učitel 38	Čj učitel 19	Nj učitel 18	Hv učitel 6	Ch učitel 16	O B Ě D	
Pá	M učitel 17	Čj učitel 19	F učitel 23	Z učitel 17	Nj učitel 18	Př učitel 2	O B Ě D	

Obrázek 1: *Rozvrh třídy VIII.B sestavený řešičem Gurobi.*

Náš model však již není optimalizovaný z pohledu učitele. Může se tedy stát, že bude mít větší mezery ve výuce. V případě učitele s pořadovým číslem 22 není výsledek tohoto řešiče příliš optimální, dochází zde k četným mezerám v daném dni. Kompletní grafické řešení můžeme opět vidět na obrázku 2.

učitel 22	1.	2.	3.	4.	5.	6.	7.	8.
	8:00-8:45	8:55-9:40	10:00-10:45	10:55-11:40	11:50-12:35	12:45-13:30	13:35-14:20	14:25-15:10
Po	Aj IX.B	Čj IX.C	Aj IX.C	Čj VI.A		VKO VI.A		VKZ IX.C
Út		Čj VI.A		Čj VIII.D		Čj IX.C		
St	Aj IX.B	Čj VIII.D	Aj IX.C	Čj VI.A	Čj IX.C			
Čt		Čj VI.A	Čj VIII.D	Čj IX.C				
Pá	Čj IX.C	Aj IX.B	Aj IX.C	Čj VI.A		Čj VIII.D		

Obrázek 2: *Rozvrh učitele 22 sestavený řešičem Gurobi.*

Poslední rozvrh, který zde analyzujeme je pro učitele 30. Tento učitel vyučuje matematiku a výtvarnou výchovu v jedné třídě VII.D. Jelikož máme v podmínkách zahrnut požadavek, aby v rámci jednoho dne probíhala ve třídě maximálně jedna hodina matematiky, musí tento učitel

docházet do školy minimálně čtyřikrát týdně. Jeho rozvrh je tedy nejméně ideální ze všech doteď zmíněných. Dalším neoptimálním parametrem je velká přestávka, kterou má učitel mezi výukou matematiky a výtvarné výchovy, jelikož matematiku preferujeme v ranních hodinách, kdežto výtvarnou výchovu spíše v těch pozdějších. Opět si můžeme rozvrh prohlédnout na obrázku 3.

učitel 30	1. 8:00-8:45	2. 8:55-9:40	3. 10:00-10:45	4. 10:55-11:40	5. 11:50-12:35	6. 12:45-13:30	7. 13:35-14:20	8. 14:25-15:10
Po								
Út		M VII.D						
St	M VII.D				Vv VII.D			
Čt		M VII.D				Vv VII.D		
Pá		M VII.D						

Obrázek 3: *Rozvrh učitele 30 sestavený řešičem Gurobi.*

To byly jen příklady některých typů rozvrhů. Kompletní řešení, které bylo vypočítáno tímto řešičem si můžeme prohlédnout v přiložených souborech — řešení výčtem prvků v souboru *Vysledek gurobi.txt* a grafické zpracování v souboru *Rozvrh - výsledek gurobi.xlsx*.

6.1.3 *CPLEX*

Řešič *CPLEX* také slouží k optimalizaci problému celočíselného lineárního programování. Využívá k tomu nástrojů primárního a duálního simplexového algoritmu a dalším metod [12].

I v případě tohoto řešiče výpočet našeho modelu trval v průměru 10 sekund a dosahuje stejné hodnoty objektivní funkce, tedy 398. Opět si zde nejprve ukážeme řešení rozvrhu pro třídu VIII.B, které se principiálně od předchozího řešiče příliš neliší. Opět máme dvě hodiny v odpolední výuce, s trochu odlišnou skupinou předmětů — tělesná výchova a výchova ke zdraví. Obecně jsou rozvrhy jednotlivých tříd při řešení daných řešičů velmi podobné, jelikož dodržují stejné podmínky. Pro řadu předmětů máme stanovené stejné váhy v matici E, tedy kdy preferujeme, aby výuka probíhala (například pro biologii, zeměpis a dějepis). Je tedy možné předměty přeskupit odlišným způsobem, aniž by došlo k poklesu objektivní funkce. Z toho důvodu máme danou hodnotu pro oba řešiče stejnou. Grafické znázornění rozvrhu, které jsme získali s pomocí řešiče *CPLEX*, je na obrázku 4.

