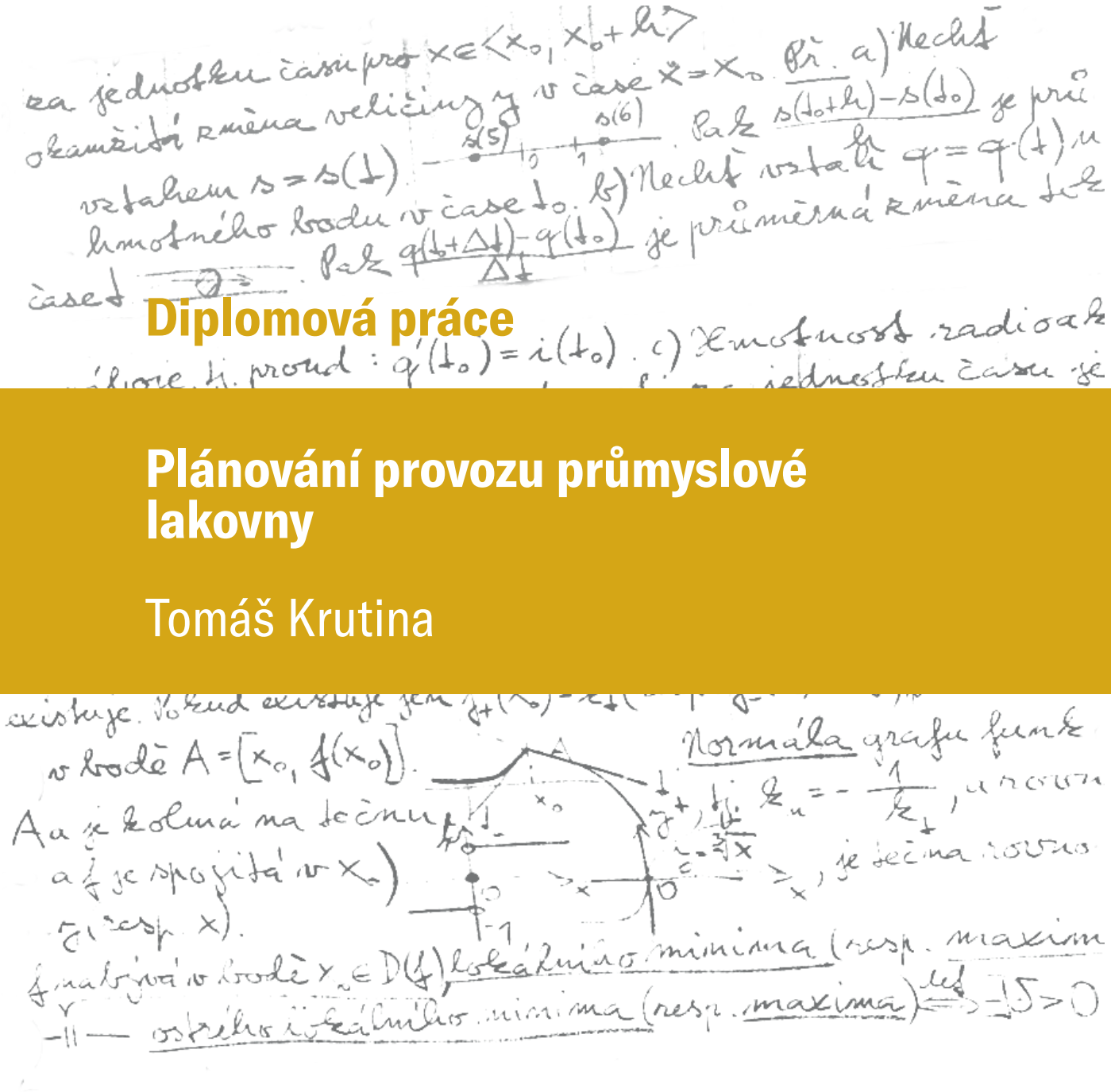
FAKULTA APLIKOVANÝCH VĚD
ZÁPADOČESKÉ UNIVERZITY
V PLZNI

KATEDRA
MATEMATIKY



Diplomová práce

Plánování provozu průmyslové lakovny

Tomáš Krutina

PLZEŇ

2023

FAKULTA APLIKOVANÝCH VĚD
ZÁPADOČESKÉ UNIVERZITY
V PLZNI

KATEDRA
MATEMATIKY

Diplomová práce

Plánování provozu průmyslové lakovny

Bc. Tomáš Krutina

Vedoucí práce
Ing. Patrice Marek, Ph.D.

PLZEŇ **2023**

Citace v seznamu literatury:
KRUTINA, Tomáš. *Plánování provozu průmyslové lakovny*. Plzeň, 2023. Diplomová práce. Západočeská univerzita v Plzni, Fakulta aplikovaných věd, Katedra matematiky. Vedoucí práce Ing. Patrice Marek, Ph.D.

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta aplikovaných věd
Akademický rok: 2022/2023

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Tomáš KRUTINA**
Osobní číslo:	**A21N0005P**
Studijní program:	**N0541A170005 Matematika a finanční studia**
Téma práce:	**Plánování provozu průmyslové lakovny**
Zadávající katedra:	**Katedra matematiky**

Zásady pro vypracování

1. Pomocí statistických metod sumarizujte vstupní data.
2. Navrhněte model vstupních dat a jejich verifikace.
3. Formulujte optimalizační úlohu na základě požadavků výroby.
4. Navrhněte metodiku plánování výroby.

Rozsah diplomové práce: **40-80 stran**
Rozsah grafických prací: **dle potřeby**
Forma zpracování diplomové práce: **tištěná**

Seznam doporučené literatury:

- Reif, Jiří, Metody matematické statistiky, Plzeň: Západočeská univerzita 2000
- Vanderbei, Robert J., Linear programming : foundations and extensions, Springer 2020

Vedoucí diplomové práce: **Ing. Patrice Marek, Ph.D.**
Katedra matematiky

Datum zadání diplomové práce: **3. října 2022**
Termín odevzdání diplomové práce: **22. května 2023**

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 3. října 2022

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů. Tato práce nebyla využita k získání jiného nebo stejného akademického titulu.

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

V Plzni dne 18. května 2023

. . . . . .
Tomáš Krutina

Abstrakt

Tato diplomová práce se zaměřuje na plánování a optimalizaci výrobního procesu průmyslové lakovny. Jejím cílem je poskytnout ucelený pohled na problematiku plánování lakovny a navrhnout optimalizovaný model, který může být efektivně využit ke zlepšení výrobního procesu. V práci je popsán proces sběru dat a jejich následné statistické zpracování, které nám poskytlo vstupní datový model pro optimalizaci plánování. Dále jsme vytvořili model lineárního programování, který slouží k plánování rozložení skidů v painting trainu. Parametry modelu byly nastaveny na základě historických dat. Následně jsme model otestovali a naplánovali rozložení skidů lakovny pro nadcházející měsíc. Výsledky simulace potvrdily, že navržený model úspěšně plánuje rozložení painting trainu a minimalizuje počet výměn skidů. Výsledkem této práce je efektivní model plánování lakovny, který umožňuje rychlé nalezení optimálního rozložení skidů v painting trainu. Navržený model je prakticky použitelný a může být implementován v reálném provozu.

Abstract

This thesis focuses on planning and optimization of an industrial paint shop production process. It aims to provide a comprehensive view of the paint shop planning issues and to propose an optimized model that can be effectively used to improve the production process. The paper describes data collection process and its subsequent statistical processing, which provided an input data model for optimizing the planning process. Furthermore, we developed a linear programming model that is used to plan the skid layout in the painting train. Model parameters were set based on historical data. We then tested the model and scheduled the skid layout of the paint shop for the upcoming month. The simulation results confirmed that the proposed model successfully schedules the painting train layout and minimizes number of skid changes. The result of this work is efficient paint shop scheduling model that can quickly find the optimal skid layout in the painting train. The proposed model is practically applicable and can be implemented in real operation.

Klíčová slova

statistické zpracování dat • optimalizace • lineární programování

Poděkování

Rád bych na tomto místě poděkoval RNDr. Bc. Radimu Hoškovi, Ph.D. za jeho přínos při představení problematiky plánování průmyslových lakoven a návrhu tématu diplomové práce, a Ing. Patrice Markovi, Ph.D. za nekonečnou trpělivost a cenné rady, které mi poskytl při vedení diplomové práce.

Obsah

1 Úvod **3**

2 Teoretická část **5**
2.1 Testy normality 5
2.2 Jednovýběrový test středí hodnoty (t-test) 6
2.3 Wilcoxonův jednovýběrový test 7
2.4 Lineární programování 8
2.5 Simplexová metoda 10
2.6 Celočíselné lineární programování 12
2.7 Řešení úlohy CLP 13
2.8 Metoda větví a mezí 14
2.9 Metoda Gomoryho řezů 15

3 Charakteristika plánování **17**
3.1 Proces lakování 17
3.2 Vazba položka – skid 18
3.3 Plánování painting trainu 18
3.4 Parametry a omezení painting trainu 19
3.5 Strategie plánování 21

4 Příprava dat a plán řešení **23**
4.1 Skutečný stav datového prostředí 24
4.2 Obdržená data 25
4.3 Sběr dynamických dat 26
4.4 Plán řešení 26

5 Statistické zpracování dat **27**
5.1 Umístění a struktura dat 27
5.2 Sumarizace vstupních dat 28
5.3 Model vstupních dat 31
5.4 Verifikace modelu vstupních dat 36

6 Formulace optimalizační úlohy **39**
6.1 Tvorba modelu lineárního programování . . . 39
6.1.1 Základní model . . . 39
6.1.2 Velikost painting trainu . . . 40
6.1.3 První lakovací kolo . . . 40
6.1.4 Výměna skidů . . . 41
6.1.5 Počet výměn skidů . . . 41
6.1.6 Minimální počet skidů . . . 42
6.1.7 Požadavky na následující den . . . 42
6.1.8 Maximální počet výměn skidů . . . 44
6.1.9 Hledání řešení přes více dnů . . . 44
6.2 Definice parametrů . . . 45
6.2.1 Vstupní parametry . . . 45
6.2.2 Odvozené parametry . . . 46
6.2.3 Proměnné . . . 47
6.3 Omezení modelu . . . 47
6.4 Cílová funkce . . . 50

7 Simulace plánování **53**
7.1 AMPL . . . 53
7.2 Highs . . . 54
7.3 Testování funkčnosti modelu . . . 54
7.3.1 Testovací data a nastavení parametrů modelu . . . 54
7.3.2 Simulace plánování na testovacích datech . . . 55

8 Diskuze **59**
8.1 Vývoj modelu . . . 59
8.2 Rozšíření cílové funkce . . . 60
8.3 Hledání vzorců v citlivostní analýze . . . 60
8.4 Vícedenní předvýroba . . . 61
8.5 Automatizace . . . 62
8.6 Výstupní parametry . . . 62
8.7 Komerční solver . . . 63
8.8 Plánování položek . . . 63
8.9 Využití lineárního programování . . . 63

9 Závěr **65**

Seznam obrázků **71**

Seznam tabulek **73**

Úvod 1

Průmyslové lakovaní je klíčovou technologickou operací v mnoha průmyslových odvětvích, od automobilového a leteckého průmyslu po stavebnictví a nábytkářský průmysl. Hlavním úkolem lakování je aplikovat ochranné a dekorační nátěry na různé povrchy, aby se zlepšil jejich vzhled a ochrana proti korozi a opotřebení.

Lakování je jedním z nejsložitějších procesů v průmyslové výrobě. Plánování lakovny je do jisté míry optimalizací s velkým množstvím omezení. Plánování musí brát v úvahu různé faktory, jako jsou kapacita lakovny, počet výrobků, které mají být lakovány, typy laku a barvy, doba sušení a další. Plánování musí být velmi přesné a správné rozhodnutí může mít velký dopad na celkovou produktivitu a ziskovost podniku.

Většina průmyslových podniků však stále plánuje lakovny heuristicky a zdaleka ne všechna omezení, která během dlouhých let předchozí praxe vznikla, lze vnímat jako omezení pro optimalizaci. V opačném případě totiž hrozí, že množina přípustných stavů bude prázdná a nebude tedy existovat způsob, jakým bychom byli schopni splnit požadavky na výrobu a dodržet při tom všechna omezení. Pokud však v realitě nastane taková situace, plánovač jednoduše některá omezení poruší. Tato omezení ve skutečnosti často bývají požadovanou hodnotou cílové funkce.

Tato diplomová práce se zaměřuje na řešení problému plánování provozu průmyslové lakovny pro jednoho výrobce dílů pro automotive v Německu. V první části práce se věnujeme teoretickému úvodu, kde nejprve popisujeme statistické metody, konkrétně testy normality a jednovýběrové testy střední hodnoty. Následně se věnujeme základům lineárního programování, zejména Simplexové metodě, celočíselnému lineárnímu programování a způsobům řešení celočíselného lineárního programování.

Po teoretické části se zaměříme na charakteristiku plánování, kde představíme lakovnu a její procesy. Dále se budeme věnovat plánování painting trainu a jeho pa-

rametrům a různým omezením, která se k plánování vztahují. Další část práce se věnuje přípravě dat, kde nejprve popisujeme ideální stav dat a poté řešíme problémy, které mohou při získávání dat nastat. Následně jsou představena data, která jsme obdrželi od zákazníka, a způsob, jakým jsme je získávali.

Poté budeme statisticky zpracovávat získaná data. Nejprve je sumarizujeme a následně tvoříme model vstupních dat, který využijeme v plánování lakovny. V následující části formulujeme optimalizační úlohu v souladu s popsanými parametry a omezeními. Popíšeme si vývoj modelu a následně určíme parametry, omezení a cílovou funkci finální verze modelu. Poté se zaměříme na způsob řešení modelu a podrobně popíšeme a zhodnotíme výsledky.

V závěrečné části práce diskutujeme možné nedostatky a vylepšení modelu a jeho další využití v praxi. Cílem této práce je poskytnout ucelený pohled na plánování průmyslové lakovny a jeho optimalizaci, a to na konkrétním příkladu výrobce dílů pro automotive v Německu.

Teoretická část 2

Tato část práce poskytuje základní teoretický rámec, který je nezbytný pro porozumění a aplikaci praktické části našeho projektu. Tato kapitola se zaměřuje na přehled klíčových teoretických konceptů a definic pojmů, které budou použity v naší analýze a aplikovány v praktickém výzkumu. Cílem této části je poskytnout čtenáři potřebné teoretické základy, které budou sloužit jako rámec pro naši práci a umožní nám provést relevantní analýzu.

V první části této kapitoly se zaměříme na statistické metody, které jsou zpracovány dle Reifa (2004). Nejprve se podíváme na testy normality dat, dále si popíšeme jednovýběrový t-test, což je jednovýběrový test střední hodnoty a je možné ho zařadit do kategorie tzv. parametrických testů, u kterých se předpokládá, že model zkoumané náhodné veličiny známe až na hodnotu jednoho nebo několika parametrů a test se pak týká hodnot těchto parametrů. U neparametrických testů se nepředpokládá konkrétní model náhodné veličiny a testy mají velmi obecnou platnost. Často se pouze předpokládá, že jde o libovolnou veličinu spojitého typu. Jako zástupce neparametrických testů si představíme Wilcoxonův jednovýběrový test.

Ve druhé části této kapitoly se zaměříme na lineární programování. Tato část kapitoly je zpracována dle Vanderbei (2020). Nejprve si definujeme úlohu lineárního programování a řešení této úlohy, konkrétně řešení pomocí Simplexové metody. Poté se budeme zabývat celočíselným lineárním programováním, což je speciální případ lineárního programování. Definujeme si úlohu celočíselného lineárního programování a uvedeme si způsoby řešení, konkrétně metodu větví a mezí a metodu Gomoryho řezů.

2.1 Testy normality

V mnoha statistických metodách se předpokládá, že soubor, se kterým pracujeme, je náhodným výběrem z normálního rozdělení. Tento předpoklad lze ověřit pomocí testů dobré shody, které porovnávají náhodný výběr s rozdělením nějaké náhodné

veličiny, ale nevíme s jistotou, o jaké rozdělení se jedná. Můžeme zvolit hypotézu buď o druhu rozdělení (např. normální rozdělení), nebo o druhu rozdělení i hodnotách jeho parametrů (např. normální rozdělení se střední hodnotou $\mu = 0$ a směrodatnou odchylkou $\sigma = 2$). Mezi nejpoužívanější testy dobré shody patří Chí-kvadrát test dobré shody, Kolmogorův test nebo Lillieforsův test. V praxi se často používají grafické metody, jako je pravděpodobnostní graf, nazývaný také P-P graf, nebo kvantil-kvantilový graf (Q-Q graf), pro kontrolu shody dat s hypotetickým rozdělením. Existuje také řada dalších metod. V simulační studii Shapiro, Wilk a Chen (1968, str. 1343–1372) byla porovnána síla devíti různých testů normality pro náhodné výběry z různých rozdělení. Na základě výsledků této studie se začal často používat test navržený Shapirem a Wilkem, který na Q-Q grafu ověřuje lineární závislost, ale s jemnějšími prostředky než korelační koeficient. Bližší popis této metody lze nalézt v knize Lamoš a Potocký (1989, str. 157).

2.2 Jednovýběrový test středí hodnoty (t-test)

Nechť $x_1, x_2, \ldots, x_n$ je náhodný výběr z normálního rozdělení $N(\mu, \sigma^2)$, kde rozptyl σ^2 neznáme, a testujeme na hladině významnosti α oproti jednostranné nebo oboustranné alternativě hypotézu

$$H_0 : \mu = \mu_0, \tag{2.1}$$

kde μ_0 je známé číslo. Označme t hodnotu

$$t = \frac{\overline{x} - \mu_0}{s}\sqrt{n}, \tag{2.2}$$

kde s je výběrová směrodatná odchylka, kterou definujeme pro $n > 1$ jako

$$s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2}. \tag{2.3}$$

Věta 1 *(pro důkaz viz např. Likeš, Machek (1983), str. 22). Buďte $X_1, X_2, ..., X_n$ nezávislé náhodné veličiny, které mají všechny rozdělení pravděpodobnosti $N(\mu, \sigma^2)$. Pak pro výběrový rozptyl S^2 platí*

$$\frac{(n-1)S^2}{\sigma^2} \backsim \chi^2(n-1). \tag{2.4}$$

Věta 2 *(pro důkaz viz např. Likeš, Machek (1983), str. 22). Buďte U, Y dvě nezávislé náhodné veličiny, $U \backsim N(0,1)$ a $Y \backsim \chi^2(\nu)$. Pak veličina*

$$t = \frac{U}{\sqrt{Y/\nu}} \text{ má rozdělení } t(\nu). \tag{2.5}$$

Z věty 1 a věty 2 plyne, že za předpokladu platnosti hypotézy H_0 je t realizací veličiny $t(\nu)$ s počtem stupňů volnosti $\nu = n - 1$. Při oboustranném testu proto volíme kritický obor

$$|t| > t_{1-\alpha/2}(n-1), \tag{2.6}$$

při jednostranném testu pak kritický obor

$$t > t_{1-\alpha}(n-1), \tag{2.7}$$

resp.

$$t < -t_{1-\alpha}(n-1). \tag{2.8}$$

Výpočetní SW poskytují informaci o nejmenší hladině, při které bychom hypotézu H_0 zamítli, tato nejmenší hladina se nazývá p-hodnota testu. Pokud je $p \leq \alpha$ (obvykle $\alpha = 0,05$ nebo $\alpha = 0,01$), znamená to, že pravděpodobnost, že jsme, za předpokladu platnosti H_0, získali takové extrémní výsledky náhodou, je velmi malá, což nám poskytuje dostatečný důkaz pro zamítnutí nulové hypotézy. Naopak, pokud je $p > \alpha$, nemůžeme zamítnout nulovou hypotézu, protože stále existuje pravděpodobnost p vyšší než α, že dané výsledky jsou náhodné a nemáme dostatek důkazů k tomu, abychom hypotézu zamítli.

Přijímání nebo zamítání hypotézy tedy závisí na hladině významnosti a p-hodnotě. Pokud p-hodnota není větší než zvolená hladina významnosti, zamítáme nulovou hypotézu a přijímáme alternativní hypotézu. Pokud je p-hodnota větší než zvolená hladina významnosti, nemůžeme zamítnout nulovou hypotézu.

2.3 Wilcoxonův jednovýběrový test

Buď $Z_1, Z_2, ..., Z_n$ náhodný výběr ze spojitého rozdělení pravděpodobnosti a testujeme hypotézu

$$H_0 : \tilde{Z} = c. \tag{2.9}$$

Uvažujme rozdíly $Z_i - c (i = 1, \ldots, n)$, které seřadíme tak, aby jejich absolutní hodnoty byly neklesající. Označíme W^+ součet pořadí odpovídajících kladným rozdílům a W^- součet pořadí odpovídajících záporným rozdílům. Jestliže se některé absolutní hodnoty navzájem rovnají a takových rovností není příliš mnoho, lze použít průměrných pořadí.

Testujeme-li H_0 proti oboustranné alternativě

$$H_1 : \tilde{Z} \neq c, \tag{2.10}$$

použijeme jako testovací kritérium $W = \min(W^+, W^-)$.

Při jednostranném dolním testu testujeme hypotézu H_0 proti jednostranné alternativě

$$H_1 : \tilde{Z} < c, \tag{2.11}$$

použijeme jako testovací kritérium $W = W^-$.

Při jednostranném horním testu testujeme hypotézu H_0 proti jednostranné alternativě

$$H_1 : \tilde{Z} > c, \tag{2.12}$$

použijeme jako testovací kritérium $W = W^+$.

Vypočtené testovací kritérium W porovnáme s tabelovanou kritickou hodnotou pro příslušný rozsah výběru n a zvolenou hladinu významnosti α. Hypotézu H_0 pro jednostrannou variantu zamítáme, je-li

$$W < W(\alpha, n). \tag{2.13}$$

Hypotézu H_0 pro oboustrannou variantu zamítáme, je-li

$$W < W(\alpha/2, n). \tag{2.14}$$

Pokud pro konkrétní případy nerovnosti 2.13 nebo 2.14 neplatí, hypotézu H_0 nelze zamítnout.

Podobně jako u t-testu z kapitoly 2.2 obdržíme od SW informaci o p-hodnotě. Hypotézu H_0 zamítáme, pokud je $p \leq \alpha$. Obvykle se pro hladinu významnosti α volí hodnoty 0, 05 nebo 0, 01 a požaduje se, aby pravděpodobnost chyby 1. druhu při testování hypotézy H_0 nepřesáhla α. Chyby 1. druhu se dopustíme v případě, kdy platí hypotéza H_0, ale my ji na základě experimentálně zjištěných hodnot zamítneme. Pokud p-hodnota není větší než zvolená hladina významnosti, znamená to, že jsme se dostali do oblasti extrémních výsledků a pravděpodobnost, že jsme je za platnosti H_0 získali náhodou, je velmi nízká. V takovém případě zamítáme nulovou hypotézu ve prospěch alternativní hypotézy. Pokud je p-hodnota větší než hladina významnosti, nulovou hypotézu nezamítáme. Pro normální rozdělení je Wilcoxonův test téměř stejně silný jako t-test.

2.4 Lineární programování

Lineární programování je matematická metoda, která se používá k optimalizaci (maximalizaci nebo minimalizaci) lineární cílové funkce za splnění lineárních omezení

na hodnoty proměnných. Úloha lineárního programování je konkrétní formulace problému, která se snaží nalézt optimální řešení pomocí lineárního programování. Formálně je úloha lineárního programování definována následovně:

$$\text{Maximalizujeme:} \quad \mathbf{c}^{\mathrm{T}}\mathbf{x} \tag{2.15}$$

$$\text{za podmínky:} \quad \mathbf{A}\mathbf{x} \leq \mathbf{b} \tag{2.16}$$

$$\mathbf{x} \geq \mathbf{0}, \tag{2.17}$$

kde

c je vektor koeficientů cílové funkce, kterou chceme maximalizovat,
x je vektor hledaných proměnných,
A je matice koeficientů omezení a
b je vektor pravých stran omezení, který udává hodnoty omezení.

Cílem úlohy lineárního programování je nalézt hodnoty proměnných **x**, které splňují všechna lineární omezení 2.16 a 2.17 a zároveň maximalizují hodnotu cílové funkce 2.15, nebo minimalizují opačnou hodnotu cílové funkce 2.15. Takto formulovaná úloha lineárního programování je označována jako úloha ve standardním tvaru. Řešením této úlohy je návrh konkrétních hodnot proměnných $x_1, x_2, ..., x_n$. Řešením úlohy lineárního programování se lze dostat do následujících situací:

- Přípustné řešení úlohy lineárního programování je množina hodnot proměnných $x_1, x_2, ..., x_n$, které splňují omezení 2.16 a 2.17.
- Optimální řešení je takové řešení, které dosahuje nejlepší možné hodnoty cílové funkce. Formálně, $\mathbf{x}^*$ je optimální řešení, pokud platí

  $$\mathbf{c}^{\mathrm{T}}\mathbf{x}^* \geq \mathbf{c}^{\mathrm{T}}\mathbf{x} \tag{2.18}$$

  pro všechna **x**, která splňují podmínky 2.16 a 2.17.
- Neomezené řešení nastává, pokud hodnota cílové funkce 2.15 může být nekonečně velká (při maximalizaci) nebo nekonečně malá (při minimalizaci), a není dosažitelná žádnou konečnou hodnotou proměnných $x_1, x_2, ..., x_n$.
- Nedosažitelné řešení nastává, pokud neexistuje žádné řešení, které by splňovalo omezení 2.16 a 2.17, čili neexistuje přípustné řešení.

Řešení lineárního programování mohou být nalezena různými algoritmy, jako je Simplexová metoda, Metoda vnitřních bodů nebo Metoda řezů, které se používají pro hledání optimálního řešení nebo pro určení, že problém nemá řešení, nebo je neomezený.

V další kapitole se podíváme na řešení úlohy lineárního programování pomocí Simplexové metody.

2.5 Simplexová metoda

Simplexová metoda je postup, kterým se iterativně hledá optimální řešení úlohy lineárního programování. Mějme úlohu lineárního programování ve standardním tvaru:

$$\text{Maximalizujeme:} \quad \mathbf{c}^T x \tag{2.19}$$

$$\text{za podmínky:} \quad \mathbf{A}\mathbf{x} \leq \mathbf{b} \tag{2.20}$$

$$\mathbf{x} \geq \mathbf{0}, \tag{2.21}$$

kde

c je vektor koeficientů cílové funkce, kterou chceme maximalizovat,
x je vektor hledaných proměnných,
A je matice koeficientů omezení a
b je vektor pravých stran omezení, který udává hodnoty omezení.

Postup jednofázové simplexové metody pro úlohu s nezápornou pravou stranou omezení lze vyjádřit následovně:

1. Výchozí základní řešení úlohy LP:
   Omezující podmínky jsou zadané ve formě nerovnic a je nejprve potřeba je převést na rovnice pomocí přídatných proměnných, které přičteme k levé straně nerovnic. Získáme tím tzv. ekvivalentní soustavu rovnic, která bude v kanonickém tvaru, tj. ve tvaru, kdy matice koeficientů obsahuje jednotkovou matici. V kanonickém tvaru jsou dva druhy proměnných – ty proměnné, u kterých jsou jednotkové vektory, se označují jako základní (bazické) proměnné, ostatní jsou nezákladní (nebazické). Každému kanonickému tvaru ekvivalentní soustavy rovnic odpovídá základní řešení úlohy LP, které se snadno získá tak, že pro každý prvek báze **B** nastavíme hodnotu bazické proměnné $x_b = b$ a nebazické proměnné nastavíme $x_j = 0$.

2. Výpočet redukovaných nákladů:
   Pro každou nebazickou proměnnou x_j spočítáme její redukovaný náklad r_j podle vzorce

   $$r_j = c_j - \mathbf{c}_\mathrm{B}^\mathrm{T}\mathbf{B}^{-1}\mathbf{A}_\mathrm{j}, \tag{2.22}$$

   kde

   c_j je j-tý prvek vektoru c,
   $\mathbf{c}_\mathrm{B}^\mathrm{T}$ je transponovaný vektor $\mathbf{c}_\mathrm{B}$, kde
   $\mathbf{c}_\mathrm{B}$ jsou prvky vektoru c odpovídající bázi **B**, a
   $\mathbf{A}_\mathrm{j}$ je j-tý sloupec matice **A**.

3. Test optimality:
   Pokud jsou všechny redukované náklady nezáporné nebo rovny nule, dosáhli jsme optimálního řešení a algoritmus končí.

4. Výběr vstupního řádku a výstupního sloupce:
   Nejprve vybereme výstupní sloupec, který odpovídá sloupci s nejmenším redukovaným nákladem r_j. Poté vybereme vstupní řádek, který odpovídá omezení s nejmenším poměrem pravé strany k hodnotě odpovídajícímu redukovanému nákladu r_j, a označíme jej jako pivota. Nechť k je řádkový index vybraného řádku. Potom

$$k = \arg\min_{i|a_{ij}>0}(b_i/a_{ij}), \tag{2.23}$$

   kde
   i je index řádku a
   j je index výstupního sloupce.

5. Úprava tabulky:
   Provádíme pivotální operace, které upraví hodnoty proměnných v bázi **B** tak, aby nové řešení splňovalo omezení a bylo optimální. Tím získáme nové přípustné řešení. Konkrétně výměnou bazické a nebazické proměnné upravíme hodnoty x_b a x_j podle pravidla

$$x'_b = x_b - qA_i \tag{2.24}$$
$$x'_j = q, \tag{2.25}$$

   kde
   q je hodnota pivota,
   A_i je hodnota prvku v pivotálním řádku,
   x_b jsou hodnoty bazické proměnné před pivotální operací,
   x_j jsou hodnoty nebazické proměnné před pivotální operací.

6. Návrat na krok 2:
   Opakujeme kroky 2–5, dokud nedosáhneme optimálního řešení, což je stav, kdy jsou všechny redukované náklady nezáporné nebo rovny nule.

To je základní postup Simplexové metody. Samozřejmě, existuje mnoho optimalizací a vylepšení této metody, jako je duální simplexová metoda, dvoufázová simplexová metoda atd. Podrobnější informace k metodám nalezneme např. ve Vanderbei (2020) nebo Jablonský (2011).

2.6 Celočíselné lineární programování

Celočíselné lineární programování (CLP) je varianta lineárního programování, kde navíc požadujeme, aby hodnoty proměnných byly celočíselné. To zavádí další omezení do optimalizačního problému, které může mít významné dopady na výsledky a výpočetní náročnost.

Celočíselné lineární programování najde široké uplatnění v mnoha reálných problémech. Následující jsou některé oblasti, kde se CLP používá:

- CLP se často používá pro plánování a rozvrhování v oblastech jako je výroba, distribuce, logistika, doprava, řízení zásob a projektový management. Například při plánování výroby je často nutné optimalizovat alokaci zdrojů, např. strojů, pracovníků a materiálů, s omezeními na jejich dostupnost a celočíselnými omezeními na počet strojů nebo pracovníků, které je možné použít.

- CLP se používá v oblasti finančního rozhodování, například při plánování investic, portfoliovém managementu, rozdělování zisků nebo stanovování cen. Například při optimalizaci portfolia investic je možné použít CLP pro nalezení optimální alokace prostředků do různých investičních instrumentů s omezeními na jejich váhu v portfoliu a celočíselnými omezeními na počet jednotlivých instrumentů.

- CLP se používá v oblasti telekomunikací a sítí pro optimalizaci plánování sítí, routingových a přenosových problémů, plánování kapacity a optimalizaci služeb. Například při optimalizaci sítě je možné použít CLP pro nalezení optimálního plánu sítě s omezeními na kapacitu spojů, celočíselnými omezeními na počet spojů nebo routovacími omezeními.

- CLP se používá v oblasti logistiky a dopravy pro optimalizaci plánování trasy, plánování rozvozu nebo plánování dodávek. Například při plánování trasy pro doručování zásilek je možné použít CLP pro nalezení optimální trasy s omezeními na kapacitu vozidel, počet zásilek na trase a celočíselnými omezeními na počet zastávek nebo trasy.

CLP je silný matematický nástroj pro řešení optimalizačních problémů s celočíselnými omezeními a nachází uplatnění v různých oblastech, kde je nutné hledat optimální hodnoty proměnných s dodatečným omezením celočíselnosti.

Mějme následující úlohu CLP:

$$\text{Maximalizujeme:} \quad \mathbf{c}^{\mathrm{T}}\mathbf{x} \tag{2.26}$$

$$\text{za podmínky:} \quad \mathbf{A}\mathbf{x} \leq \mathbf{b} \tag{2.27}$$

$$\mathbf{x} \geq \mathbf{0} \tag{2.28}$$

$$\mathbf{x} \in \mathbb{Z}^n, \tag{2.29}$$

kde

c je vektor koeficientů cílové funkce, kterou chceme maximalizovat,
x je vektor hledaných proměnných,
A je matice koeficientů omezení a
b je vektor pravých stran omezení, který udává hodnoty omezení.

Cílem CLP je nalézt takový vektor **x**, který maximalizuje hodnotu cílové funkce 2.26 za splnění všech omezení. Omezení celočíselnosti hodnot proměnných je to, co odlišuje CLP od klasického lineárního programování, kde hodnoty proměnných jsou obecně reálné.

2.7 Řešení úlohy CLP

Existuje několik metod, které lze použít k řešení úlohy CLP. Mezi nejběžnější metody patří:

- Úplné prohledávání (exaktní metody):
  Tato metoda vyžaduje, aby byly vyzkoušeny všechny možné kombinace hodnot celočíselných proměnných, což je v praxi často nepraktické pro velké a složité problémy, protože počet kombinací může být enormní.

- Metoda větví a mezí (branch and bound):
  Tato metoda se zakládá na rozdělení problému na menší podproblémy pomocí větvení na různé hodnoty jedné z proměnných. Pro každý podproblém se pak použije dolní mez na hodnotu objektivní funkce a neperspektivní větve jsou odřezány (pruning). Tím se snižuje počet zkoumaných kombinací, což umožňuje efektivnější řešení problému.

- Metoda Gomoryho řezů:
  Tato metoda se zakládá na postupném generování nových omezení (řezů) na základě nepovolených nebo nedostatečně povolených celočíselných hodnot v optimálním řešení. Tato nová omezení se pak přidávají do původního modelu, čímž se omezují možné hodnoty proměnných a zlepšuje se relaxační dolní mez na hodnotu objektivní funkce.

- Metoda dynamického programování:
  Tato metoda se používá pro specifické typy úloh CLP, které mají určitou strukturu, jako například problémy s opakujícími se vzory nebo problémy s diskrétními časovými okamžiky. Metoda dynamického programování umožňuje efektivní vyhledávání optimálního řešení tím, že rozděluje problém na menší podproblémy a ukládá si výsledky, které se používají k optimalizaci celkového řešení.

- Heuristické metody:
  Tyto metody se snaží najít rychlá přibližná řešení problému pomocí určitých pravidel nebo technik. Příklady heuristických metod zahrnují metodu hrubé síly (brute-force), metodu blízkého souseda (nearest neighbor), genetické algoritmy (genetic algorithms), simulované ochlazování (simulated annealing) nebo prohledávání s omezeným seznamem (limited list search).

Každá z těchto metod má své výhody a nevýhody a jejich volba závisí na charakteru konkrétního problému, jeho velikosti, složitosti a požadované přesnosti výsledků. Někdy může být také efektivní kombinace různých metod nebo využití specializovaného softwaru, který obsahuje pokročilé algoritmy a optimalizační techniky, které umožňují efektivní a přesné řešení problému. Mezi populární komerční software pro řešení úloh CLP patří například CPLEX, Gurobi nebo Xpress.

V další části si podrobněji popíšeme metodu větví a mezí a metodu Gomoryho řezů.

2.8 Metoda větví a mezí

Mějme dánu úlohu celočíselného lineárního programování (CLP) ve standardním tvaru definovanou v kapitole 2.6. Metoda větví a mezí řeší tuto úlohu CLP následujícím způsobem:

1. Relaxace:
   Nejprve se provede relaxace původní úlohy CLP, tedy najde se optimální řešení relaxované úlohy lineárního programování (LP), kde x je omezeno na reálné hodnoty místo celočíselných hodnot.

2. Test celočíselnosti:
   Pokud je nalezené optimální řešení úlohy LP celočíselné, jedná se o optimální řešení původní úlohy CLP a metoda končí. Jinak pokračujeme na krok 3.

3. Větvení:
   Vybereme jednu proměnnou x_i, která nemá celočíselnou hodnotu v optimálním řešení úlohy LP, a rozdělíme ji na dvě větve: jedna větev s omezením

$x_i \geq \lfloor x_i \rfloor$ (dolní celou hodnotou) a druhá větev s omezením $x_i \geq \lceil x_i \rceil$ (horní celou hodnotou).

4. Řešení podproblémů:
   Pro každou větev z kroku 3 opakujeme kroky 1 až 3, dokud nejsou nalezena celočíselná optimální řešení pro všechny proměnné.

5. Ořízntutí větví:
   Během větvení a řešení podproblémů udržujeme nejlepší dosud nalezené celočíselné řešení a průběžně aktualizujeme dolní mez, která je rovna hodnotě nejlepšího dosud nalezeného celočíselného řešení. Pokud je hodnota optimálního řešení některého podproblému menší než dosud nalezená dolní mez, můžeme tento podproblém oříznout, protože nemůže obsahovat lepší celočíselné řešení než dosud nalezené.

6. Ukončení:
   Metoda končí, když jsou vyčerpány všechny možnosti větvení nebo je dokázáno, že hodnota optimálního řešení je menší než dolní mez. V takovém případě je dosud nalezené celočíselné řešení považováno za globální optimální řešení původní úlohy CLP.

Metoda větví a mezí je efektivní algoritmus pro řešení úloh celočíselného lineárního programování, protože systematicky prohledává prostor možných řešení a provádí ořezávání neperspektivních větví, čímž zkracuje výpočetní čas a zvyšuje efektivitu hledání optimálního řešení. Více o metodě větví a mezí viz Wolsey (1998, str 98).

2.9 Metoda Gomoryho řezů

Mějme dánu úlohu celočíselného lineárního programování (CLP) ve standardním tvaru definovanou v kapitole 2.6. Metoda Gomoryho řezů spočívá v postupném hledání a přidávání Gomoryho řezů do původní úlohy CLP, které omezují hodnoty neceločíselných proměnných na celočíselné hodnoty. Postup je následující:

1. Získání relaxovaného řešení:
   Řešíme původní úlohu CLP bez omezení celočíselnosti proměnných, tedy bez podmínky x je celočíselné. Získáme relaxované řešení $\mathbf{x}^* = (x_1^*, x_2^*, ..., x_n^*)$, kde x_i^* je hodnota i-té proměnné v relaxovaném řešení.