U rozvrhu učitele 22 můžeme sledovat již patrnější zlepšení. V rozvrhu se nenachází tolik přestávek a nemáme v případě tohoto řešení pro daného učitele žádně odpolední vyučování. To

VIII.B	1. 8:00-8:45	2. 8:55-9:40	3. 10:00-10:45	4. 10:55-11:40	5. 11:50-12:35	6. 12:45-13:30	7. 13:35-14:20	8. 14:25-15:10
Po	Čj učitel 19	M učitel 17	F učitel 23	Aj učitel 15 Aj učitel 38	Inf učitel 7 Inf učitel 20	Ch učitel 16	O B Ě D	Tv učitel 1
Út	M učitel 17	Čj učitel 19	F učitel 23	D učitel 8	Nj učitel 18	Pč učitel 3 Pč učitel 24	O B Ě D	
St	M učitel 17	Nj učitel 18	Čj učitel 19	Př učitel 2	Hv učitel 6	Ch učitel 16	O B Ě D	VKZ učitel 17
Čt	Př učitel 2	Aj učitel 15 Aj učitel 38	M učitel 17	VKO učitel 25	Vv učitel 7	Tv učitel 1	O B Ě D	
Pá	Aj učitel 15 Aj učitel 38	M učitel 17	D učitel 8	Čj učitel 19	Nj učitel 18	Z učitel 17	O B Ě D	

Obrázek 4: *Rozvrh třídy VIII.B sestavený řešičem CPLEX.*

však nemusí implicitně poukazovat na zlepšení kompletního modelu. Může to znamenat nižší uživatelský komfort rozvrhu jiného učitele. Opět si můžeme řešení prohlédnout na obrázku 5.

učitel 22	1. 8:00-8:45	2. 8:55-9:40	3. 10:00-10:45	4. 10:55-11:40	5. 11:50-12:35	6. 12:45-13:30	7. 13:35-14:20	8. 14:25-15:10
Po	Čj VIII.D		Aj IX.C	Čj VI.A	VKO VI.A	Čj IX.C		
Út	Čj VI.A	Čj VIII.D	Aj IX.B	Aj IX.C		Čj IX.C		
St	Čj VIII.D	Čj IX.C	Aj IX.B	Čj VI.A		VKZ IX.C		
Čt	Čj VI.A	Čj VIII.D	Aj IX.B		Čj IX.C			
Pá	Aj IX.C	Čj IX.C	Čj VI.A					

Obrázek 5: *Rozvrh učitele 22 sestavený řešičem CPLEX.*

Lepší výsledek jsme dostali také v případě rozvrhu učitele 30. Nečeká již na vyučování výtvarné výchovy tři hodiny dva dny v týdnu. Výuka výtvarné výchovy probíhá v jeden den jako dvouhodinové vyučování a tím se kvalita tohoto rozvrhu značně zvyšuje. Zlepšení lze však vidět jen u výuky výtvarné výuky. V případě matematiky musí opět učitel dojíždět do školy na jednu

vyučovací hodiny čtyři dny v týdnu, což je značně neoptimální. Daný rozvrh můžeme vidět na obrázku 6.

učitel 30	1.	2.	3.	4.	5.	6.	7.	8.
	8:00-8:45	8:55-9:40	10:00-10:45	10:55-11:40	11:50-12:35	12:45-13:30	13:35-14:20	14:25-15:10
Po	M VII.D							
Út			M VII.D		Vv VII.D	Vv VII.D		
St	M VII.D							
Čt		M VII.D						
Pá								

Obrázek 6: *Rozvrh učitele 30 sestavený řešičem CPLEX.*

Jedná se opět jen o částečné prezentování výsledků, které jsme získali za pomoci tohoto řešiče. Kompletní řešení pro všechny třídy a několik vybraných učitelů si můžeme prohlédnout v přiložených souborech *Vysledek cplex.txt* a *Rozvrh - výsledek cplex.xlsx*.

6.1.4 *CBC*

Řešič *CBC* je napsán v programovacích jazyce *C++* a lze jeho pomocí opět řešit úlohy lineárního programování. Je velmi přístupný, jelikož jeho licence je veřejná. Využívá se tedy pro komerční softwary bez jakýchkoliv větších problémů. Slouží primárně jako knihovna, kterou lze pro vyřešení zavolat [12].

V případě tohoto řešiče nedosahujeme již tak dobrých časových výsledků. Výpočet našeho modelu v případě *CBC* trvá v průměru 7 minut a 30 sekund, tady 45× déle než v případě zbylých využitých řešičů. Dosahujeme však opět stejné hodnoty objektivní funkce, tedy 398. To se děje ze stejného důvodu, jako bylo popsáno již dříve. Pomocí různých permutací daného rozvrhu lze dosáhnout stejného výsledku objektivní funkce i přesto, že rozvrh není úplně stejný.