2. Identifikace neceločíselných proměnných:
   Procházíme hodnoty relaxovaného řešení $\mathbf{x}^*$ a identifikujeme proměnné x_i^*, které mají neceločíselné hodnoty, tedy x_i^* není celé číslo.

3. Sestavení Gomoryho řezu:
   Rozdíl mezi dolní celou hodnotou zaokrouhlené hodnoty x_i^* dolů na nejbližší celé číslo ($\lfloor x_i \rfloor$) a původní hodnotou x_i^* je základní hodnota Gomoryho řezu pro proměnnou x_i^*. Sestavíme Gomoryho řez ve formě lineární rovnice, který omezuje hodnotu proměnné x_i^* na zaokrouhlenou hodnotu $\lfloor x_i \rfloor$, tedy:

   $$x_i - \lfloor x_i \rfloor \leq 0 \tag{2.30}$$

   nebo

   $$x_i \leq \lfloor x_i \rfloor. \tag{2.31}$$

   Tato rovnice je novým omezením, které přidáváme do původního CLP jako Gomoryho řez.

4. Aktualizace původní úlohy CLP:
   Přidáme nově sestavený Gomoryho řez do původního CLP jako další omezení a opět ho řešíme bez omezení celočíselnosti proměnných. Pokud nově řešené CLP má neceločíselné řešení, opakujeme kroky 1– 3, dokud nenajdeme celočíselné optimální řešení.

Tímto postupem pokračujeme v hledání a přidávání Gomoryho řezů do původní úlohy CLP, dokud nedosáhneme celočíselného optimálního řešení. Pro bližší informace o metodě Gomoryho řezů viz Wolsey (1998, str. 124). Popis dalších metod řešení úloh CLP nalezneme také v Wolsey (1998).

Charakteristika plánování

3

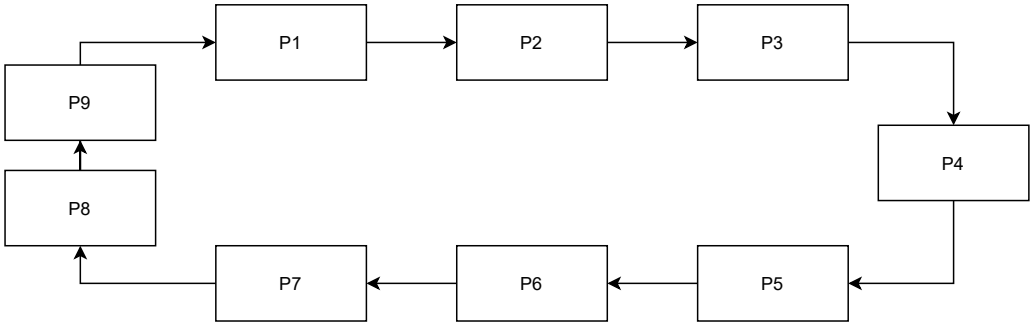
Lakovací linku si lze představit jako lanovku. Lakovnou projíždějí jednotlivé skidy, na které se zavěšují lakované položky. Po posledním skidu opět přijde na řadu ten první. Posloupnost těchto skidů se označuje jako painting train, tento painting train se stihne otočit několikrát za den a každá otočka je označována jako lakovací kolo. Rozložení painting trainu není fixní, dá se z části měnit na tzv. nádraží, kde je možné libovolný skid z painting trainu vyjmout a jiný skid zase zařadit, přičemž nový skid můžeme zařadit na libovolné místo v painting trainu. Lakovací linka je v provozu od 6:00 až do 22:00. Rozložení painting trainu v prvním lakovacím kole daného dne je stejné jako rozložení painting trainu v posledním lakovacím kole předchozího dne (přes noc je lakovací linka vypnutá a v painting trainu neprobíhají žádné změny).

3.1 Proces lakování

Na obrázku 3.1 je zobrazen lakovací proces lakovací linky, kde následující procesy jsou:

P1 navěšování položek na skidy,
P2 čištění položek,
P3 sušení položek,
P4 nános primárního laku,
P5 kontrola,
P6 nanášení základního laku,
P7 nanášení bezbarvého laku,
P8 přijetí a sundání nalakovaných položek ze skidů a
P9 nádraží, na kterém probíhá výměna skidů v painting trainu.

Na celý proces lakování však můžeme nahlížet jako na black box, jelikož jediné místo, na kterém můžeme proces lakování ovlivnit, je nádraží – tzn. rozložení položek v painting trainu.



Obrázek 3.1: Lakovací proces lakovací linky.

3.2 Vazba položka – skid

Každé ráno lakovna dostane odvolávky[1] zákazníků na jednotlivé položky (přesněji řečeno požadavky na lakování převedené pomocí toku materiálu z odvolávek zákazníků). Lakovací linkou ovšem neprojíždějí položky samostatně, ale jsou ve větším počtu umístěné na skidech. Pro účely lakovny je tedy potřeba znát požadavky na jednotlivé skidy.

V lakovně se používá přibližně 200 druhů věšáků, tzv. skidtype, přičemž k dispozici máme určitý počet jednotlivých skidtypů. Každý skidtype má k sobě jednoznačně přiřazené položky, které je možné na daný skidtype umístit. Položky na skidu není možné kombinovat, v jednu chvíli tedy může být na skidu umístěn pouze jeden druh položky a každý skidtype nese informaci o tom, kolik má věšáků čili kolik kusů dané položky lze na skidtype umístit. Na obrázku 3.2 je zobrazeno lakování zadních nárazníků umístěných na skidech o kapacitě 6 plozek.

3.3 Plánování painting trainu

Předpokládejme, že jsme z požadavků na položky vypočetli požadavky na počet jednotlivých skidtypů, který musí během dne projet lakovnou. Cílem plánování painting trainu je tedy vytvoření posloupnosti skidů, které během dne projedou s navěšenými položkami lakovnou, aby tím byly splněny požadavky na skidtypy čili i požadavky na položky.

Lakovací linku je možné libovolně zrychlit, zpomalit nebo dokonce i zastavit, což ovlivňuje počet lakovacích kol během dne. Při poruše je potřeba lakovací linku zastavit. Při výměně skidů v painting trainu (vyjmutí nebo vložení skidu) je možné

[1] Jde o specifikaci požadovaných dodávek poskytující výhled na delší období, která je základním podkladem pro plánování výroby a nákup.

Obrázek 3.2: Lakování zadních nárazníků. Zdroj: Plastics Painting (2005).

lakovací linku zpomalit a posléze opět zrychlit. Výměna skidů je náročný proces a počet výměn tedy značně ovlivňuje rychlost lakovny, a tedy i počet lakovacích kol a množství nalakovaných položek. Z tohoto důvodu je naším cílem najít pro každé lakovací kolo takové rozložení painting trainu, aby počet výměn mezi jednotlivými lakovacími koly byl co nejnižší, kdy pod výměnou skidu rozumíme vyjmutí nebo vložení skidu do painting trainu.

V painting trainu je velké množství skidů a ne na všechny z nich musíme každý den obdržet požadavky od zákazníků. Pokud na určitý skidtype nepřijdou žádné požadavky, není nutné ho vyjímat z painting trainu. Skidy projedou lakovnou prázdné, aniž bychom na ně navěsili jakékoliv položky.

3.4 Parametry a omezení painting trainu

Tato kapitola se věnuje parametrům a omezením používaných při pánování painting trainu. Tyto informace byly získány od našeho zákazníka a jsou klíčové pro úspěšné plánování a provoz lakovny. V následující části je uveden seznam všech parametrů a omezení.

- Počet typů výrobků je přibližně 800.
- Počet skidtypů je přibližně 200.
- Je dán počet dostupných skidů od každého skidtypu.
- Počet lakovacích kol, tj. kolikrát projede painting train lakovnou během dne, je přibližně 4,5–5.
- Velikost painting trainu, tj. počet skidů v jednom lakovacím kole, je mezi hodnotami 195–210.
- Skidy stejného typu (stejný skidtype) jsou v painting trainu umístěny vedle sebe.
- Je dán minimální a maximální počet skidtypů, který může být v painting trainu umístěn za sebou – maximální počet je dán celkovým počtem dostupných skidtypů, minimální počet je většinou roven 0 a je větší než 0 v případě, kdy se jedná o pravidelně využívaný skidtype, čili není vhodné tento skidtype z painting trainu vyřazovat.
- Vložení skidu do painting trainu nemusí být na stejné místo, ze kterého byl právě jiný skid vyjmut, ale na libovolné místo v painting trainu (po vyjmutí skidu se následující skidy mohou posunout dopředu a mezeru tím vyplnit, v případě vkládání nového skidu je možné určitý skid v painting trainu pozastavit, aby se před ním vytvořila mezera a bylo tak možné vložit nový skid).
- Maximální počet vyřazených skidů v jednom lakovacím kole je 12.
- Maximální počet zařazených skidů v jednom lakovacím kole je 10.

Další omezení se vztahují k barvám lakovaných položek. Některé barvy, které je pro operátory těžké mezi sebou rozeznat pouhým okem (např. diamantové bílá a polárně bílá) se nesmí nacházet v painting trainu vedle sebe a musejí mezi sebou mít rozestup alespoň 20 skidů. V lakovně jsou k dispozici hlavní lakovací stroje a sekundární lakovací stroje. Hlavní lakovací stroje slouží k lakování nejčastěji používaných barev a každý stroj obsahuje pouze jednu barvu (barva v těchto strojích se nemění). Sekundární lakovací stroje jsou k dispozici 3 a slouží k lakování ojediněle používaných barev. V těchto strojích se barvy mění dle potřeby lakovny, ale výměna barev trvá 20 minut. Pro výměnu barev v sekundárních lakovacích strojích existují další omezení, kdy u některých dvojic barev není možné přejít z jedné barvy na druhou.

3.5 Strategie plánování

Během konzultací s plánovačem vyšlo najevo, že splnění požadavků na výrobu je hlavní prioritou a v případě, kdy není z důvodu omezení výměn skidů možné požadavky splnit, provede se podle nutnosti vyšší počet změn. Počet vyřazení a zařazení skidů do painting trainu je tedy spíše cílovou funkcí než omezením. Nastane-li situace, kdy lakovna není plně vytížena, je možné začít plnit požadavky na následující den. Více jak na jeden den dopředu však z důvodu omezení kapacity skladu lakovat nelze.

Z důvodu, že máme velké množství položek a cílová funkce je závislá pouze na výměně skidů, nikoli na výrobku samotném, navrhujeme strategii, která nejprve optimalizuje složení painting trainu v jednotlivých lakovacích kolech (jde vlastně o optimalizaci na úrovni tříd výrobků, které používají stejné skidy). Tím pádem nejsme schopni aplikovat omezení vztahující se k barvám lakovaných položek a teprve na nalezeném rozložení painting trainu se snažíme najít proveditelné řešení (které v dosavadní praxi nebývá problém najít). Tato strategie je nezbytná s ohledem na dostupný výpočetní výkon a potřebu najít vhodné řešení nejpozději v řádu několika málo minut.

Příprava dat a plán řešení

4

V další kapitole se blíže podíváme na proces získávání dat od zákazníků. Popíšeme si nejprve obecné problémy spojené se získáváním dat a následně se podíváme na konkrétní data využívaná v této práci.

Příprava a oprava dat je klíčovou částí každého projektu, která je často náročná na čas a vyžaduje důkladnou komunikaci se zákazníkem. Správná příprava, kontrola a oprava dat je nezbytná pro zajištění kvality a spolehlivosti výsledků projektu. Příprava dat zahrnuje sběr, kontrolu a organizaci dat, která budou použita v projektu. To může zahrnovat extrakci dat z různých zdrojů, jako jsou interní databáze, externí zdroje dat, nebo data zákazníků. Následně je nutné data kontrolovat, čistit a upravovat tak, aby byla připravena pro zpracování v projektu. To může zahrnovat opravu chyb, odstranění duplicit, normalizaci formátů, oříznutí nepotřebných nebo doplnění potřebných informací.

Oprava dat je klíčovým krokem, který zajišťuje, že data jsou přesná, kompletní a kvalitní. Nalezení a korekce chyb v datech může být náročné a vyžaduje odborné znalosti a dovednosti. Oprava dat může zahrnovat kontrolu validity dat, detekci a opravu chybných hodnot, vyplnění chybějících hodnot nebo začlenění datových pravidel a standardů. Komunikace se zákazníkem je také klíčovým aspektem práce s daty. Správné porozumění požadavkům na data, která jsou potřebná pro projekt, je nezbytné pro jejich přesnou a efektivní přípravu. Komunikace se zákazníkem tedy zahrnuje jasnou specifikaci dat, diskuzi nad možnými způsoby přípravy dat a následné požadavky na opravu chybně dodaných dat.

Správná příprava a oprava dat má klíčový vliv na úspěch celého projektu. Kvalitní vstupní data jsou základem pro obdržení kvalitních výsledků. Nedostatečně připravená nebo neopravená data mohou vést k nesprávným výsledkům, špatnému rozhodování a ztrátě důvěryhodnosti projektu.

4.1 Skutečný stav datového prostředí

Většina zákazníků je přesvědčena, že jsou jejich data ve skvělém stavu. Opak však bývá pravdou. Existuje více způsobů, jak data udržovat. Jeden z nejhorších způsobů, jak jsou data udržována, je v hlavách plánovačů.

S léty zkušeností plánovače výroby může dojít k určitým změnám ve způsobu, jakým je výroba plánována a organizována. Tyto změny mohou být založeny na intuici, znalostech a zkušenostech plánovače, které nejsou formálně zaznamenány a dokumentovány. Místo toho jsou tyto informace pouze v hlavě plánovače, což může mít určité důsledky pro celkovou efektivitu a transparentnost plánovacího procesu. Plánovač s dlouholetou zkušeností může lépe předvídat možné problémy nebo přizpůsobovat plán výroby na základě svých znalostí. Jedná se například o výrobní kapacity, kapacity skladů nebo obalů a další omezení.

Dalším příkladem, jak mohou být data udržována jsou různé textové nebo excelovské soubory. V tomto případě máme již data zdokumentována, ale je s tím spojena řada rizik. Každý textový soubor může být uložen na jiném místě a data tak nejsou držena a spravována hromadně. Existuje velké riziko, že ať už omylem, nebo záměrně bude textový soubor smazán, přejmenován, přesunut, nebo data v něm budou upravena či smazána. V takovém případě se může stát, že nebude správně fungovat integrace na jiný systém využívající tato data, nebo dokonce data mohou být ztracena. Bývá poměrně složité, někdy až nemožné, dohledat příčinu a nastalou chybu opravit.

Jedním z nejlepších způsobů, jak společnosti udržují data, je v ERP systému, což je systém pro plánování podnikových zdrojů. Příkladem mohou být systémy SAP, QAD nebo HELIOS. Z těchto systémů je celkem jednoduché data dostat do databáze a dále s nimi pracovat. Dále je možná kontrola nad různými změnami, kdy každá změna je po určitou dobu archivována. Samotný systém však nedokáže vyřešit všechny problémy. Důležitá jsou hlavně data, která jsou v systému udržována.

Nejčastějšími chybami, se kterými se v datech setkáváme jsou duplikované řádky, špatně zadané záznamy, kdy například v číselném parametru jsou zadány jiné znaky, chybějící záznamy, nebo dokonce chybějící celé parametry, které jsou důležité pro plánování. V takových případech je potřeba, aby dal zákazník data do pořádku a všechny nedostatky opravil.

4.2 Obdržená data

V rámci plánování lakoven je potřeba obdržet parametry lakovny, požadavky na lakovnu a informace o jednotlivých položkách a skidech. Data u tohoto konkrétního zákazníka z Německa nebyla v dostatečné kvalitě. Malá část dat byla spravována v systému Legacy, většina dat se nachází v textových souborech a některé podstatné informace byly pouze v hlavě plánovače výroby.

Ze strany zákazníka byly alokovány pro přípravu dat nedostatečné kapacity a část projektu týkající se přípravy dat se kvůli tomu značně protáhla. Postupně jsme obdrželi požadavky na jednotlivé položky na dobu jednoho týdne a data týkající se skidů. Konkrétně se jednalo o seznam všech skidů, počet dostupných skidů daného skidtypu a kapacitu daného skidtypu. Jednu z nejdůležitějších informací, kterou je napojení položky na skid, tedy na jaké skidtypy se dají pověsit jednotlivé položky, jsme však nikdy neobdrželi. Tato informace se nachází v systému Legacy, ale dle zákazníka neexistuje jednoduchý způsob, jak tato data ze systému exportovat a díky nedostatku alokované kapacity ze strany zákazníka se tímto problémem nemá kdo zabývat a požadovaná data tedy neobdržíme včas.

Bez této informace však není možné lakovny plánovat. Kromě uvedených problémů se vedení společnosti navíc rozhodlo migrovat všechna data do nového ERP systému SAP. Migrací dat do nového systému by se měla zlepšit i jejich kvalita. Jelikož se tedy pracuje na implementaci nového systému, nedává smysl pokračovat v integraci na staré systémy a celý projekt je tak pozastaven a bude pokračovat po zavedení nového systému.

Pro testování funkčnosti modelu jsme se tedy rozhodli využít data z jiné lakovny, která má podobné parametry, omezení a princip plánování jako námi řešená lakovna. Dostupná data obsahují požadavky na položky, seznam položek a jejich napojení na skidtyp, seznam skidtypů a jejich kapacitu. Jediná informace, kterou nemáme k dispozici je dostupný počet skidů daného skidtypu. Některé skidy jsou v opravě, na čištění, ve skladě nebo v painting trainu a dlouhodobě se nikdy nestalo, že by byl nedostatek skidů a zákazník tedy nemá potřebu počty skidů kontrolovat a udržovat. V modelu tedy nebudeme uvažovat omezení na maximální počet skidů jednotlivých skidtypů. Tato data již splňují všechny požadavky pro plánování a lze je v práci využít.

4.3 Sběr dynamických dat

Dynamická data jsou data, která se mění nebo aktualizují v průběhu času. Jsou to data, která nemají statickou hodnotu, ale jsou poddána změnám, aktualizacím nebo úpravám. Dynamická data, která jsou pro nás důležitá, jsou požadavky na položky.

K dispozici máme požadavky na následujících přibližně 20 dní. Požadavky se však mění ze dne na den. Poměrně přesné požadavky máme pouze na pár dní až týden. Je to tím, že většina zákazníků může měnit požadavky na následující dny, někteří zákazníci dokonce mohou změnit požadavky i na aktuální den, a ať to zní jakkoliv nesmyslně, existují i zákazníci, kteří mají právo změnit požadavky i na předchozí den, kdy změnou je ve všech případech myšleno zrušení, přidání nebo úprava množství a času požadavku.

Abychom se co nejvíce přiblížili realitě, budeme požadavky na položky aktualizovat dvakrát týdně čili přibližně jednou za tři dny. Tímto způsobem byly celkem nashromážděny požadavky na 84 dní od 28. 1. 2023 do 21. 4. 2023.

4.4 Plán řešení

V následujících sekcích naší práce nejprve provedeme kontrolu všech obdržených dat a vhodně je připravíme pro statistické zpracování. Poté použijeme statistické metody ke zpracování těchto dat a určíme počty skidtypů, které jsou stabilně využívány. Vycházíme z analýzy historických dat, díky které předpokládáme, že tyto skidtypy budou potřebné i v námi stanoveném časovém horizontu. Tento krok ve zpracování dat a určení počtu stabilních skidtypů nám umožní zmenšit velikost optimalizačního problému, který budeme řešit v další fázi.

Následně se budeme zabývat hledáním optimálního rozložení skidtypů v rámci painting trainu přes více dní. Toho dosáhneme pomocí celočíselného lineárního programování, které nám umožní nalézt nejlepší možné rozložení skidtypů tak, abychom byli každý den schopni splnit požadavky na lakovnu.