V případě prvního rozvrhu, tedy třídy VIII.B, máme opět velice podobný výsledek. Tentokrát se v odpoledním vyučování nachází předměty výchova ke zdraví a výchova k občanství. Můžeme si všimnout, že u žádného řešiče jsme nedostali výsledek, ve kterém by se předmět vyučoval ve dvou po sobě jdoucích hodinách. V případě řady předmětů jsme tento přístup zakázali, avšak jsou výjimky, u kterých nám to nevadí. Například výuka tělesné výchovy by mohla probíhat v kuse v rámci jednoho dne. Tento požadavek by se mohl uplatnit především u výuky výtvarné výchovy a pracovních činností. V případě osmých ročníků se tyto předměty sice vyučují jen jednou týdně, avšak v sedmých ročnících jsou to již dvě hodiny v týdnu. Pokud jsou pracovní

činnosti brány jako hodiny vaření, dvouhodinové vyučování bychom mohli dokonce požadovat. Toto kritérium však zatím není v našem modelu zavedeno a tudíž není náš model plně optimální. Rozvrh třídy VIII.B, který jsme dostali pro řešič *CBC* je znázorněn na obrázku 7.

VIII.B	1. 8:00-8:45	2. 8:55-9:40	3. 10:00-10:45	4. 10:55-11:40	5. 11:50-12:35	6. 12:45-13:30	7. 13:35-14:20	8. 14:25-15:10
Po	Čj učitel 19	Nj učitel 18	M učitel 17	Aj učitel 15 Aj učitel 38	Ch učitel 16	Př učitel 2	O B Ě D	VKO učitel 25
Út	Př učitel 2	M učitel 17	F učitel 23	Čj učitel 19	Pč učitel 3 Pč učitel 24	Tv učitel 1	O B Ě D	
St	D učitel 8	Čj učitel 19	M učitel 17	Aj učitel 15 Aj učitel 38	Tv učitel 1	Inf učitel 7 Inf učitel 20	O B Ě D	VKZ učitel 17
Čt	Nj učitel 18	Ch učitel 16	M učitel 17	Aj učitel 15 Aj učitel 38	Vv učitel 7	D učitel 8	O B Ě D	
Pá	M učitel 17	Čj učitel 19	F učitel 23	Nj učitel 18	Hv učitel 6	Z učitel 17	O B Ě D	

Obrázek 7: *Rozvrh třídy VIII.B sestavený řešičem CBC.*

Rozvrh učitele 22 se spíše podobá řešení pomocí *CPLEXu*, avšak přestávek je zde již více. Nenachází se v něm však žádné odpolední vyučování, a proto se jeví o něco lépe než za pomoci řešiče *Gurobi*. Grafické řešení je znázorněno na obrázku 8.

učitel 22	1. 8:00-8:45	2. 8:55-9:40	3. 10:00-10:45	4. 10:55-11:40	5. 11:50-12:35	6. 12:45-13:30	7. 13:35-14:20	8. 14:25-15:10
Po		Aj IX.B	Čj IX.C	Čj VI.A		Čj VIII.D		
Út	Aj IX.C		Čj VI.A	Aj IX.B	Čj IX.C	VKZ IX.C		
St	Čj VIII.D	Čj VI.A	Aj IX.B	Aj IX.C		Čj IX.C		
Čt	Aj IX.C		Čj VIII.D	Čj VI.A		Čj IX.C		
Pá			Čj VIII.D	Čj VI.A	VKO VI.A	Čj IX.C		

Obrázek 8: *Rozvrh učitele 22 sestavený řešičem CBC.*

Pro učitele 30 dosahujeme velice podobných výsledků jako v případě *CPLEXu*. Rozvrh je značně zlepšen dvouhodinovou výukou výtvarné výchovy. To však není v modelu přímo implementováno a jedná se spíše o náhodný jev. Daný rozvrh si můžeme opět prohlédnout na obrázku 9.

učitel 30	1. 8:00-8:45	2. 8:55-9:40	3. 10:00-10:45	4. 10:55-11:40	5. 11:50-12:35	6. 12:45-13:30	7. 13:35-14:20	8. 14:25-15:10
Po	M VII.D							
Út	M VII.D							
St			M VII.D		Vv VII.D	Vv VII.D		
Čt								
Pá			M VII.D					

Obrázek 9: *Rozvrh učitele 30 sestavený řešičem CBC.*

Za pomoci tohoto řešiče byly sestaveny rozvrhy i pro zbylých patnáct tříd. Toto řešení si můžeme podrobně prohlédnout v přiložených souborech — výčet výsledných proměnných v souboru *Vysledek cbc.txt* a tabulkové zpracování v *Rozvrh - výsledek cbc.xlsx*.