Statistické zpracování dat

5

V této kapitole se zaměříme na statistické zpracování dat. Předtím, než se pustíme do analýzy, je důležité provést sumarizaci, základní vyhodnocení posbíraných nebo obdržených dat a jejich základní kontrolu. Tato fáze nám umožňuje získat přehled o datové sadě a identifikovat její hlavní charakteristiky. Začneme tím, že přehledně shrneme výsledky sběru dat a obdržených dat, zahrnující jejich strukturu, obsah, rozsah a případné chybějící hodnoty. Poté budeme navrhovat model pro vstupní data, který následně využijeme při plánování. Dále se také budeme věnovat verifikaci nalezeného modelu. Verifikace je důležitým krokem, který nám umožňuje ověřit platnost a spolehlivost navrženého modelu na základě dostupných dat. Veškeré statistické zpracování dat bude provedeno v softwaru R (R Core Team, 2021). Sumarizace dat je zpracována v souboru *zpracovani_dat.R* a hledání modelu vstupních dat je zpracováno v souboru *model_vstupnich_dat.R*.

5.1 Umístění a struktura dat

V této podkapitole se zaměříme na popis, kde a jak jsou data uložena v rámci naší práce, abychom měli jasný přehled o jejich umístění a organizaci. Abychom zajistili anonymitu společnosti, se kterou jsme spolupracovali, rozhodli jsme se přejmenovat skidy a položky datové sady.

Veškerá vstupní data byla umístěna do dvou textových CSV souborů. První z nich (*polozky.csv*) obsahuje statická data čili seznam položek spolu s jejich specifikací jako je skidtype a počet. Skidtype je identifikační kód, který se používá k označení specifického věšáku (skidtypu), na který je možné danou položku pověsit. Počet u každé položky nám udává, kolik položek je možné na daný skidtype pověsit, tedy kolik míst je na skidtypu k dispozici. Druhý soubor (*pozadavky.csv*) obsahuje dynamická data v podobě požadavků na položky. Tyto požadavky jsou zaznamenány v řádcích dat, které obsahují parametry název položky, datum a počet požadovaných kusů. Název položky nám udává konkrétní položku, pro kterou je požadavek vytvořen.

Tento název jednoznačně identifikuje danou položku. Datum nám udává časový okamžik, kdy má být položka k dispozici a počet požadovaných kusů nám udává, kolik kusů dané položky je požadováno.

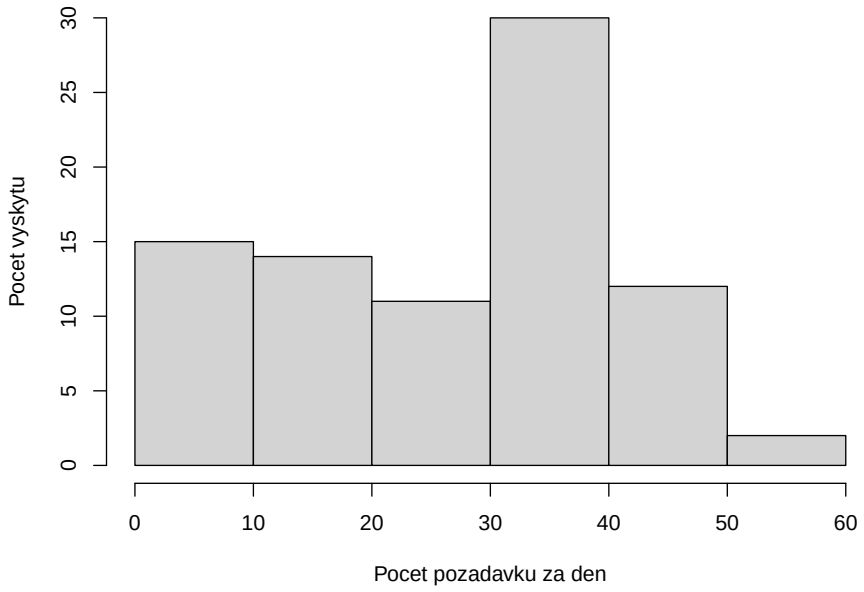
5.2 Sumarizace vstupních dat

V této kapitole se budeme věnovat sumarizaci dat získaných od zákazníka. Předpokládáme, že data jsou kvalitní a použitelná, protože byla průběžně kontrolována a opravena na straně zákazníka. V kapitole 4 jsme již popsali, jak jsme tato data získali a jak jsme zajistili jejich kvalitu. Nyní se budeme soustředit na proces sumarizace dat, který nám umožní získat užitečné informace a přehled o datech. Základní nedostatky dat, jako jsou například outliery, bychom mohli identifikovat a odstranit i v rámci této sumarizace dat.

V první řadě se zaměříme na soubor *polozky.csv*, který obsahuje informace o lakovaných položkách. V tomto souboru jsou k dispozici informace o 788 položkách s jednoznačně přiřazeným skidtypem. Celkem existuje 178 různých skidtypů, přičemž nejvíce položek je přiřazeno ke skidtypu *Skid*056, který zahrnuje 23 druhů položek. Každý skidtype má kapacitu, která se může lišit v rozmezí od 2 do 1176. Nejčastější kapacitou je hodnota 12, kterou má 27 skidtypů, následovaná hodnotou 8, kterou má 26 skidtypů.

V další části se zaměříme na soubor *pozadavky.csv*, ve kterém se nachází požadavky zákazníků na položky. Každý den máme k dispozici jiný počet požadavků. Jak již bylo zmíněno v kapitole 4.3 o sběru dynamických dat, máme požadavky na 12 týdnů od 28. 1. 2023 do 21. 4. 2023. Abychom mohli data lépe analyzovat, začínáme se sobotou a poslední požadavek je z pátka, takže každý den v týdnu má stejné zastoupení v datech. Celkem v souboru nalezneme 2414 požadavků. Některé položky mohou být v jeden den vyžadovány více zákazníky, což znamená, že bude existovat více požadavků na tuto položku v jeden den. Jelikož se jedná o jednu položku a můžeme ji tedy pověsit na stejný skid, je potřeba tyto požadavky sečíst a spojit je do jednoho požadavku. Po této operaci se celkový počet požadavků sníží na 2292. Počet požadavků se může každý den lišit v závislosti na potřebách zákazníků. V období, pro které máme data, se počet požadavků pohyboval od 1 požadavku na den až po 58 požadavků na den. Histogram na obrázku 5.1 zobrazuje počty požadavků na jeden den na ose x a počet dnů, ve kterých se daný počet požadavků vyskytl, na ose y.

Z histogramu je patrné, že největší počet požadavků se pohybuje v rozmezí [30,40), s následným přibližně polovičním počtem požadavků v kategorii [0,10) a [10,20).



Obrázek 5.1: Histogram počtu požadavků na položky za den.

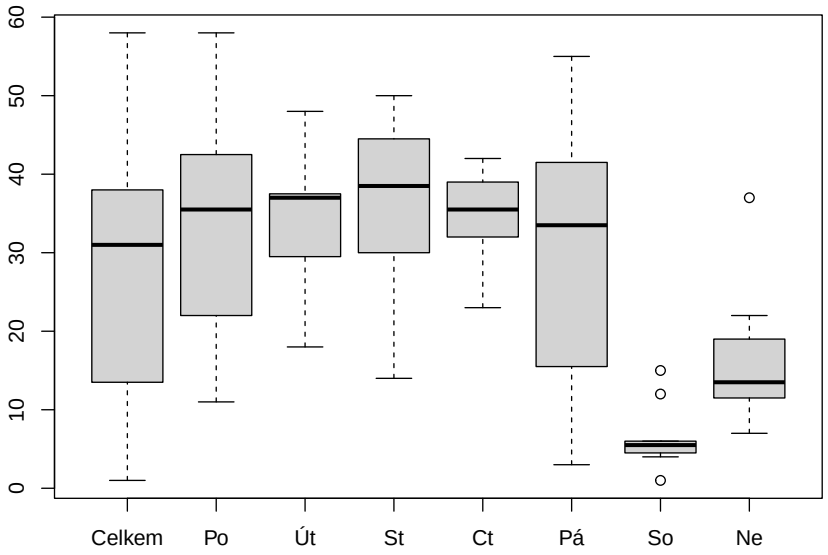
Tato nízká čísla jsou ovlivněna hlavně víkendy, kdy počet požadavků v sobotu a neděli je v porovnání s ostatními dny v týdnu nejnižší. Podrobnější informace o počtu požadavků v jednotlivých dnech v týdnu jsou zobrazeny na obrázku 5.2, který zahrnuje 8 boxplotů. První boxplot zobrazuje počet požadavků v každém dni bez rozlišení v celém zkoumaném období, zatímco zbylých 7 boxplotů ukazuje počet požadavků v každém dni v týdnu. Z grafu je patrné, že počet požadavků v sobotu a neděli je nižší než v ostatních dnech v týdnu.

V práci budeme hlavně pracovat s počty skidů, je tedy potřeba požadavky na položky převést na požadavky na skidtypy. Naštěstí máme k dispozici informace o jednoznačném přiřazení položek k určitému skidtypu a také o kapacitě každého skidtypu. Díky těmto údajům můžeme snadno vypočítat počty požadavků na skidy pomocí vzorce

$$S_i = \lceil x_i / y_i \rceil, \tag{5.1}$$

kde

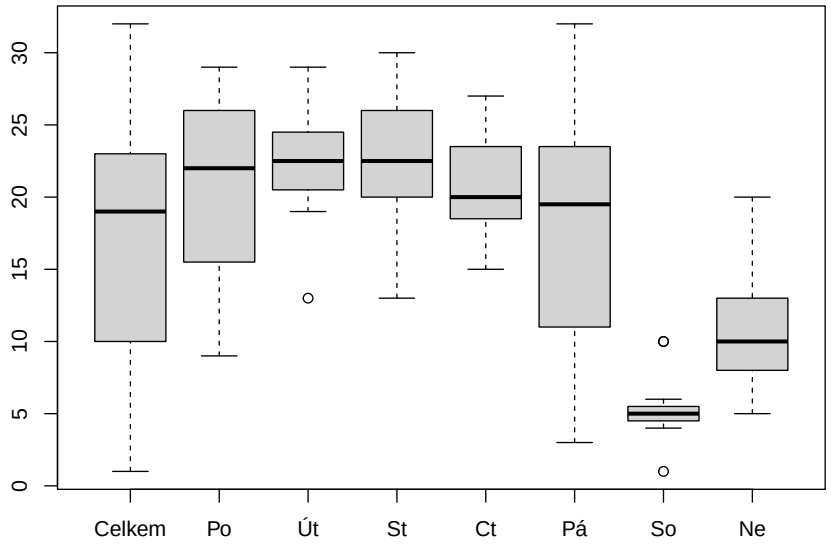
S_i je počet skidů potřebných pro splnění i-tého požadavku,
x_i je požadované množství položek i-tého požadavku,
y_i je počet položek, který je možné pověsit na příslušný skidtype a
$\lceil \rceil$ označuje zaokrouhlení nahoru na celé číslo.



Obrázek 5.2: Boxploty počtu požadavků na položky v jednotlivých dnech v týdnu.

Podobně jako u položek, může nastat situace, kdy máme více požadavků na jeden skidtype v jeden den. Proto, abychom získali celkový počet skidtypů potřebných v daném dni pro plánování painting trainu, sečteme požadavky na stejný skidtype a vytvoříme z nich jeden požadavek. Po této operaci se snížil celkový počet požadavků na skidy na 1434. Na obrázku 5.3 vidíme 8 boxplotů, přičemž první zobrazuje počty požadavků na skidtypy v celém zkoumaném období bez rozlišení a zbylých 7 ukazuje počty požadavků na skidy na jednotlivé dny v týdnu.

Počet požadavků v jednotlivých dnech nám však stále neřekne, jaké nároky jsou kladeny na výrobu. Podívejme se proto nyní na celkové počty požadovaných skidů. Sečtením požadavků na jednotlivé dny jsme obdrželi celkový počet skidů, který musí projet daný den lakovnou. Jak již bylo popsáno v kapitole 3.4, painting train má maximální kapacitu 206 skidů a projede lakovnou za den pětkrát. Maximální počet skidů, který za den projede lakovnou je tedy 1030. Tato hodnota však byla v 11 případech překonána, tedy celkový požadovaný počet skidů byl větší, než jsme schopni za den nalakovat. Největší množství skidů je požadováno 7. 4. 2023 a tento den je potřeba nalakovat 1472 skidů, což značí nárůst o 43 % nad maximální kapacitu lakovny. Tímto problémem se budeme zabývat v kapitole 6 o formulaci optimalizační úlohy.



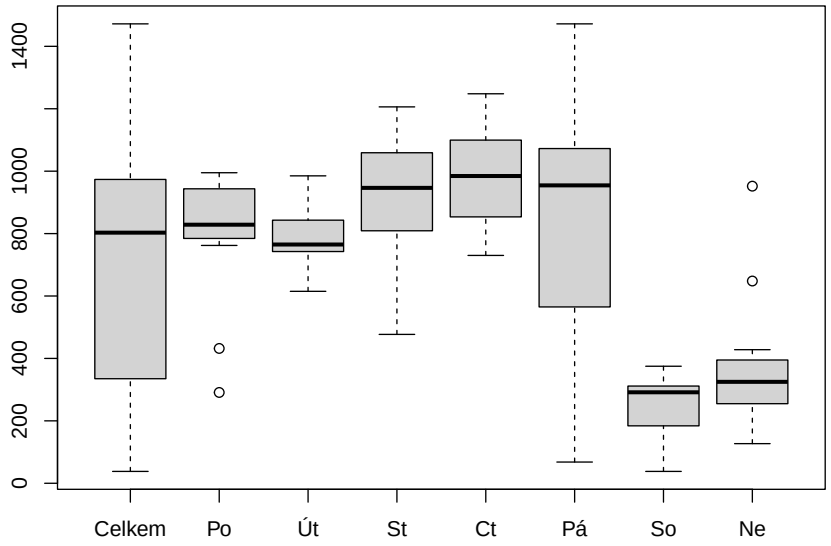
Obrázek 5.3: Boxploty počtu požadavků na skidy v jednotlivých dnech v týdnu.

Na obrázku 5.4 jsou zobrazeny boxploty počtu požadovaných skidů nejprve pro všechny dny bez rozlišení a následně pro jednotlivé dny v týdnu. V obrázku si můžeme povšimnout, že v úterý a v sobotu jsou požadavky poměrně stabilní a naopak nejméně stabilní jsou požadavky v pátek. Na obrázcích nebyly zjištěny žádné zásadní problémy s daty. Bylo možné pozorovat několik outlierů, nicméně jejich výskyt neodporuje očekávání, a tudíž nejsou považovány za závažný problém.

Histogram na obrázku 5.5 ukazuje, jak často se vyskytují různé počty požadovaných skidů za jeden den. Na ose x jsou vypsány počty skidů a na ose y je zobrazen počet dnů, ve kterých byl daný počet skidů požadován. Z grafu je patrné, že nejčastěji se požaduje 700–1000 skidů za den, následovaných počtem 200–400 skidů. Je také vidět, že některé dny mají požadavky přesahující maximální kapacitu lakovny.

5.3 Model vstupních dat

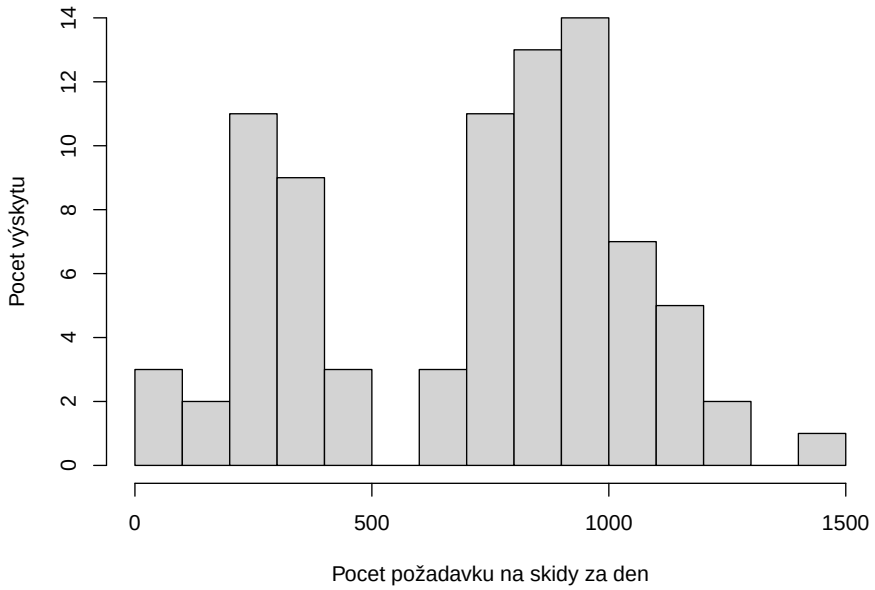
V následující fázi naší práce se budeme věnovat hledání modelu vstupních dat, konkrétně se zaměříme na určení minimálního počtu skidů, které musí být v painting



Obrázek 5.4: Boxploty počtu požadovaných skidů v jednotlivých dnech v týdnu.

trainu vždy k dispozici. Tyto hodnoty stanovíme tím, že identifikujeme skidy, které jsou stabilně využívány v painting trainu, protože na ně dostáváme požadavky ať už pravidelně či nepravidelně, ale ve větším množství. Minimální počet skidtypů by měl odpovídat průměrnému počtu požadovaných skidtypů na lakovací kolo, abychom byli schopni pokrýt běžné potřeby a zároveň minimalizovali nutnost častých výměn skidů. V případě, kdy dostáváme pravidelné požadavky na určitý skidtype, je výhodné nechat tyto skidy v painting trainu, abychom minimalizovali četnost jejich vyjímání a opětovného vkládání. Ve druhém případě, kdy požadavky na daný skidtype přicházejí nepravidelně, ale s velkým požadovaným množstvím, je stále nutné udržovat v painting trainu alespoň průměrný počet skidů, abychom byli schopni požadavky ve stanoveném čase lépe splnit a nemuseli provádět velké množství výměn skidů najednou. Udržováním skidů v painting trainu také umožníme předvýrobu kritických požadavků bez nutnosti okamžité výměny skidů.

Nalezení tohoto rozložení je pro nás velice důležité, jelikož nechceme v painting trainu zbytečně měnit skidy, které jsou často používané a museli bychom je s vysokou pravděpodobností do painting trainu brzo vracet. Důležitým pravidlem či omezením je, že při hledání minimálního počtu skidů známe pouze přibližné hodnoty budoucích požadavků a je tedy nutné využít pouze data z minulosti. V našem případě tedy data rozdělíme na část trénovací, která se bude týkat požadavků od



Obrázek 5.5: Boxploty počtu požadovaných skidů v jednotlivých dnech v týdnu.

28. 1. 2023 až do 25. 3. 2023, a část testovací, která obsahuje data na měsíc od 26. 3. 2023 do 21. 4. 2023. K nalezení modelu vstupních dat využijeme první sadu dat čili testovací data.

Jelikož v painting trainu nepracujeme s jednotlivými položkami, ale s celými skidy, prvním krokem při hledání minimálního počtu skidů je přepočtení požadavků na položky na požadavky na skidy. Jak již bylo řečeno v předchozí kapitole, může nastat situace, kdy v jeden den se sejde více požadavků na jednu položku. Jelikož se ale jedná o stejnou položku a můžeme ji tedy pověsit na jeden skid, je potřeba všechny tyto požadavky spojit a vytvořit z nich jeden požadavek na danou položku, kde výsledné požadované množství bude součtem požadovaných množství jednotlivých požadavků. Poté již můžeme pomocí vzorce 5.1 provést přepočet z požadavků na položky na požadavky na skidy.