6.1.5 Prostor pro zlepšení modelu

V předchozích kapitolách jsme uvedli příklady některých rozvrhů, které jsme vypracovali za pomoci řešičů *Gurobi*, *CPLEX* a *CBC*. Tato řešení podchycují řadu podmínek, které musí daný rozvrh splnit, a také těch, které slouží k většímu komfortu při výuce. Dané rozvrhy, které jsou výsledkem tohoto modelu, jsou splnitelné a v praxi by bylo možné je využít při běžném chodu základní školy. Je zde však stále prostor pro jistá zlepšení.

Daný rozvrh je optimalizovaný především z pohledu tříd, bylo by však možné přidat i podmínky, které by více zajišťovaly optimalitu z pohledu učitele. V aktuálním modelu se může stát, že má učitel více přestávek během výuky a tyto pauzy můžou být různě dlouhé.

Mezi další možná zlepšení lze zahrnout dvouhodinové vyučování vybraných předmětů. Toto kritérium bylo již dříve zmíněno. Je žádoucí, například při výuce výtvarné výchovy, kde žákům trvá delší dobu příprava a sklízení pomůcek, aby výuka probíhala ve dvou po sobě jdoucích hodinách.

Poslední možné zlepšení, které zde zmíníme, je zkrácená výuka v pátek. Tento požadavek není k funkčnosti rozvrhu nijak zásadní, je však příjemným zlepšením vyučování jak pro třídy, tak pro učitele. Komfort rozvrhu by se tedy s přidáním tohoto kritéria zvýšil.

Zmínili jsme některá kritéria, která by do budoucna mohla být přidána do našeho již vytvořeného modelu. Tento model je založen na úloze matematického programování. V předchozích kapitolách jsme si zmiňovali druhý možný přístup a to za pomoci přiřazovacího algoritmu. Tento postup by mohl být zpracován například prostřednictvím programovacího prostředí *MATLAB*. Veškerá tato zlepšení a návrh dalšího algoritmu by mohly být předmětem další práce.

6.2 Program *aSc Rozvrhy*

V této části práce si představíme program *aSc Rozvrhy*, který je využíván na již zmíněné základní škole. Tato aplikace je velice interaktivní a rozvrhovatel může sám činit rozhodnutí, která značně ovlivňují budoucí vytváření rozvrhu. Také z velké míry spolupracuje se školní internetovou stránkou *Škola Online*, ve které se zobrazují rozvrhy pro učitele i žáky. A především práce s touto aplikací je snadná a intuitivní.

Při sestavování rozvrhu se první kroky provádí přímo na stránce *Škola Online*, na které se aktualizují informace pro následující školní rok. Mezi ně patří seznam učeben, rozdělení tříd do skupin a úvazky učitelů. Tyto informace jsou následně importovány do samotné aplikace programu *aSc Rozvrhy*. V aplikaci již nastavujeme podrobnější podmínky pro učitele, třídy i dané předměty. Podobně jako v našem modelu si nastavíme, kdy preferujeme výuku daného předmětu. Takto můžeme nastavit i nedostupnost učitele či třídy v případě, že v některou vyučovací hodinu nejsou k dispozici. S těmito parametry by bylo možné daný rozvrh spustit, předchází tomu však ještě jeden krok.

Na této základní škole je vždy jedna třída v ročníku sportovní. To komplikuje daný rozvrh. Sportovci dojíždí na tréninky i mimo školu a podřizuje se tomu tedy částečně rozvrh zbylých hodin. V první řadě, než se spustí vytváření samotného rozvrhu, se ručně vytváří rozvrh tělesné výchovy, který se vloží do programu jako předřazené vyučovací hodiny. S těmito zadanými hodinami následně řešíme zbytek rozvrhu. Tento parametr modelu jsme dříve neuvažovali, jelikož k mimoškolnímu rozvrhu tělesné výchovy nemáme přístup, proto jsme u tělesné výchovy brali v potaz jen kapacitu tělocvičen.