Podíl zaokrouhlujeme na celá čísla, jelikož počet skidů je prvkem přirozených čísel, a zaokrouhlujeme ho nahoru, jelikož musí být použit takový počet skidů, aby bylo možné požadavky splnit. Rozdílem mezi desetinným číslem a zaokrouhleným celým číslem se nemusíme zabývat, jelikož skidy není nutné osazovat do jejich maximální kapacity čili mohou jet z části prázdné. V realitě většinou tato místa plánovač doplní

ručně položkami ze skladu černých dílů, do kterého se odvážejí všechny položky, které lakovna není schopna nalakovat z důvodu rychlejší výroby předchozího procesu.

Jelikož některé položky mohou sdílet stejný skidtype, může opět nastat situace, tentokrát ne na úrovni položek, ale na úrovni skidů, kdy na jeden den bude více požadavků na jeden skidtype. Opět tedy tyto požadavky sečteme, přesněji řečeno sečteme požadované množství jednotlivých požadavků a ostatní parametry ponecháme nezměněny, a tak z nich vytvoříme jeden požadavek.

Pro zvýšení přehlednosti a jednodušší statistické zpracování dat vytvoříme matici požadavků **A**, která bude mít řádky odpovídající jednotlivým skidtypům a sloupce obsahující data od 28. 1. 2023 do 25. 3. 2023. Prvky matice **A**, označené jako a_{ij}, obsahují nenulovou hodnotu, pokud existuje požadavek na skid i v čase j, a hodnotu 0, pokud požadavek neexistuje. Hodnota a_{ij} bude odpovídat požadovanému množství skidu i v čase j.

Náš cíl je identifikovat stabilní skidtypy, které pravidelně procházejí lakovnou. Poté budeme stanovovat počet jednotlivých skidtypů, které chceme udržovat v painting trainu. Stabilní skidtypy, které jsou statisticky významné, jsou ty, pro které je střední hodnota (medián) počtu požadovaných skidtypů na lakovací kolo nejméně 2. U těchto skidtypů můžeme s vysokou pravděpodobností předpokládat, že se budou pravidelně vyskytovat v painting trainu a budeme je považovat za významné.

Abychom získali průměrné požadavky na jedno lakovací kolo v daném dni, vydělíme počet požadavků na skidy za den počtem lakovacích kol. Poté najdeme stabilní skidtypy, které ve zjištěném počtu budeme zařazovat do painting trainu pro každé lakovací kolo. Nulová a alternativní hypotéza jsou formulovány následovně:

H_0 : Střední hodnota (medián) požadovaných skidů na lakovací kolo je 1

H_1 : Střední hodnota (medián) požadovaných skidů na kolo je větší než 1

V případě t-testu se nulová hypotéza týká střední hodnoty, zatímco ve Wilcoxonově testu se nulová hypotéza vztahuje k mediánu. K ověření hypotézy H_0 chceme použít jednostranný t-test (studentův test střední hodnoty). Nejprve je však potřeba ověřit předpoklad, že výběr pochází z normálního rozdělení. K ověření tohoto předpokladu využijeme Shapirův-Wilkův test normality. Požadavky u žádného ze skidů nenabývají normálního rozdělení, nelze tedy použít t-test. Jako alternativu tedy použijeme jeden z neparametrických testů, jelikož se u nich nepředpokládá normalita náhodných veličin.

Konkrétně využijeme k otestování hypotézy H_0 Wilcoxonův jednovýběrový test, pomocí kterého jsme nulovou hypotézu zamítli celkem 17krát. Těchto 17 skidtypů je pro nás statisticky významných a chceme je nechávat v painting trainu. Seznam významných skidtypů společně s p-hodnotou Wilcoxonova testu nalezneme v tabulce 5.1

Tabulka 5.1: Statisticky významné skidtypy a p-hodnota Wilcoxonova testu.

Skidtype	p-hodnota	Skidtype	p-hodnota	Skidtype	p-hodnota
Skid001	$< 0,001$	Skid005	$< 0,001$	Skid007	$< 0,001$
Skid011	$< 0,001$	Skid013	$< 0,001$	Skid017	$< 0,001$
Skid031	$< 0,016$	Skid041	$< 0,001$	Skid054	$0,048$
Skid066	$< 0,001$	Skid067	$0,008$	Skid163	$0,014$
Skid169	$0,003$	Skid170	$< 0,001$	Skid171	$< 0,001$
Skid173	$< 0,001$	Skid176	$0,008$		

Nyní musíme stanovit optimální počet významných skidtypů, které chceme udržovat v painting trainu a využít jako minimální počet skidů při optimalizaci. Jako nejvhodnější se jeví použít průměrný požadovaný počet skidů na jedno lakovací kolo, protože tímto způsobem teoreticky pokryjeme všechny požadavky napříč všemi lakovacími koly. Přestože skidy mohou být některé dny prázdné, není to pro nás problém, protože víme, že v průměru za delší období obdržíme požadavek, na který tyto skidy velice pravděpodobně využijeme.

Pro skidtypy, u kterých jsme nulovou hypotézu nezamítli, volíme minimální počet skidů držených v painting trainu roven nule. Pro statisticky významné skidtypy volíme minimální počet skidů roven průměrné hodnotě požadovaného počtu skidů na lakovací kolo zaokrouhlené matematicky na celé číslo. Statisticky významné skidtypy, jejich průměrný požadovaný počet na lakovací kolo a minimální počet skidů držených v painting trainu je zobrazen v tabulce 5.2.

Z tabulky vidíme, že nejvyužívanějším skidtypem je *Skid*017, pro který je minimální hodnota skidů držených v painting trainu rovna 29. Hned za ním je skidtype *Skid*169, pro který je minimální hodnota skidů v painting trainu rovna 21. Celkový počet skidů, který chceme v painting trainu udržovat je 127, což odpovídá přibližně dvěma třetinám velikosti painting trainu. Zbývající třetinu skidů ponecháme k dispozici pro případné nepravidelné a ojedinělé požadavky. Tento přístup nám umožní optimalizovat pouze část skidů v trainu, což nám ušetří čas a zlepší efektivitu provozu.

Tabulka 5.2: Statisticky významné skidtypy, průměrný požadovaný počet a minimální počet v painting trainu.

Skidtype	Průměr	Počet	Skidtype	Průměr	Počet
Skid001	3,89	4	Skid005	8,64	9
Skid007	8,91	9	Skid011	3,53	4
Skid013	3,75	4	Skid017	29,49	29
Skid031	2,65	3	Skid041	8,44	8
Skid054	1,24	1	Skid066	2,81	3
Skid067	2,10	2	Skid163	6,26	6
Skid169	20,78	21	Skid170	10,05	10
Skid171	2,55	3	Skid173	1,60	2
Skid176	9,22	9			

Minimální počet skidů držený v painting trainu exportujeme ze softwaru R do textového souboru *min_skidu1.csv*. Dále také exportujeme matici požadavků na skidy do textového souboru *pozadavky_matice_uceni.csv*. Oba soubory budeme dále využívat pro hledání optimálního rozložení painting trainu.

5.4 Verifikace modelu vstupních dat

Výroba je dynamický proces a požadavky na ni se mohou měnit během času z mnoha důvodů. Může se jednat o nové trendy na trhu nebo potřeby zákazníků. Proto je důležité průběžně aktualizovat modely, které používáme k plánování a optimalizaci výroby.

V našem případě jsme vytvořili model vstupních dat na základě posledních 2 měsíců (přesněji 8 týdnů). Tento model nám udává skidy, které chceme v painting trainu neustále držet a nechceme je tedy vyměňovat za jiné. Avšak chceme si být jisti, že náš model je stále platný a přesný, a proto budeme model pravidelně verifikovat. Každý měsíc budeme model verifikovat na základě dat z posledních 2 měsíců, což nám umožní zahrnout do modelu aktuální vývoj a změny na trhu a budeme tak schopni efektivně plánovat a optimalizovat výrobu a přizpůsobit se měnícím se požadavkům trhu.

Data zpracujeme stejným způsobem jako v předchozí kapitole o hledání modelu vstupních dat, posuneme se však v čase o 4 týdny dopředu čili budeme mít požadavky od 25. 2. 2023 do 21. 4. 2023. Bereme tedy měsíc požadavků z trénovacích dat a přidáme k nim data testovací. U seznamu položek a jejich specifikace nedošlo v průběhu tohoto měsíce k žádné změně a budeme tedy používat stejná data ze souboru *polozky.xlsx*.

Při testování normality dat jsme opět nenašli jediný skidtype, u kterého by požadavky nabývaly normálního rozdělení a k testování hypotézy o hodnotě mediánu jsme pro všechny skidtypy využívali Wilcoxonův jednovýběrový test. Stejně jako při hledání původního modelu vstupních dat jsme nalezli 17 statisticky významných skidtypů a nenastala zde žádná změna. Zřejmě však došlo ke snížení počtu požadovaných položek u těchto skidtypů, a tedy i minimálního počtu skidů jednotlivých skidtypů. Tabulku významných skidtypů, průměrné požadované množství na lakovací kolo a minimální počet skidů udržovaných v painting trainu můžeme vidět v tabulce 5.3.

Tabulka 5.3: Statisticky významné skidtypy, průměrný požadovaný počet na lakovací kolo a minimální počet v painting trainu.

Skidtype	Průměr	Počet	Skidtype	Průměr	Počet
Skid001	3,81	4	Skid005	7,78	8
Skid007	8,74	9	Skid011	3,29	3
Skid013	3,75	4	Skid017	27,83	28
Skid031	2,44	2	Skid041	7,93	8
Skid054	1,31	1	Skid066	2,90	3
Skid067	1,99	2	Skid163	6,00	6
Skid169	18,76	19	Skid170	9,93	10
Skid171	2,23	2	Skid173	1,65	2
Skid176	8,69	9			

Celkový počet skidů, který chceme držet v painting trainu klesl ze 127 na 120. Z tabulky 5.3 vidíme, že u skidů *Skid005, Skid011, Skid017, Skid031* a *Skid171* klesl minimální počet o jeden skid a u skidu *Skid169* klesl minimální počet z 21 na 19 čili o dva skidy.

Minimální počet skidů jednotlivých skidtypů z tabulky 5.3 budeme využívat při hledání optimálního rozložení painting trainu pro následující měsíc čili od 22. 4. 2023 až do 19. 5. 2023. Tento postup bude aplikován každý měsíc a dle aktuálních výsledků bude upraven minimální počet skidů jednotlivých skidtypů, který bude vždy v painting trainu umístěn.

Formulace optimalizační úlohy

6

V této kapitole se budeme věnovat definici důležitých parametrů, proměnných a omezení pro optimalizační úlohu, kterou se snažíme vyřešit. Definice těchto parametrů, proměnných a omezení je klíčová pro správné formulování úlohy a nalezení optimálního řešení. V první části kapitoly se zaměříme na tvorbu modelu lineárního programování. V další části se budeme zabývat definicí parametrů a definicí omezení, jako je maximální počet výměn skidů v jednom lakovacím kole nebo kapacita lakovny. Poté definujeme proměnné, které vstupují do cílové funkce a nakonec způsob, jakým budeme úlohu řešit.

6.1 Tvorba modelu lineárního programování

V této kapitole se budeme zabývat procesem tvorby modelu lineárního programování (LP), zejména jeho iterativním zpracováním a testováním. Iterace jsou zásadní pro správné vytvoření modelu, protože umožňují postupné vylepšování a doladění modelu a jeho komponentů. Všechny modely LP byly tvořeny a řešeny v optimalizačním softwaru AMPL (AMPL Optimization LLC, 2019). V dalších kapitolách se zaměříme na soubor DP.mod, který jsme vytvořili v optimalizačním softwaru AMPL a obsahuje parametry a omezení našeho modelu. Tyto prvky budou podrobně popsány a vysvětleny v následujících částech.

6.1.1 Základní model

Požadovaným výstupem ze strany zákazníka jsou počty skidtypů, které pojedou v jednotlivých lakovacích kolech. V prvním modelu jsme se tedy zaměřili právě na definici proměnných odpovídajících požadovanému výstupu. Definovali jsme matici **final_rozlozeni** o velikosti $m \times n$, kde m je počet skidtypů, n je počet lakovacích kol a $final\_rozlozeni[i, j]$ je počet skidů skidtypu i, který je nasazen v lakovacím kole j. Jelikož se jedná o počet skidů, je nutnou podmínkou, aby tyto hodnoty byly celočíselné. Aby jednotlivé řádky a sloupce matice **final_rozlozeni** odpovídaly kon-

krétním skidtypům a lakovacím kolům, bylo nutné tyto dva seznamy definovat. Seznam skidtypů se může v průběhu času měnit a bude tedy pravidelně importován jako vstupní parametr. Pro zavedení označení lakovacích kol je potřeba znát počet lakovacích kol, který stihneme nalakovat během jednoho dne. Počet lakovacích kol také definujeme jako vstupní parametr, aby bylo možné ho v případě potřeby změnit. Lakovací kola následně označíme čísly od 1 do počtu lakovacích kol.

Dále jsme definovali omezení na počet skidů jednotlivých skidtypů, který bude během dne umístěn v painting trainu. Našim cílem je, aby lakovnou projelo více skidů, než je požadavků pro daný den. Aby mohlo být toto omezení splněno, je nejprve potřeba obdržet požadavky na skidy na daný den. Požadavky na skidy tedy definujeme jako další ze vstupních parametrů, který je potřeba před výpočtem importovat.

Tímto jsme vytvořili základní model, jehož výstupem je počet skidtypů v jednotlivých lakovacích kolech a celkový počet nasazených skidů je dostatečný pro splnění všech požadavků na skidy. Proměnné v matici **final_rozlozeni** však zatím nemají žádná pravidla a omezení, aby hodnoty dávaly v realitě smysl.

6.1.2 Velikost painting trainu

Ve výstupu základního modelu bylo jednoduché splnit požadavky na skidy, jelikož velikost painting trainu nebyla žádnými hodnotami omezena. Přidáme tedy omezení na velikost painting trainu. Painting train má dynamickou velikost, která je však omezena na určitý interval. Nejprve je tedy nutné definovat krajní hodnoty tohoto intervalu. Jako další vstupní parametry volíme maximální velikost painting trainu a minimální velikost painting trainu. Počet všech skidů umístěný v painting trainu v jednotlivých lakovacích kolech musí být vždy alespoň minimální a nejvýše maximální velikost painting trainu.

6.1.3 První lakovací kolo

Další omezení, které je potřeba přidat se týká prvního lakovacího kola. Jak již bylo zmíněno v kapitole 3 o charakteristice plánování, rozložení painting trainu v prvním lakovacím kole je dáno rozložením z posledního lakovacího kola předchozího dne.

Je tedy potřeba jako vstupní parametr definovat vektor počátečního rozložení painting trainu, který bude obsahovat počet jednotlivých skidtypů umístěných v painting trainu v posledním lakovacím kole předchozího dne. Následně první sloupec matice **final_rozlozeni** položíme roven vektoru počátečního rozložení painting trainu.

6.1.4 Výměna skidů

V dalším kroku definujeme návaznost mezi jednotlivými lakovacími koly. Pokud je počet skidů daného skidtypu v lakovacím kole větší než v předchozím lakovacím kole, musí to nutně znamenat, že byly skidy daného skidtypu do painting trainu přidány. Pokud je naopak počet skidů v daném lakovacím kole menší než v předchozím kole, znamená to, že byly skidy tohoto skidtypu z painting trainu vyjmuty.

Definujme dvě matice proměnných, obě o velikosti $m \times (n-1)$, kde stejně jako v matici **final_rozlozeni** parametr m značí počet skidtypů a parametr n značí počet lakovacích kol. Počet sloupců matice je o jeden menší než počet kol, jelikož se před prvním kolem žádné skidy nemění. Pro definování řádků matic máme již připraven seznam skidtypů, sloupce matic označíme čísly od 2 do n. První matice **skidy_in** bude obsahovat proměnné $skidy\_in[i,j]$ označující počet skidů skidtypu i, který byl přidán do painting trainu mezi lakovacími koly $j-1$ a j. Druhá matice **skidy_out** bude obsahovat proměnné $skidy\_out[i,j]$ označující počet skidů skidtypu i, který byl z painting trainu odebrán mezi lakovacími koly $j-1$ a j.

S využitím matic **skidy_in** a **skidy_out** můžeme nyní definovat prvky matice **final_rozlozeni**, jelikož počet skidů v lakovacím kole přímo závisí na počtu skidů v předchozím lakovacím kole a následně buď na počtu skidů, které byly vyjmuty z painting trainu, nebo na počtu skidů, které byly do painting trainu vloženy. Prvky matice **final_rozlozeni** tedy definujeme pro $i = 1, 2, ..., m$ a $j = 2, 3, ..., n$ jako

$$final\_rozlozeni[i,j] = final\_rozlozeni[i,j-1] + skidy\_in[i,j] - skidy\_out[i,j].$$

Zavedením tohoto omezení jsme se značně přiblížili reálnému provozu lakovny, kdy finální rozložení skidů v lakovacím kole nyní závisí na počtu výměn mezi lakovacími koly. V další kapitole využijeme nově zavedené proměnné k definování cílové funkce.

6.1.5 Počet výměn skidů

V další iteraci se budeme věnovat problematice počtu výměn skidů a jeho zanesení do cílové funkce při optimalizaci plánování. Jak již bylo zmíněno v kapitole 3 o charakteristice plánování, výměna skidů je náročný proces a cílem optimalizace je minimalizovat jejich počet a tím i snížit celkovou časovou náročnost procesu.

Definujeme novou proměnnou vyjadřující celkový počet výměn v průběhu celého dne. Tuto proměnnou získáme sumou všech prvků matice **skidy_in** a **skidy_out**. Tím získáme celkový počet výměn skidů ve všech lakovacích kolech daného dne, kde výměnou skidu rozumíme vyjmutí skidu z painting trainu či zařazení skidu

do painting trainu. Tato proměnná je následně přidána do cílové funkce, kterou se snažíme minimalizovat.

Při podrobné analýze bylo zjištěno několik nedostatků souvisejících s minimalizací počtu výměn skidů pomocí proměnné vyjadřující celkový počet výměn ve všech lakovacích kolech. Pro jednoduchost a v reálném provozu uspokojivé výsledky nebudeme nyní cílovou funkci více komplikovat. Konkrétní nedostatky budou detailně popsány v diskuzi v kapitole 8.

6.1.6 Minimální počet skidů

V kapitole 5.3, která se věnuje modelování vstupních dat, jsme se zaměřili na hledání skidtypů, které jsou pro nás klíčové a chtěli bychom je držet v painting trainu, jelikož na ně pravidelně dostáváme požadavky. Výsledkem této analýzy je vektor minimálního počtu skidů jednotlivých skidtypů, který musí být vždy v painting trainu k dispozici.

Tento vektor minimálního počtu skidů bude dále sloužit jako vstupní parametr pro náš model. Budeme také aplikovat omezení na počet skidů jednotlivých skidtypů v painting trainu tak, aby v každém lakovacím kole byl alespoň minimální počet skidů příslušného skidtypu. Tímto způsobem zajistíme, že přibližně dvě třetiny painting trainu budou fixní a bude možné plánovat s maximální efektivitou a minimalizovat počet výměn skidů.

6.1.7 Požadavky na následující den

Dosud jsme přijímali požadavky pouze na jeden den. Pokud by však počet požadavků překročil naše současné kapacity, bylo by nemožné všechny požadavky splnit v daném termínu. Proto jsme se rozhodli rozšířit náš systém a umožnit přijímání požadavků na další den. Tím budeme schopni lépe plánovat a zajistit včasnou realizaci všech požadavků. V rámci této změny jsme také zavedli předvýrobu na jeden den dopředu, což nám umožňuje být flexibilnější a lépe reagovat na případné změny v požadavcích našich zákazníků.

Požadavky na následující den se stanou dalším vstupním parametrem modelu. Je důležité je zahrnout do výpočtu, ale nelze je zahrnout do omezení stejně jako požadavky na aktuální den, protože by to zvýšilo nároky na výrobu a pravděpodobnost nesplnění požadavků by byla vysoká. Pokusíme se je tedy zahrnout do cílové funkce.

Přidáním požadavků na následující den jsme zavedli jednodenní předvýrobu, což nám umožňuje částečně splnit nejkritičtější požadavky, které by jinak nebylo možné

splnit během jednoho dne. Navíc můžeme lépe přizpůsobit výměny skidů potřebám následujícího dne a minimalizovat počet výměn.

Požadavky na následující den budou dále sloužit jako vstupní parametr pro náš model. V modelu definujeme vektor nesplněných požadavků na následující den, kde každý prvek vektoru představuje daný skidtype. Vektor definujeme jako součet požadavků na skidtypy na aktuální den a požadavků na skidtypy na následující den a odečteme od nich počet nasazených skidů daného skidtypu ve všech lakovacích kolech. Vektor omezíme tak, aby nenabýval záporných hodnot. Výsledkem je počet skidů daného skidtypu, který se nepovedlo splnit z požadavků na následující den. Tyto zbývající požadavky bude potřeba splnit následující den.

Abychom co nejlépe splnili požadavky pro následující den, přidáme do cílové funkce součet prvků vektoru nesplněných požadavků pro následující den. Jelikož minimalizujeme cílovou funkci, bubou minimalizovány i jednotlivé prvky vektoru. Nelze však ovlivnit, který prvek vektoru bude minimalizován. Pokud se například objeví velké množství požadavků na jeden skidtype a my nebudeme schopni všechny požadavky následující den splnit, chtěli bychom, aby tento skidtype měl prioritu pro nasazení do painting trainu. Pro zajištění, že se největší prvky vektoru nesplněných požadavků na následující den budou snižovat nejdříve, přidáme do cílové funkce další parametr, kterým je maximální hodnota z počtu nesplněných skidtypů na následující den.

Nebudeme brát pouze maximální hodnotu z prvků vektoru nesplněných požadavků na následující den, jelikož se může stát, že nejvyšší požadavky budou na skidtype, který je ovšem v painting trainu hojně zastoupen a další den nemusíme provést žádnou výměnu pro to, abychom požadavky na tento skidtype splnili. Od nesplněných požadavků tedy nejprve odečteme počet skidů, který by byl nalakován následující den nepředpokládáme-li žádné výměny skidů. Vypočtené hodnoty nám udávají počty skidů, kvůli kterým bude potřeba následující den v painting trainu provádět změny a až z těchto hodnot hledáme maximum, které bude parametrem cílové funkce.

Protože splnění požadavků má vyšší prioritu než počet výměn skidů, přidáme k novým parametrům v cílové funkci koeficienty, kterými zvýšíme váhu těchto parametrů. Volba těchto koeficientů bude podrobně popsána v kapitole 6.4.

6.1.8 Maximální počet výměn skidů

V předchozí iteraci jsme do cílové funkce přidali parametry pro plnění požadavků na následující den. Jelikož plnění požadavků přikládáme větší prioritu než výměně skidů, může nyní nastat několik nežádoucích situací. Zaprvé se může stát, že se budeme snažit splnit většinu požadavků na úkor počtu výměn, i když bychom tyto požadavky mohli bez problému splnit až následující den. Zadruhé, nemůžeme ovlivnit to, aby se všechny výměny skidů neprovedly v jednom lakovacím kole a následující kola by pak proběhla beze změny.

Abychom vyřešili tyto dva problémy, stanovíme maximální počet výměn skidů v jednom lakovacím kole. Tímto omezením zajistíme, že plnění požadavků na další den bude záviset na volné kapacitě vzhledem k maximálnímu počtu výměn. Navíc se již nebudou vyskytovat situace, kdy je provedeno mnoho výměn v jednom lakovacím kole, ačkoliv by bylo možné některé z nich provést v kole jiném.

V kapitole 3 o charakteristice plánování jsme uvedli, že maximální počet vložení skidů do painting trainu v jednom lakovacím kole je 10 a maximální počet vyjmutí skidů z painting trainu v jednom lakovacím kole je 12. Tyto dvě hodnoty použijeme k omezení počtu výměn, kdy suma hodnot každého sloupce matice **skids_in** bude shora omezena hodnotou maximálního počtu vložení a suma každého sloupce **skids_out** bude shora omezena maximálním počtem vyjmutí.

S omezeným počtem výměn se může stát, že nebudeme schopni splnit požadavky na aktuální den. Nicméně, jak jsme popsali v kapitole 3.5 o strategii plánování, v případě, že stanovený počet výměn není dostačující pro splnění požadavků, je možné toto omezení překročit a provést více výměn. Při hledání řešení budeme postupovat iterativně a pokud v dané iteraci nenajdeme řešení, zvýšíme maximální počet vyjmutí a vložení skidů o jeden a opakujeme výpočet, dokud nenajdeme řešení.

Je možné, že se ocitneme v situaci, kdy požadavky na výměny skidů budou tak vysoké, že i po mnoha iteracích nebudeme schopni nalézt řešení. To však nemusí být nutně způsobeno maximálním počtem výměn, ale jiným omezením, jako je například kapacita lakovny. Toto téma si detailněji probereme v diskuzi v kapitole 8.

6.1.9 Hledání řešení přes více dnů

Až doposud jsme uvažovali pouze řešení na jeden den. Samozřejmě je možné najít výsledek na jeden den, změnit podle výsledku vstupní parametry modelu a následně hledat řešení pro další den. Pro zjednodušení a lepší efektivitu budeme iterativně hledat řešení na více dnů.

Požadavky na skidy budeme importovat ve formě matice požadavků, kde sloupce matice budou tvořeny požadavky na jednotlivé dny. Dalšími vstupními parametry bude den, od kterého chceme výpočet začít a den, kterým chceme řešení ukončit. Přesněji budou hodnoty vyjadřovat sloupce matice požadavků. Tyto parametry nelze volit tak, aby byl začátek větší než konec, začátek větší než jedna a konec větší než počet sloupců bez jedné, jelikož potřebujeme požadavky i na následující den.

V každé iteraci nalezneme rozložení painting trainu pro jednotlivá lakovací kola. Na konci jedné iterace využijeme hodnoty nesplněných požadavků na následující den a vytvoříme z nich požadavky pro následující iteraci a jako požadavky na následující den vezmeme další sloupec z matice požadavků. Maximální počet výměn vrátíme na původní hodnotu a rozložení painting trainu v prvním lakovacím kole nastavíme na rozložení painting trainu v posledním lakovacím kole předchozí iterace.

Řešením tak bude rozložení painting trainu v jednotlivých lakovacích kolech na zadaný počet dnů. Je důležité zmínit, že při změně začátku z prvního dne na jiný je potřeba ke zvolenému dni nastavit i příslušný vektor počátečního rozložení painting trainu a dále je nutné uvažovat, že požadavky na daný den mohly být částečně splněny předchozí den.

6.2 Definice parametrů

V této kapitole se budeme věnovat definici parametrů mezi které patří například požadavky, počet lakovacích kol, kapacita lakovny a další. Bude také důležité specifikovat možné hodnoty těchto parametrů a popsat, jak se tyto hodnoty budou ovlivňovat navzájem. V další části budou vypsány parametry modelu, jejich popis a vymezení.

6.2.1 Vstupní parametry

- pocet_kol $\in \mathbb{N}$: počet lakovacích kol
- min_velikost $\in \mathbb{N}$: minimální velikost painting trainu
- max_velikost $\geq$ min_velikost, max_velikost $\in \mathbb{N}$: maximální velikost painting trainu
- max_vlozeni $\in \mathbb{N}$: maximální počet vložení skidů v jednom lakovacím kole
- max_vyjmuti $\in \mathbb{N}$: maximální počet vyjmutí skidů v jednom lakovacím kole

- pocet_dni $\in \mathbb{N}$: na kolik dní máme požadavky
- den $\in \mathbb{N}$: jaký den začínáme s řešením
- den_konec $\in \mathbb{N}$: jaký den končíme s řešením
- skidy: seznam dostupných skidtypů
- **matice_pozadavku** $\geq \mathbf{0}$: matice velikosti $m \times n$, kde řádky značí skidtypy, sloupce dny a prvky matice požadavky na daný skidtype a daný den
- **min_skidy** $\in \mathbb{N}^m$: vektor délky m (počet skidtypů) s minimálním počtem skidů jednotlivých skidtypů umístěných v painting trainu
- **pocatecni_rozlozeni_vlaku** $\in \mathbb{N}^m$: vektor délky m (počet skidtypů) s počátečním rozložením painting trainu
- a $\in \mathbb{N}$: konstanta cílové funkce
- b $\in \mathbb{N}$: konstanta cílové funkce

6.2.2 Odvozené parametry

- lakovaci_kolo: seznam lakovacích kol označen čísly od 1 do parametru počet_kol
- vymena: seznam lakovacích kol, před kterými probíhají změny v painting trainu, označeny čísly od 2 do parametru pocet_kol
- **pozadavky** $\geq \mathbf{0}$: vektor požadavků na skidtypy na aktuální den získaný z nedokončených požadavků na následující den z předchozího dne
- **poadavky_dalsi_den** $\geq \mathbf{0}$: vektor požadavků na skidtypy na následující den, jedná se o sloupec $den + 1$ matice **matice_pozadavku**
- max_vlozeni2 = max_vlozeni: archivace původního maximálního počtu vložení v lakovacím kole
- max_vyjmuti = max_vyjmuti: archivace původního maximálního počtu vyjmutí v lakovacím kole

6.2.3 Proměnné

- **final_rozlozeni** $\in \mathbb{N}^{m \times n}$: matice finálního rozložení skidtypů v painting trainu v jednotlivých lakovacích kolech, kde řádky značí jednotlivé skidtypy a sloupce značí lakovací kola
- **skidy_in** $\in \mathbb{N}^{m \times (n-1)}$: matice počtu vložených skidů, kde řádky značí skidtypy, sloupce lakovací kola, před kterými je provedeno přidání skidů, a prvky matice počet skidů daného skidtypu, který byl přidán do painting trainu
- **skidy_out** $\in \mathbb{N}^{m \times (n-1)}$: matice počtu vyjmutých skidů, kde řádky značí skidtypy, sloupce lakovací kola, před kterými je provedeno vyjmutí skidů, a prvky matice počet vyjmutých skidů daného skidtypu
- **sum_vymeny_in** $\leq max\_vlozeni$, **sum_vymeny_in** $\in \mathbb{N}^n$: vektor počtu vložení skidů do painting trainu v jednotlivých lakovacích kolech
- **sum_vymeny_out** $\leq max\_vyjmuti$, **sum_vymeny_out** $\in \mathbb{N}^n$: vektor počtu vyjmutí skidů z painting trainu v jednotlivých lakovacích kolech
- $celkem\_vymen \in \mathbb{N}$: celkový počet výměn (vyjmutí a vložení) skidů provedených během dne
- **vyrobene_polozky** $\geq \mathbf{0}^m$: vektor celkového počtu jednotlivých skidtypů, který projede během dne lakovnou
- **nesplnene_pozadavky_dalsi_den** $\in \mathbb{N}^m$: vektor s počty skidů jednotlivých skidtypů, který se nepodařilo splnit z požadavků na následující den
- $max\_nesplnene\_pozadavky\_dalsi\_den$: maximální hodnota prvků vektoru **nesplnene_pozadavky_dalsi_den** bez počtu nalakovaných skidtypů následujícího dne za předpokladu neprovedení žádné změny v painting trainu
- sum_dalsi_den: suma prvků vektoru **nesplnene_pozadavky_dalsi_den**

6.3 Omezení modelu

- Počet nalakovaných skidů: Definujme si celkový počet skidů, který projede během dne lakovnou. Jedná se o součet počtu nasazených skidů přes všechna lakovací kola.

$$nalakovane\_skidy[i] = \sum_j finalni\_rozlozeni[i, j]$$

- Splnění požadavků: Máme zadané požadavky od zákazníků a je potřeba dodržet, aby počet nalakovaných skidů během dne splnil zadané požadavky.

$$pozadavky[i] \leq nalakovane\_skidy[i]$$

- Velikost painting trainu: Pro zajištění správné velikosti painting trainu, která se může dynamicky měnit, jsme definovali minimální a maximální velikost. Součet všech skidů v lakovacím kole omezíme zdola minimální a shora maximální velikostí painting trainu.

$$\forall j \in lakovaci\_kolo : \sum_i finalni\_rozlozeni[i,j] \geq min\_velikost$$

$$\forall j \in lakovaci\_kolo : \sum_i finalni\_rozlozeni[i,j] \leq max\_velikost$$

- Počáteční rozložení painting trainu: Rozložení prvního lakovacího kola máme dané z předchozího dne. V omezení tedy nastavíme rovnost mezi rozložením prvního lakovacího kola a počátečním rozložením painting trainu.

$$\forall i \in skidy : final\_rozlozeni[i,1] = pocatecni\_rozlozeni\_vlaku[i]$$

- Definice finálního rozložení: Další omezení se týká finálního rozložení painting trainu. Jednotlivá lakovací kola jsou na sobě závislá a painting train se mezi lakovacími koly mění vyjmutím nebo vložením skidů. Finální rozložení painting trainu budeme definovat rozložením předchozího lakovacího kola a přidáním vložených skidů a odebráním vyjmutých skidů.

$$\forall i \in skidy, \forall j \in vymena : final\_rozlozeni[i,j] = \\ final\_rozlozeni[i,j-1] + skidy\_in[i,j] - skidy\_out[i,j]$$

- Počet výměn v lakovacím kole: V omezení počtu výměn skidů v lakovacím kole vycházíme z předchozí definice finálního rozložení skidů pomocí matic výměn. Limitujeme maximální počet výměn (vložení a vyjmutí skidů) a reprezentujeme je vektory **sum_vymeny_in** a **sum_vymeny_out**, udávající počet vložení a vyjmutí v jednotlivých lakovacích kolech. Tyto vektory jsou omezeny shora maximálním počtem vyjmutí nebo vložení.

$$\forall i \in vymena : sum\_vymeny\_out[i] = \sum_j skidy\_out[j,i]$$

$$\forall i \in vymena : sum\_vymeny\_in[i] = \sum_j skidy\_in[j,i]$$

- Celkový počet výměn: Dalším omezením, které nám pomůže získat celkový počet výměn, je spočítání součtu všech prvků vektorů **sum_vymeny_in** a **sum_vymeny_out**, které jsme definovali v předchozím omezení. Tento součet poté použijeme jako jeden z parametrů cílové funkce.

$$celkem\_vymen = \sum_i (sum\_vymeny\_in[i] + sum\_vymeny\_out[i])$$

- Minimální počet skidů: Máme daný vektor minimálního počtu skidů v painting trainu. Tímto vektorem zdola omezíme počet skidů daného skidtypu nacházející se v painting trainu.

$$\forall i \in skidy, \forall j \in lakovaci\_kolo : finalni\_rozlozeni[i,j] \geq min\_skidy[i]$$

- Splnění požadavků na další den: Dalším omezením definujeme vektor s počtem nedokončených skidů na následující den tak, že od součtu požadavků na aktuální a následující den odečteme nalakované skidy.

$$\forall i \in skidy : nesplnene\_pozadavky\_dalsi\_den[i] \geq \\ pozadavky[i] + pozadavky\_dalsi\_den[i] - nalakovane\_skidy[i]$$

- Celkový počet nenalakovaných skidů: Při plánování se snažíme splnit jak požadavky na aktuální den, tak požadavky na následující den. Požadavky na následující den však nejsou omezením, ale chtěli bychom je zahrnout do cílové funkce. První proměnnou vstupujícím do cílové funkce je celkový počet nenalakovaných skidů na následující den. Tuto proměnnou definujeme jako součet prvků vektoru nesplněných požadavků na následující den.

$$sum\_dalsi\_den = \sum_i nesplnene\_pozadavky\_dalsi\_den[i]$$

- Maximální hodnota nesplněných požadavků: Dalším omezením definujeme maximální hodnotu nesplněných požadavků na následující den, kterou také přidáváme do cílové funkce. Jedná se o maximální hodnotu vektoru nesplněných požadavků poníženého o výrobu následujícího dne, za předpokladu neprovedení žádné změny v painting trainu po posledním lakovacím kole.

$$\forall i \in skidy : max\_nesplnene\_pozadavky\_dalsi\_den \geq \\ nesplnene\_pozadavky\_dalsi\_den[i] \\ -finalni\_rozlozeni[i, pocet\_kol] \cdot pocet\_kol$$

  Proměnná *max_nesplnene_pozadavky_dalsi_den* omezuje hodnoty nesplněných požadavků shora a jelikož je zařazena do cílové funkce, kterou se snažíme minimalizovat, bude se jednat o maximální hodnotu pravé strany nerovnice.

6.4 Cílová funkce

V předchozí kapitole jsme definovali proměnné cílové funkce, které budeme minimalizovat, a v této kapitole se budeme věnovat konstantám cílové funkce a jejich volbě. Každá konstanta má vliv na výsledné řešení a je důležité ji správně zvolit, aby bylo dosaženo požadovaných výsledků. Budeme diskutovat o tom, jak byly konstanty určeny a jaké faktory byly zohledněny.

Cílová funkce obsahuje tři proměnné: *celkem_vymen*, která označuje celkový počet výměn (vyjmutí a vložení) skidů v painting trainu během celého dne, další proměnnou je *max_nesplnene_pozadavky_dalsi_den*, která označuje maximální hodnotu nesplněného požadavku na skidy na následující den a je násobena koeficientem *a*, a poslední proměnnou *sum_dalsi_den*, která označuje celkový počet nesplněných požadavků na následující den a je násobena koeficientem *b*.

Protože počet výměn má nejnižší prioritu ze všech proměnných cílové funkce, má u něj koeficient hodnotu jedna. Splnění požadavků má pro nás vyšší prioritu než počet výměn, proto zvolíme koeficienty *a* a *b* větší nebo rovny jedné. Abychom zvolili správné koeficienty, provedeme citlivostní analýzu. To znamená, že budeme hledat řešení úlohy na trénovacích datech a podle výsledků se rozhodneme, která kombinace parametrů je nejlepší.

Pro účely citlivostní analýzy byl vytvořen soubor *DP_cit_analyza.run*, kterým hledáme řešení modelu pro různé kombinace koeficientů *a*, *b*. Vstupní data modelu jsou uložena v souboru *DP_uceni.dat* a zahrnují trénovací vstupní data. Pro každou kombinaci si ukládáme maximální hodnotu, kolikrát došlo k navýšení maximálního počtu výměn, než bylo nalezeno řešení. Výpočet vypneme, navýšíme-li maximální počet výměn alespoň 10krát. Pro výpočet těchto hodnot byly voleny kombinace parametrů *a*, *b* z intervalu [1, 10] s krokem 1. Výsledné počty navýšení můžeme vidět v tabulce 6.1.