Pokud máme připravená data s danými podmínkami, můžeme se pustit do automatického výpočtu rozvrhu. Ten probíhá v řádu několika minut, výsledek však nebývá kompletní. Daná základní škola má na rozvrhy řadu požadavků, které je složité splnit všechny najednou. Zpravidla se tedy stane, že program nahlásí chybu a vypíše seznam předmětů, které nebyl schopen přiřadit. Tyto rozvrhové akce musí být následně přiřazeny ručně rozvrhovatelem. To je značně neoptimální řešení. To také může být způsobeno tím, jak daný program rozvrhy sestavuje. Ten vyhodnotí řadu možností rozvrhů, které by byly vyhovující vzhledem k požadavkům.

S manuálním upravováním rozvrhů souvisí také následující problém. Pokud je rozvrh doplňován ručně, může docházet k zanášení chyb. Rozvrh pro větší školy je těžké sestavovat bez použití programů, to stejné platí pro částečnou komplementaci. Takto zpravidla dochází k nechtěným chybám, které by znemožnily běžný chod výuky. Některé podmínky (které nejsou pro proveditelné řešení nezbytné) jsou však při ruční úpravě porušeny záměrně, jelikož by jinak rozvrh nešel sestavit (z důvodu vysokého počtu podmínek). Například jsou do odpoledního vyučování přiřazeny některé předměty, které byly v původních omezeních zakázány.

Velkým problémem v praxi bývá přiřazení půlených hodin. Na tento úkol je program *aSc Rozvrhy* přizpůsoben a měla by tato přiřazení fungovat automaticky. Není to však příliš pravidlem. Nastávají situace, ve kterých se půlené hodiny přiřazují v jiných časových období.

Následně musí být tyto chyby opět upraveny ručně.

V tomto programu však lépe funguje dvouhodinová výuka. Pokud má třída dvě hodiny výtvarné výchovy nebo pracovních činností týdně, automaticky jsou přiřazeny do jednoho dne jako dvouhodinová výuka.

Uvedeme si zde poslední kritérium, které je zaneseno do rozvrhů vytvořených na dané základní škole. Tyto rozvrhy zpravidla sestavují přijatelnější řešení pro učitele s částečnými úvazky, jejichž počet ve sledovaném období značně narostl (v této práci jsme pracovali s daty a rozvrhy, které byly aktuální pro první pololetí školního roku 2023/2024 a všechny výstupy, které zde zmíníme, byly platné v tomto období). Řada těchto učitelů jsou studenti vysokých škol, kteří při studiu pracují na pár vyučovacích hodin týdně. Tento případ je i učitel 30, jehož rozvrh jsme již dříve analyzovali. V případě těchto zaměstnanců jsou rozvrhové možnosti omezeny i jinými závazky, proto se rozvrh těmto učitelům uzpůsobuje primárně (dané hodiny jsou předběžně přiřazeny podle dostupnosti daného učitele). I z toho důvodu jsou rozvrhy lépe vyhovující než v případě našeho vytvořeného modelu, u něhož jsme nechali dané rozvrhové akce přiřadit bez jakýchkoliv omezení, jelikož jsme k daným údajům nedostupnosti neměli přístup.

Stejně jako v předchozích případech si zde představíme možné řešení daného programu na příkladu rozvrhů jedné třídy a dvou učitelů. V případě třídy VIII.B máme podobný rozvrh, který jsme vytvořili pomocí našeho modelu. Máme zde však dovolenou dvouhodinovou výuku matematiky. Rozdíl je v předmětech, které jsou v odpolední výuce — hudební výchova a německý jazyk. Výuka cizího jazyku však není pro odpolední vyučování příliš vhodná. Jinak je daný rozvrh stejně přijatelný jako námi vytvořený. Můžeme jej vidět na obrázku 10.

VIII.B	1. 8:00-8:45	2. 8:55-9:40	3. 10:00-10:45	4. 10:55-11:40	5. 11:50-12:35	6. 12:45-13:30	7. 13:35-14:20	8. 14:25-15:10
Po	Nj učitel 18	M učitel 17	D učitel 8	Čj učitel 19	Z učitel 17	Inf učitel 7 Inf učitel 20	OBĚD	
Út	M učitel 17	Nj učitel 18	Tv učitel 1	Aj učitel 15 Aj učitel 38	F učitel 23	VKZ učitel 17	OBĚD	
St	M učitel 17	M učitel 17	F učitel 23	Čj učitel 19	D učitel 8	Pč učitel 3 Pč učitel 24	OBĚD	Nj učitel 18
Čt	Tv učitel 1	Ch učitel 16	M učitel 17	Čj učitel 19	Aj učitel 15 Aj učitel 38	Př učitel 2	OBĚD	Hv učitel 6
Pá	Čj učitel 19	Př učitel 2	VKO učitel 25	Ch učitel 16	Aj učitel 15 Aj učitel 38	Vv učitel 7	OBĚD	

Obrázek 10: *Rozvrh třídy VIII.B sestavený programem aSc Rozvrhy.*

Rozvrh pro učitele 22 se naopak zdá nejlepší ze všech zmíněných. Přestávky se v něm objevují nejméně a výuka končí obecně dříve než v předchozích případech. Tento rozvrh je znázorněn na obrázku 11.

Posledním rozvrhem, který v této práci zmíníme, je opět pro učitele 30. I přesto, že je učitelům s částečnými úvazky rozvrh často přizpůsobován, zde to nelze pozorovat příliš patrně.

učitel 22	1. 8:00-8:45	2. 8:55-9:40	3. 10:00-10:45	4. 10:55-11:40	5. 11:50-12:35	6. 12:45-13:30	7. 13:35-14:20	8. 14:25-15:10
Po	Čj IX.C	Aj IX.C		Čj VI.A	Čj VI.A			
Út	Čj IX.C	Čj VIII.D		Aj IX.B	VKO VI.A			
St	Čj VI.A	Čj VIII.D	Aj IX.B	Čj IX.C				
Čt	Aj IX.B	Čj VIII.D	Aj IX.C	Čj VI.A	Čj IX.C			
Pá	Čj IX.C	Aj IX.C	Čj VI.A	Čj VIII.D		VKZ IX.C		

Obrázek 11: *Rozvrh učitele 22 sestavený programem aSc Rozvrhy.*

Výuka matematiky v sedmé třídě činí 4 vyučovací hodiny v týdnu. Počet pracovních dní je tedy stále vysoký, ale v tomto případě je zkrácen prostřednictvím dvouhodinové výuky na 3 pracovní dny. V předchozích modelech to byly vždy 4 dny výuky. I tento rozvrh je zobrazen v tabulkové podobě na obrázku 12.

učitel 30	1. 8:00-8:45	2. 8:55-9:40	3. 10:00-10:45	4. 10:55-11:40	5. 11:50-12:35	6. 12:45-13:30	7. 13:35-14:20	8. 14:25-15:10
Po	M VII.D							
Út								
St								
Čt			M VII.D	M VII.D	Vv VII.D	Vv VII.D		
Pá	M VII.D							

Obrázek 12: *Rozvrh učitele 30 sestavený programem aSc Rozvrhy.*

Uvedli jsme si zde příklady tří rozvrhů sestavených tímto programem. Kompletní výsledky jsou v tabulkové podobě uvedeny v přiloženém souboru *Rozvrh - výsledek aSc rozvrhy.xlsx*.

7 Závěr

Cílem této práce bylo si přiblížit problematiku vytváření rozvrhů. Tato úloha je obecně známá, její řešení už je náročnější. Je více způsobů, kterými lze výsledného vyhovujícího rozvrhu dosáhnout. V této práce byly představeny dva základní přístupy.

Jeden matematický přístup, na kterém se dá založit rozvrhovací model, je matematické programování. Pod tímto pojmem si můžeme vybavit více pojmů — lineární programování, kvadratické programování, atd. V modelech, které optimalizují rozvrhovací úlohy, se zpravidla využívá celočíselného lineárního programování, konkrétně jeho binární verze. Hledaná veličina $x_{h,i,j,k,l}$ nabývá pouze hodnot 0 a 1. Hodnoty 1 nabývá v případě, že třída i má v den k a vyučovací hodinu l předmět h s učitelem j. Tato proměnná vstupuje do objektivní funkce, která rozvrh optimalizuje primárně pro třídy a vytváří pro ně rozvrh uživatelsky co nejpříjemnější. Pro učitele není náš model příliš optimální. Pro ty s plným úvazkem, který činí 22 vyučovacích hodin týdně, jsou rozvrhy sestavované přijatelně, jen s občasnými přestávkami. Pro učitele s menšími úvazky se však v rozvrhu vyskytují četné pauzy, během kterých učitelem nemá výuku. Přístup matematického programování jsme otestovali v programovacím prostředí *AMPL*, který je řešičem těchto matematických úloh. Model byl řešen prostřednictvím tří řešičů — *Gurobi*, *CPLEX* a *CBC*. Všechny sestavené rozvrhy splňovaly stejné podmínky, byly si tedy relativně podobné, zejména v případě žákovských tříd. V případě učitelů byly již vidět četnější rozdíly, zejména pro učitele s malými úvazky, kde se dá rozvrh značně zlepšit. Z pohledu třídy jsou tedy optimalizovány lépe, což je implikováno v objektivní funkci. I přesto jsou zde některé podmínky, kterými by se daly tyto rozvrhy ještě zlepšit.