Z tabulky 6.1 vidíme, že u 24 z celkových 100 kombinací koeficientů *a*, *b* nedošlo ke zvýšení maximálního počtu výměn a některé z nich se v tabulce nachází u sebe a tvoří stabilnější oblasti. Samotné kritérium však nestačí k rozhodnutí o nejlepší kombinaci. Přidáváme tedy další kritérium, kterým bude rychlost řešení. Cílem je minimalizovat čas potřebný k nalezení řešení. V tabulce 6.2 jsou uvedeny časy řešení pro všechny kombinace parametrů *a*, *b*, pro které bylo nalezeno řešení bez nutnosti zvýšení maximálního počtu výměn. Podle tabulky 6.2 můžeme konstatovat, že všechny časy řešení jsou delší než 20 sekund, s výjimkou kombinace koeficientů [9,2]. Tato kombinace je pro nás nejlepší a budeme ji nadále v rámci práce používat.

Tabulka 6.1: Maximální počet navýšení maximálního počtu výměn.

a\b	1	2	3	4	5	6	7	8	9	10
1	10	10	10	10	10	10	10	10	10	10
2	3	10	10	1	10	10	10	10	0	10
3	10	10	1	10	2	1	10	10	10	4
4	2	10	6	0	0	10	1	1	0	0
5	0	4	3	8	10	1	0	1	10	5
6	10	10	10	10	0	10	7	10	4	1
7	10	10	10	0	10	7	0	0	0	10
8	10	0	0	0	10	0	10	0	0	10
9	3	0	10	5	9	10	10	10	6	0
10	0	1	10	10	0	10	0	10	10	0

Tabulka 6.2: Čas výpočtu pro různé kombinace koeficientů.

a	b	Čas [s]	a	b	Čas [s]
2	9	22,6	8	2	26,3
4	4	40,8	8	3	28,6
4	5	23,7	8	4	23,2
4	9	23,3	8	6	26,0
4	10	21,2	8	8	21,2
5	1	32,5	8	9	23,3
5	7	38,8	9	2	17,4
6	5	21,8	9	10	37,7
7	4	33,2	10	1	31,2
7	7	20,1	10	5	25,5
7	8	28,2	10	7	29,0
7	9	29,2	10	10	24,7

Parametry, omezení a koeficienty cílové funkce byly zvoleny na základě analýzy trénovacích dat. Toto nastavení modelu bude dále použito pro řešení úlohy se sadou testovacích dat, která obsahuje informace na následující měsíc.

Simulace plánování 7

V další kapitole se budeme věnovat simulaci plánování s využitím SW AMPL a solveru Highs (Huangfu a Hall, 2018). Nejprve stručně popíšeme používaný software a solver a poté přejdeme k testování funkčnosti modelu a zvolených parametrů. Pomocí testovací sady dat vygenerujeme plán pro celý měsíc a zhodnotíme výstup systému.

7.1 AMPL

AMPL (A Mathematical Programming Language) je programovací jazyk a prostředí pro formulaci a řešení matematických programovacích problémů, jako jsou lineární a nelineární programování, celočíselné programování, kvadratické programování a další. Se SW AMPL můžeme zapsat model problému, který chceme řešit, pomocí matematického popisu, který obsahuje omezení a cílovou funkci. Systém poté automaticky překládá uživatelský zápis do formy, kterou mohou přijímat různé lineární, nelineární a celočíselné programovací solvery, jako jsou například CPLEX, Gurobi nebo Highs. AMPL obsahuje mnoho funkcí pro analýzu a vizualizaci výsledků, a také umožňuje import a export dat z různých formátů. Je často používán v akademickém prostředí, ale také v průmyslu pro optimalizaci různých procesů a rozhodování.

Při práci s AMPL se běžně pracuje s několika soubory:

- Modelový soubor (.mod): Tento soubor obsahuje popis matematického modelu v jazyce AMPL. Modelový soubor definuje proměnné, parametry, omezení a cílovou funkci.
- Datový soubor (.dat): Tento soubor obsahuje hodnoty parametrů využívaných v modelu. Datový soubor umožňuje oddělit popis modelu od vstupních dat a usnadňuje tak opakované použití modelu pro různá vstupní data.
- Soubor se skriptem (.run): Tento soubor obsahuje posloupnost příkazů, které se mají vykonat v AMPLu. Skript se používá pro automatizaci postupů, které

je třeba vykonat při řešení modelu, například pro spuštění modelu se specifickými vstupními daty.

7.2 Highs

Highs je open-source řešič (solver) lineárních programů a celočíselných lineárních programů. Je vyvinutý na univerzitě v Edinburghu a podporuje standardní formáty lineárních programů, jako je MPS, LP a MIP. Highs umožňuje řešit velké a složité problémy s různými parametry a omezeními. Highs nabízí různé algoritmy pro řešení lineárních programů, včetně simplexové metody, metody vnějšího bodu a vnitřního bodu, které se automaticky přepínají v průběhu výpočtu podle toho, který algoritmus se ukáže jako nejvhodnější. V AMPLu se Highs volá jako externí solver, což umožňuje snadné a efektivní řešení lineárních programů.

7.3 Testování funkčnosti modelu

V této kapitole se zaměříme na testování funkčnosti našeho modelu plánování painting trainu. Nejprve si připomeneme nastavení parametrů, omezení a koeficientů cílové funkce dle analýzy trénovacích dat. Poté budeme aplikovat tento model na testovací sadu dat, která obsahuje požadavky na skidy na následující měsíc. Tato testovací sada dat byla pravidelně aktualizována a velice dobře představuje reálné požadavky. Při simulaci plánování využijeme software AMPL a solver Highs. Na závěr této kapitoly provedeme celkové zhodnocení výstupu našeho systému plánování painting trainu.

7.3.1 Testovací data a nastavení parametrů modelu

Vstupní parametry modelu, které jsme popisovali v kapitole 6.2.1, jsou uloženy v souboru *DP_testovani.dat*. Pro nalezení vektoru minimálního počtu skidů v painting trainu jsme využili statistické zpracování dat v kapitole 5. Matice požadavků na skidy zahrnují data pro období od 25. 3. 2023 do 21. 4. 2023. Počáteční rozložení painting trainu jsme určili jako rozložení painting trainu posledního lakovacího kola, které jsme našli v kapitole 6.4 o citlivostní analýze.

Matematický model pro plánování painting trainu je popsán v souboru *DP.mod*. Pro vstupní parametry modelu používáme soubor *DP_testovani.dat*. Model spouštíme v softwaru AMPL pomocí souboru *DP.run*, kde lze nastavit koeficienty a a b cílové funkce, nebo hodnotu parametru *dat*. V kapitole o citlivostní analýze jsme zvolili hodnoty koeficientů $a = 9, b = 2$. Parametr dat může nabývat hodnot $\{1,2\}$. Při $dat = 1$ využíváme soubor *DP_uceni.dat*, obsahující trénovací data, a pro $dat = 2$

používáme soubor *DP_testovani.dat* s testovacími daty. Pro simulaci plánování na testovacích datech volíme parametr $dat = 2$.

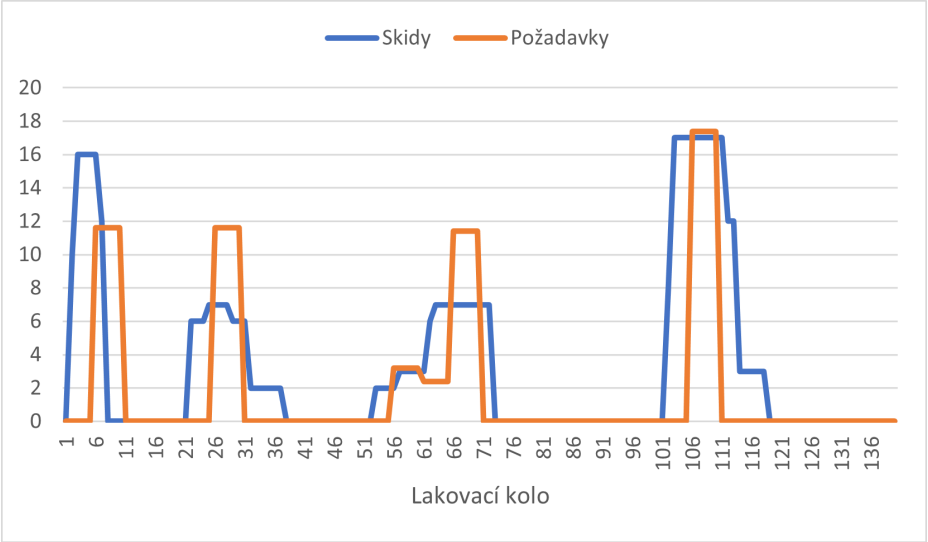
7.3.2 Simulace plánování na testovacích datech

V této kapitole provedeme testování funkčnosti našeho modelu na plánování painting trainu. Pro tento účel máme k dispozici testovací sadu dat, která obsahuje požadavky na skidy na následující 4 týdny. Tyto požadavky byly pravidelně aktualizovány, což nám umožňuje získat přesnou představu o reálných požadavcích. Na základě této datové sady vygenerujeme plán na celé 4 týdny. V praxi můžeme plánovat na delší časový horizont, ale pevný plán by byl obvykle jen na 1 až 3 dny dopředu.

Řešení modelu s testovacími daty pomocí solveru Highs zabere přibližně 20 sekund. Výsledné rozložení painting trainu pro jednotlivá lakovací kola v každém z 28 dnů, na které máme požadavky, je exportováno do souboru *out2.csv*. Nicméně, v tomto textovém souboru jsou data exportována v nepřehledném formátu, kde řádek dat obsahuje parametry skidtype, lakovací kolo a počet skidů daného skidtypu umístěných v daném lakovacím kole. Pro získání přehlednějšího formátu dat používáme software R, který data transformuje do matice, kde řádky představují jednotlivé skidtypy, sloupce lakovací kola a prvky matice počet skidů daných skidtypů umístěných v painting trainu v daném lakovacím kole. Transformace dat probíhá v souboru *transformace_vysledku.R* a výsledná matice je exportována do souboru *finalni_rozlozeni2.csv* (soubor *out.csv* obsahuje finální rozložení trénovacích dat a soubor *finalni_rozlozeni.csv* obsahuje transformovaná data s výslednou maticí).

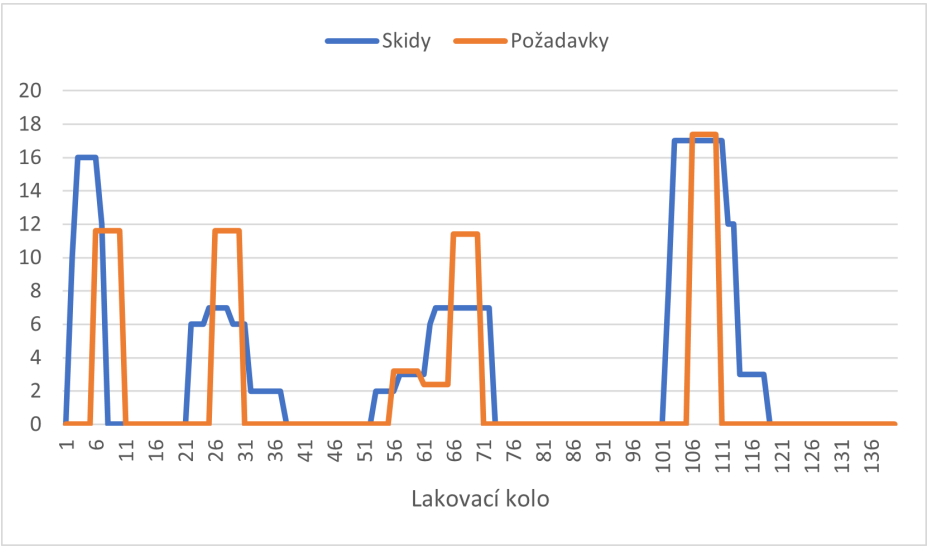
V další části textu budeme sledovat vývoj počtu skidů různých skidtypů v painting trainu během lakování. Zaměříme se blíže na skidtypy *Skid*002, *Skid*065, *Skid*001 a *Skid*169 a na obrázcích 7.1–7.4 zobrazíme vývoj počtu skidů v porovnání s požadovaným množstvím skidů na jednotlivá lakovací kola. Požadovaný počet skidů na lakovací kolo byl vypočítán jako průměrná hodnota z požadavků na celý den. Na ose x jsou jednotlivá lakovací kola, na ose y je počet skidů umístěných v painting trainu a počet požadovaných skidů pro jednotlivá lakovací kola.

Na obrázku 7.1 je znázorněn vývoj počtu skidů skidtypu *Skid*002. Lze vidět, že minimální počet skidů je 0 a požadavky se zde vyskytují pouze 4krát, což je snadno splnitelné. U všech požadavků je patrná jednodenní předvýroba, kdy množství nasazených skidů nemusí dosahovat požadovaného počtu skidů, jelikož část požadavku je splněna v předchozích lakovacích kolech.



Obrázek 7.1: Počet skidů v painting trainu a požadavky na skidtype *Skid*002.

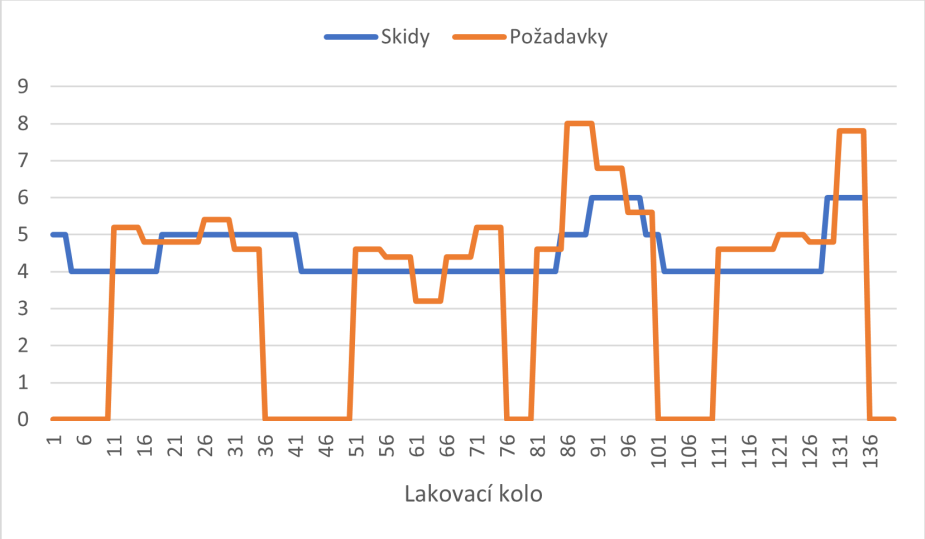
Obrázek 7.2 ukazuje vývoj počtu skidů skidtypu *Skid*065. Znovu zde vidíme, že minimální počet skidů je 0 a že existují 4 větší požadavky na tento skidtype plus 2 menší mezi lakovacími koly 56 a 65, ale počet požadovaných skidů je v tomto případě vyšší než u skidtypu *Skid*002. Nicméně, jako u předchozího skidtypu, s jednodenní předvýrobou není problém požadavky na skidy splnit.



Obrázek 7.2: Počet skidů v painting trainu a požadavky na skidtype *Skid*065.

Z obrázku 7.3 vyplývá, že počet požadovaných skidů na skidtype *Skid*001 zůstává během týdne stabilní a na víkend klesá na nulu. Pro tento skidtype je stanovena mi-

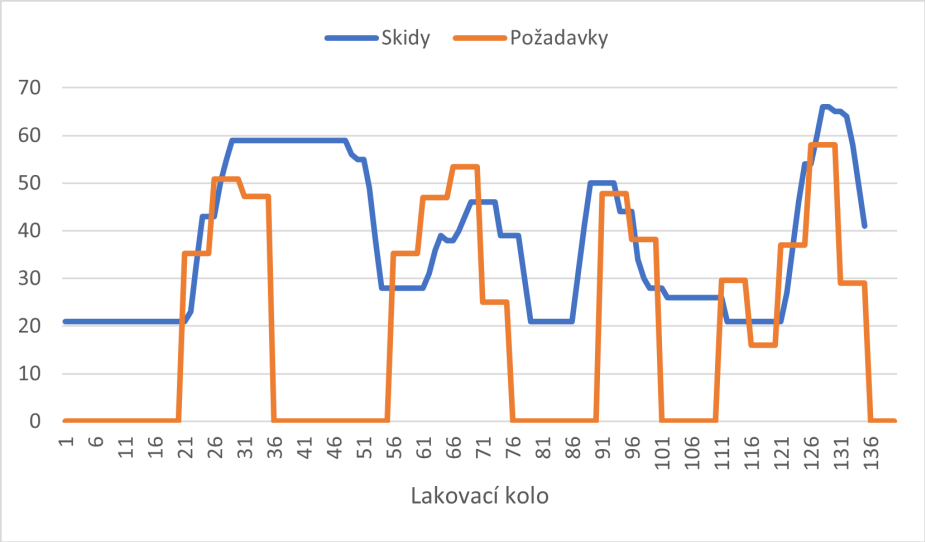
nimální hranice na 4 skidy, což se zdá být vhodné pro pokrytí požadavků na skidy s minimálním počtem změn, přičemž maximální počet skidů v painting trainu dosáhl hodnoty 6.



Obrázek 7.3: Počet skidů v painting trainu a požadavky na skidtype *Skid*001.

Na obrázku 7.4 je zobrazen vývoj počtu skidů skidtypu *Skid*169. Na první pohled vidíme, že požadovaný počet skidů je u tohoto skidtypu vyšší než u předchozích uvedených, kdy nejvyšší požadovaná množství skidů na lakovací kolo se pohybují mezi 50–60. V prvních lakovacích kolech se počet skidů v painting trainu držel na minimální hladině, což je 21 skidů. Poté přišla první vlna požadavků a počet skidů v painting trainu se vyšplhal až na 59 a jelikož nebyla potřeba skidy z painting trainu vyjmout, zůstali na této hladině po dobu 20 lakovacích kol. Poté se počty skidů v painting trainu pohybovaly podobně jako počty požadovaných skidů, kdy dvakrát stouply a vrátily se zpět na minimální hladinu 21 skidů. Na konci měsíce jsme obdrželi nejvyšší požadavky, kdy bylo na jedno lakovací kolo požadováno 58 skidů a počet skidů v painting trainu se vyšplhal až na hodnotu 66.

Během testování našeho modelu na plánování painting trainu jsme byli schopni na každý den nalézt plán pro lakovnu, což potvrzuje funkčnost modelu na testovacích datech. Při výpočtu byl solver Highs nucen strávit delší dobu vypočítáváním rozložení painting trainu 14. dne, kdy bylo celkem provedeno 40 vložení a 40 vyjmutí skidů, což byl maximální počet vložení, který mohl nastat. Níže uvedená tabulka 7.1 obsahuje přesná čísla výměn pro jednotlivá lakovací kola 14. den. Stejný celkový počet výměn byl dosažen také při plánování 19., 26. a 28. dne. Při plánování nebylo potřeba ani jednou navýšit maximální počet výměn a celé plánování s nastavenými



Obrázek 7.4: Počet skidů v painting trainu a požadavky na skidtype *Skid*169.

parametry proběhlo bez problémů a bylo by možné tento model použít v reálném plánování provozu lakovny.

Tabulka 7.1: Celkový počet vložení a vyjmutí skidů v jednotlivých lakovacíhc kolech 14. dne.

Lakovací kolo	Vložení	Vyjmutí
2	10	12
3	10	8
4	10	10
5	10	10

Diskuze 8

V následující kapitole se zaměříme na detailní diskuzi ohledně výsledků našeho modelu a přiblížíme si možné nedostatky, které by mohly ovlivnit jeho přesnost a úspěšnost. V rámci této diskuse navrhneme řadu možností, jak bychom mohli model vylepšit. Kromě toho budeme také zvažovat možné aplikace jak tohoto modelu, tak lineárního programování v praxi a zhodnotíme jeho potenciál pro další využití.