Druhým možným přístupem rozvrhovacích úloh je hranové barvení grafů. V tomto grafu máme dvě skupiny vrcholů — vrcholy znázorňující učitele a druhé znázorňující třídy. Tyto vrcholy bipartitního grafu jsou spojeny hranou v případě, že mají v některý čas společnou vyučovací hodinu. Každá vyučovací hodina je následovně reprezentována jednou barvou. Tyto barvy pak vhodně přiřazujeme hranám. V případě korektního rozvrhu nevedou k žádnému vrcholu učitele ani třídy dvě hrany se stejným obarvením. Tento přístup je často implementován pomocí přiřazovacího algoritmu, který je založen na heuristickém přístupu, který přiřazuje jednu vyučovací hodina za druhou. Tento model by mohl být implementovám v programovacím prostředí *MATLAB*, což nebylo předmětem této práce, avšak mohlo by to být tématem následujících prací spolu s lepší aproximací rozvrhu založeném na matematickém programování.

Na základních školách jsou rozvrhovací úlohy problémem, který se řeší pravidelně každý rok. K tomuto účelu slouží již sestavené programy a aplikace. Během výzkumu k naší práci jsme spolupracovali s 31. základní školou v Plzni, která nám poskytla data k praktické části. Současně s tím jsme se seznámili s aplikací, kterou daná škola využívá při sestavování rozvrhů — *aSc Rozvrhy*. Tento program je vysoce interaktivní a práce s ním je poměrně snadná. Plně spolupracuje s dalšími aplikacemi, které slouží k zobrazování rozvrhů pro žáky nebo učitele, jako je například *Škola Online*. Aplikace *aSc Rozvrhy* je schopna sestavit rozvrh s řadou podmínek, které jsou pro něj zadány. Daná základní škola má však na rozvrh spoustu požadavků, které není daný program často schopen splnit, je tedy potřeba rozvrh doupravit ručně. To je velmi problematické, jelikož při manuální komplementaci mohou být do rozvrhu zanášeny chyby. Zkoumali jsme i rozvrhy, které byly vytvořeny tímto programem. Z pohledu některých aspektů, jako je například dvouhodinová výuka, sestavoval dané rozvrhy lépe. Je však značně neoptimální, že se do vytváření rozvrhu musí zasahovat ručně s ohledem na nesplnitelnost podmínek. To může být způsobeno tím, že program nefunguje na principu maximalizace objektivní funkce, ale prohledává možná

řešení a z nich se snaží vybrat to nejpřijatelnější.

Pokud bychom chtěli jednotlivá řešení rozvrhovací úlohy porovnat, lze říct, že z jistého pohledu je náš model lépe optimalizován. Řešení je nalezeno právě tehdy, když existuje (jsou-li zavedeny splnitelné podmínky). To o programu *aSc Rozvrhy* nelze obecně říct, jelikož často není výstupem proveditelné řešení a rozvrh musí být dotvořen manuálně, k čemuž v případě popsaného modelu v *AMPLu* nedochází.

Přílohy

Rozvrh - výsledek aSc rozvrhy.xlsx
Rozvrh - výsledek cbc.xlsx
Rozvrh - výsledek cplex.xlsx
Rozvrh - výsledek gurobi.xlsx
Rozvrh.txt
Vysledek cbc.txt
Vysledek cplex.txt
Vysledek gurobi.txt

Reference

[1] Awad, F. H., Al-kubaisi, A., Mahmood, M. (2022). Large-scale timetabling problems with adaptive tabu search. De Gruyter [online]. [cit. 3.4.2024]. Dostupné z: https://www.degruyter.com/document/doi/10.1515/jisys-2022-0003/html

[2] Bočkayová, T. (2019). Optimalizace v přepravních úlohách. FAV ZČU [online]. [cit. 2.3.2024]. Dostupné z: https://otik.uk.zcu.cz/bitstream/11025/46891/1/DP_Bockayova_Tina_Optimalizace_v_prepravnich_ulohach.pdf

[3] Boyd, S., Vandenberghe, L. (2004). Convex Optimization. Cambridge University Press. [cit. 13.5.2024]. Dostupné z: https://web.stanford.edu/~boyd/cvxbook/bv_cvxbook.pdf

[4] Branda, M. (2021). Úvod do celočíselné lineární optimalitace. Univerzita Karlova [online]. [cit. 4.3.2024]. Dostupné z: https://www.karlin.mff.cuni.cz/~branda/download/UdO_celocislelnost.pdf

[5] Conforti, M., Cornuéjols, G., Zambelli, G. (2014). Integer Programming. Springer. [cit. 10.1.2024].

[6] Dvořák, Z. (2018). Rovinné grafy. Univerzita Karlova [online]. [cit. 25.3.2024]. Dostupné z: https://iuuk.mff.cuni.cz/~rakdver/dm/lesson10.pdf

[7] Ganguli, R., Roy, S. (2017). A Study on Course Timetable Scheduling using Graph Coloring Approach. International Journal of Computational and Applied Mathematics [online]. [cit. 1.4.2024]. Dostupné z: https://www.ripublication.com/ijcam17/ijcamv12n2_26.pdf

[8] Gecmen, D. (2016). Solving the open day timetabling problem using integer linear programming. Technische Universiteit Delft [online]. [cit. 4.4.2024].

[9] Gross, J. L., Yellen, J., Anderson, M. (2019). Graph Theory And Its Applications (third edition). CRC Press. [cit. 15.3.2024].

[10] Havås, J., Olsson, A., Persson, J., Schierscher, M. S. (2013). Modeling and optimization of university timetabling. Chalmers, Göteborgs Universitet. [cit. 4.3.2024].

[11] Hladík, M. (2023). Celočíselné programování. KAM MFF UK [online]. [cit. 10.3.2024]. Dostupné z: https://kam.mff.cuni.cz/~hladik/CP/text_cp.pdf

[12] Jin, H. (2014). A Tutorial of AMPL for Linear Programming. University of Illinois Chicago [online]. [cit. 29.4.2024]. Dostupné z: https://www.cs.uic.edu/~hjin/files/ampl_tutorial.pdf #page3

[13] Kalvoda, T. (2021). Dolní a horní celá část. ČVUT [online]. [cit. 3.4.2024]. Dostupné z: ht tps://kam.fit.cvut.cz/deploy/bi-pkm/mirror/textbook/sec-dolni-a-horni-cela-cast.html

[14] Kovář, P. (2022). Teorie grafů. VŠB-TU Ostrava, ZČU Plzeň. [cit. 17.3.2024].

[15] Kovář, P. (2021). Úvod do Teorie grafů. VŠB–TU Ostrava, ZČU Plzeň. [cit. 18.3.2024].

[16] Lachout, P. (2011). Matematické programování. Univerzita Karlova [online]. [cit. 30.3.2024]. Dostupné z: https://www.karlin.mff.cuni.cz/~lachout/Vyuka/TEX TY/111016-MP_skripta.pdf

[17] Matoušek, J. (2006). Lineární programování. KAM MFF UK [online]. [cit. 4.3.2024]. Dostupné z: https://iti.mff.cuni.cz/series/2006/311.pdf

[18] Nemhauser, G., Wolsey, L. (1988). Integer and Combinatorial Optimization. Wiley. [cit. 14.2.2024].

[19] Ryjáček, Z. (2016). Teorie grafů a diskrétní optimalizace 2. KMA ZČU. [cit. 10.3.2024].

[20] Scheinerman, E. R., Ullman, D. H. (2008). Fractional Graph Theory. John Wiley & Sons. [cit. 18.3.2024].

[21] Turzík, D. (1999). Dualita v lineárním programování. VŠCHT Praha [online]. [cit. 21.3.2024]. Dostupné z: http://147.33.74.135/knihy/uid_isbn-80-7080-363-0/pdf/0 53.pdf

[22] Vanderbei, R. J. (2014). Linear Programming. Springer. [cit. 19.1.2024].

[23] Včelař, F. (2018). Celočíselné lineární programování. Univerzita Tomáše Bati ve Zlíně [online]. [cit. 14.3.2024]. Dostupné z: https://digilib.k.utb.cz/bitstream/handle/10563/4313 8/v%C4%8Dela%C5%99_2018_dp.pdf?sequence=1&isAllowed=y

[24] de Werra, D. (1985). An Introduction to Timetabling. European Journal of Operational Research : Volume 19, Pages 151-162. [cit. 14.1.2024].

[25] Zuzaňáková, J. (2020). Lineární programování v praxi. Masarykova univerzita [online]. [cit. 15.3.2024]. Dostupné z: https://is.muni.cz/th/i23nz/DP_Zuzanakova.pdf