8.1 Vývoj modelu

V kapitole 6.1 se věnujeme procesu tvorby modelu lineárního programování a zaměřujeme se na jednotlivé iterace vývoje, které vedly k výsledné podobě modelu. Nicméně, kapitola se omezuje pouze na popis omezení a parametrů, které jsou obsaženy ve finálním modelu. Během vývoje jsme se často dostávali do slepých uliček a byli nuceni zkoušet různé podmínky nebo vícekrát předělávat některá omezení.

Jako příklad můžeme uvést definici finálního rozložení skidů, kdy jsme v první fázi vývoje definovali počty výměn podle finálního rozložení. V této fázi jsme určili počty výměn jako absolutní hodnotu rozdílu počtu skidů mezi jednotlivými lakovacími koly. Nicméně, nebyli jsme schopni určit, zda se jedná o vložení či vyjmutí skidů, a proto jsme poté změnili definici na opačnou logiku. Nyní je finální rozložení definováno pomocí rozložení painting trainu předchozího lakovacího kola a počtu vložených a vyjmutých skidů.

Dále jsme na začátku vývoje uvažovali o vytvoření defaultního složení painting trainu, ke kterému bychom se na konci dne vraceli. Nicméně jsme zjistili, že nalezení vhodného složení by bylo obtížné a návrat k němu by zvyšoval počet nutných výměn, což by zvýšilo pravděpodobnost nesplnění požadavků. Proto jsme tuto myšlenku přehodnotili a rozhodli se udržovat v painting trainu minimální počet skidů.

Jako poslední příklad uvedeme definici maximálního počtu chybějících nalakovaných skidů na následující den. Původně byla tato proměnná definována bez zohled-

není výroby dalšího dne, což vedlo k situacím, kdy jsme snižovali požadavky na další den, které by nebyly problematické, protože jsme v painting trainu měli dostatečné množství skidů na splnění těchto požadavků bez nutnosti provádět výměny. Na druhé straně však zůstávaly požadavky na skidtypy, které byly sice malé, ale v painting trainu nebyl dostatečný počet skidů. Buď nebyl zařazen žádný skid nebo bylo zařazeno menší množství, než bylo nutné. Proto jsme do definice následně přidali počet skidů, který by byl bez výměny splněn následující den.

8.2 Rozšíření cílové funkce

V kapitole 6 o formulaci optimalizační úlohy jsme definovali proměnné a jejich koeficienty obsažené v cílové funkci. Mezi nimi je i proměnná celkového počtu vložení a vyjmutí skidů mezi lakovacími koly, která slouží k minimalizaci počtu výměn a zajišťuje funkčnost modelu. Nicméně jsme zjistili, že tato proměnná je příliš obecná a existují nedostatky, kterým bychom se mohli věnovat přidáním dalších proměnných do cílové funkce. Tyto proměnné by nám umožnily identifikované nedostatky vyřešit.

Zatímco minimalizujeme počet výměn skidů, nemůžeme ovlivnit, kdy a kde jsou tyto výměny prováděny. Existuje tedy riziko, že některé lakovací kolo bude mít maximální počet výměn, zatímco jiné nebude mít žádnou. Místo toho bychom chtěli dosáhnout rovnoměrného počtu výměn v každém kole. Dalším možným požadavkem by mohlo být, aby se většina výměn skidů uskutečnila na začátku dne a ke konci dne by se snížilo množství změn v painting trainu.

Vzhledem k tomu, že je důležité splnit kritické požadavky na skidy co nejdříve, mohl by být dalším požadavkem optimalizace procesu plnění požadavků na začátku pracovního dne. To znamená, že se zaměříme na efektivní řešení výměn skidů tak, aby co nejdříve byly splněny požadavky na skidy pro daný den. Naopak, požadavky na skidy pro další den by byly splněny až ke konci pracovního dne. V současnosti však tuto prioritu plnění požadavků nelze ovlivnit, a proto by bylo nutné optimalizovat proces tak, aby bylo dosaženo co nejlepšího kompromisu mezi plněním požadavků na daný den a požadavků na následující den.

8.3 Hledání vzorců v citlivostní analýze

V kapitole 6.4 o citlivostní analýze jsme se snažili najít nejlepší kombinaci koeficientů a, b pro minimalizaci počtu potřebných výměn. Provedli jsme analýzu s parametry v intervalu [1,10] a předpokládali jsme, že s vyššími koeficienty se maximální počet výměn nebude navyšovat a najdeme hladinu přijatelných koeficientů. Bohužel jsme

však nepozorovali žádný jasný vzor v chování výsledných hodnot. Proto jsme se pokusili rozšířit analýzu i na vyšší hodnoty koeficientů.

Provedli jsme rozsáhlou citlivostní analýzu pro hodnoty koeficientů, nejprve pro interval [20,200] s krokem 20, a následně pro interval [50,500] s krokem 50. V prvním případě jsme pozorovali, že se s rostoucími koeficienty zvyšuje počet výměn potřebných pro nalezení optimálního řešení. Pouze v 18 případech ze 100 jsme byli schopni najít optimální řešení bez nutnosti navýšení maximálního počtu výměn. V tabulce 8.1 jsou uvedeny výsledky analýzy pro interval [50,500], kde jsme pozorovali, že v 45 případech ze 100 nedošlo k navýšení maximálního počtu výměn. I když jsme získali lepší výsledky s rostoucími hodnotami koeficientů, počty navýšení maximálního počtu výměn se stále nepravidelně rozkládají v tabulce a nenašli jsme konkrétní vzor chování. V některých oblastech tabulky se vyskytuje větší koncentrace nízkých hodnot, na které bychom se mohli zaměřit a provést další výzkum. Takové zkoumání by nám mohlo poskytnout další poznatky a závěry.

Tabulka 8.1: Maximální počet navýšení maximálního počtu výměn.

a\ b	50	100	150	200	250	300	350	400	450	500
50	2	0	3	10	0	10	10	0	0	0
100	0	10	0	0	8	10	0	1	10	0
150	0	7	0	1	0	0	0	7	0	1
200	3	0	0	5	0	6	10	0	0	0
250	0	2	7	0	0	1	5	3	10	0
300	10	0	3	10	0	0	0	0	2	0
350	3	1	0	3	0	2	10	6	1	5
400	10	2	10	2	1	1	1	10	10	0
450	0	0	0	0	0	10	0	0	10	10
500	10	0	1	0	10	0	10	0	0	4

8.4 Vícedenní předvýroba

V kapitolách 6.1.7 a 6.1.8 jsme se zaměřili na přidání požadavků na následující den do modelu, což zahrnuje jednodenní předvýrobu a iterativní hledání řešení s postupným zvyšováním maximálního počtu změn. Nicméně, existuje možnost, že nároky budou příliš vysoké a ani nekonečné zvyšování maximálního počtu výměn nebo jednodenní předvýroba nebude stačit kvůli omezené kapacitě lakovny. V takovém případě by bylo rozumné tvořit plán výroby více dní dopředu a přistoupit k výrobě kritických požadavků klidně i dříve, aby bylo možné je nalakovat včas. Jak jsme již uvedli v kapitole 5.3 o modelu vstupních dat, skidy nemusíme plnit na maximální kapacitu a jednou z možností, jak plnit kritické požadavky, je ruční doplňování poloprázdných skidů.

Tento problém bychom mohli řešit také přímo v modelu. Namísto pouhé jednodenní předvýroby bychom mohli zahrnout vícedenní předvýrobu, kdy požadavky na více dní dopředu by měly nižší prioritu než požadavky na následující den. Snažili bychom se plnit pouze nejkritičtější požadavky na více dní dopředu, aby se snížila pravděpodobnost, že je nebudeme schopni splnit včas. Předvýrobu na více dní dopředu bychom mohli nastavit na libovolný počet dní a pro každý následující den bychom pouze snížili prioritu v cílové funkci. Je však otázka, zda je toto řešení proveditelné s ohledem na skladové kapacity nalakovaných dílů, a proto by bylo nutné provést analýzu tohoto problému.

8.5 Automatizace

V naší práci využíváme hned několik softwarů, mezi které patří R, AMPL a solver Highs. Programy R a AMPL se však momentálně spouští odděleně a komunikace mezi nimi je značně neefektivní. Proto bychom měli najít řešení, jak je propojit do jednoho integrovaného systému, který umožní provést všechny výpočty najednou a minimalizovat lidský zásah a pravděpodobnost chyby. V ideálním případě by tento systém umožnil předávat data mezi softwary a provádět všechny výpočty současně. Pro zajištění efektivity by bylo vhodné exportovat výsledky v požadovaném formátu. V následující kapitole se detailněji zaměříme na export výstupních parametrů našeho modelu.

8.6 Výstupní parametry

V kapitole 7.3.2 o simulaci plánování na testovacích datech jsme popsali způsob exportu finálního rozložení skidtypů v painting trainu pro jednotlivá lakovací kola, což je pro plánovače nejdůležitější informace. Nicméně, v modelu se nachází mnoho dalších parametrů, které by mohly být užitečné pro práci plánovače i pro operátory ve výrobě. V následující kapitole se proto podrobněji zaměříme na export dalších důležitých výstupních parametrů z modelu, které mohou pomoci s optimalizací procesů a s vytvořením efektivnějšího plánu výroby.

Jako první parametry k exportu zmíníme matice výměn skidů, což jsou matice vkládání a vyjímání skidů. Tato informace je velice důležitá pro operátory, kteří se starají o výměnu skidů v painting trainu. Dalším užitečným výstupním parametrem by mohl být celkový počet výměn skidů v jednotlivých lakovacích kolech. Tento parametr by pomohl lépe naplánovat kapacity operátorů potřebných k výměně skidů.

Dále by se mohlo exportovat množství nesplněných skidů pro následující den. Tyto požadavky budou potřeba zaplánovat následující den, a proto jsou pro plánovače velmi důležitou informací. Kromě toho bychom mohli exportovat rozložení painting trainu posledního lakovacího kola, což bude počáteční rozložení pro následující den. Jelikož se výroba často neobejde bez komplikací, je nutné toto počáteční rozložení zkontrolovat a porovnat s reálným stavem lakovny, aby byl plán co nejblíže realitě.

8.7 Komerční solver

V současné době řešíme úlohu lineárního programování pomocí solveru Highs, který je volně dostupný. Nicméně jsme chtěli zjistit, zda by placený solver Gurobi poskytl lepší výsledky a zkrátil celkový čas řešení. S tímto solverem jsme úlohu dokázali vyřešit bez nutnosti navyšování maximálního počtu výměn a celkový čas řešení se snížil na zhruba 4,5 sekundy, tedy přibližně 4krát rychleji než u solveru Highs. Vzhledem k tomu, že rychlost řešení solverem Highs je pro nás dostatečná a ušetříme tak náklady na komerční solver, zůstaneme u něj.

8.8 Plánování položek

Vytvořeným modelem jsme schopni během několika sekund určit počty skidtypů potřebných v painting trainu. Původně jsme zvolili toto řešení, protože jsme se obávali, že rozmisťování položek by bylo výpočetně a časově náročné a pouhé rozložení skidtypů by trvalo několik minut. Nyní jsme však zjistili, že rozložení skidů lze vyřešit v pár sekundách, a tak uvažujeme o tom, že bychom model vylepšili na řešení rozmístění položek.

Tento postup by zahrnoval jak plánování výměn a konkrétní rozmístění skidů do pozic, tak i rozmístění položek na skidy za dodržení veškerých omezení na položky (např. barevné restrikce). Tím bychom ještě výrazněji usnadnili práci plánovačů, kteří po získání výstupu současného modelu ve formě počtů skidů v painting trainu musí rozmístit skidy na pozice a následně na ně umístit požadované položky. Očekáváme, že i pro řešení rozmístění položek bude možné využít volně dostupný solver Highs a najít řešení v řádu několika minut.

8.9 Využití lineárního programování

V LP vidíme velký potenciál pro řešení problémů v oblasti plánování a optimalizace výroby. Využití solveru Highs, který je volně dostupný, nám umožňuje úspěšně řešit

problémy, pokud se nám podaří je formulovat jako lineární programy. V našem případě jsme se zaměřili na lakovny a vytvořili jsme model pro plánování počtu skidů v painting trainu. V předchozí kapitole jsme diskutovali o možnostech rozšíření modelu o plánování pozic skidů a rozmístění položek na skidy.

V naší práci se zaměřujeme na plánování lakovny s proměnlivou velikostí painting trainu. Nicméně, existují i další typy lakoven, u kterých lze využít lineární programování pro plánování. Jedním z nich je lakovna s fixními pozicemi pro skidy v painting trainu. V takovém případě se mohou objevit různá omezení, například rozdělení painting trainu na segmenty a udržování stejného rozložení vlaku v jednotlivých segmentech. Počty výměn skidů pak mohou být omezeny na jednotlivé segmenty.

Lineární programování lze dále v plánování využít k optimalizaci libovolných loop conveyor systémů. Tyto systémy umožňují přepravovat materiál po kruhové trase pomocí pohyblivého dopravníku napojeného na několik stanic, kde se provádí různé úkony, jako například kontrola kvality, balení nebo naskladňování.

Dalším příkladem může být plánování výroby produktů s různými variantami, jako jsou například barevné varianty nebo varianty s různými funkcemi. Lineární programování může být využito k maximalizaci využití výrobních prostředků a minimalizaci nákladů na výrobu jednotlivých variant.

Závěr 9

V rámci této diplomové práce jsme se zabývali řešením problému plánování provozu průmyslové lakovny pro výrobce dílů pro automotive v Německu. Na začátku naší práce jsme poskytli teoretický úvod, který zahrnoval představení statistických metod a základů lineárního programování. Poté jsme se podrobněji zaměřili na charakteristiku procesů a plánování lakovny a představili jsme sadu omezení a parametrů, které jsou s tímto plánováním spojeny.

V následující části práce jsme se zaměřili na přípravu dat, přičemž jsme nejprve popisovali ideální stav dat a následně řešili případné problémy, které se mohou vyskytnout při získávání dat. Dále jsme představili data, která jsme obdrželi od zákazníka, a detailně popisovali postup získávání těchto dat. Poté jsme přistoupili k statistickému zpracování získaných dat, kdy jsme je nejprve sumarizovali a provedli základní kontrolu. Následně jsme data adekvátně připravili pro další statistickou analýzu.

V rámci statistického zpracování dat jsme na základě historických dat vytvořili model vstupních dat. Hledali jsme stabilní skidtypy, na něž pravidelně dostáváme požadavky od zákazníka. Pro tyto stabilní skidtypy jsme zjistili průměrný počet požadovaných skidtypů na lakovací kolo v painting trainu. V případě pravidelných a konzistentních požadavků na skidtype je pro nás výhodné udržovat v painting trainu alespoň průměrný počet požadovaných skidtypů. Tím minimalizujeme zbytečné výměny těchto skidtypů. Další scénář, který může nastat, je absence pravidelných požadavků na skidtype, ale pokud přijde nějaký požadavek, bude požadováno velké množství skidů. Pokud bychom v painting trainu neudržovali určitou hladinu tohoto skidtypu, museli bychom provést velké množství výměn najednou, což by mohlo ohrozit splnění požadavku.

V našem modelu jsme zavedli jednodenní předvýrobu právě za účelem plnění požadavků na velké množství skidů a nevadí nám, že některé dny skidy pojedou prázdné. Důležité je, že máme v painting trainu skidy neustále k dispozici, což umožňuje

okamžité zahájení předvýroby bez nutnosti provádět výměny skidů. Průměrné požadované množství stabilních skidtypů je dále v práci využito v optimalizační úloze jako minimální počet skidtypů, který chceme v painting trainu pravidelně udržovat.

Následně jsme provedli verifikaci modelu vstupních dat. Vzhledem k tomu, že požadavky zákazníků se mohou z různých důvodů měnit v průběhu času, je důležité pravidelně aktualizovat naše plánovací a optimalizační modely. Po jednom měsíci jsme aktualizovali model vstupních dat a zjistili jsme, že požadavky na některé skidtypy klesly. Je tedy nezbytné upravit množství těchto skidtypů, které udržujeme v painting trainu. Celkový minimální počet skidů se v našem modelu snížil ze 127 na 120.

V další části jsme se zaměřili na formulaci optimalizační úlohy. Nejprve jsme popsali vývoj modelu lineárního programování, který začal s jednoduchým modelem, kde byly počty skidů v painting trainu voleny náhodně tak, aby byly splněny požadavky. Postupně jsme iterativně vylepšovali model až k finálnímu řešení, které plánuje rozložení painting trainu s přihlédnutím k reálným podmínkám a omezením.

Následně jsme definovali všechny potřebné parametry, omezení a cílovou funkci pro finální model. Pro cílovou funkci jsme provedli citlivostní analýzu výběru koeficientů proměnných. Hledali jsme takové koeficienty, které minimalizují maximální povolený počet výměn skidů v painting trainu a zároveň umožňují nalézt řešení co nejrychleji. Vzhledem k tomu, že priorita splnění požadavků je vyšší než minimalizace počtu výměn skidů, přidělili jsme koeficient 1 pro celkový počet výměn a zaměřili jsme se na hledání koeficientů proměnných pro účel splnění požadavků. Pro proměnnou maximálního počtu nesplněných skidů na následující den jsme stanovili koeficient 9 a koeficient 2 pro celkový počet nesplněných skidů na následující den. Tato kombinace nám umožnila dosáhnout řešení bez nutnosti zvyšovat maximální počet výměn skidů, a pro trénovací data jsme dosáhli řešení během 17,4 sekundy.

S využitím vytvořeného modelu s nastavenými parametry jsme provedli simulaci reálného provozu lakovny. K tomu jsme použili testovací data, která obsahovala požadavky na následující měsíc a byla pravidelně aktualizována, aby co nejlépe odpovídala skutečnosti. Nejprve jsme podrobně popsali software AMPL a solver Highs pro řešení celočíselných lineárních programů. Poté jsme oba SW využili k otestování funkčnosti modelu na testovacích datech.

Řešení modelu s použitím solveru Highs trvalo přibližně 20 sekund a v žádném dni jsme nemuseli zvyšovat maximální počet výměn skidů. Výsledné rozložení painting trainu bylo exportováno a transformováno do přehledné matice, kde řádky předsta-

vovaly jednotlivé skidtypy, sloupce představovaly lakovací kola a jednotlivé prvky matice zobrazovaly počet skidů jednotlivých skidtypů v daném lakovacím kole.

V rámci práce jsme dále podrobně analyzovali a vizualizovali vývoj počtu skidů v painting trainu pro 4 vybrané skidtypy. Z grafů bylo dobře patrné využití jednodenní předvýroby a vhodné stanovení minimálního počtu skidů v painting trainu. Během testování našeho modelu plánování painting trainu jsme byli schopni pro každý den nalézt plán pro lakovnu, což potvrzuje funkčnost modelu na testovacích datech.

V závěrečné části naší práce jsme se věnovali diskuzi o překážkách, které jsme museli překonat při vývoji modelu, a také jsme identifikovali možné nedostatky a vylepšení. Prováděli jsme důkladnou citlivostní analýzu s cílem najít optimální hranici koeficientů, která by nám umožňovala dosahovat stabilních řešení bez nutnosti zvyšovat maximální počet výměn skidů. Dále jsme diskutovali o řešení vyšších požadavků, na které lakovna nemá dostatečnou kapacitu, a navrhli jsme přístup vícedenní předvýroby jako možné řešení. Dalším tématem naší diskuse byla automatizace procesu hledání řešení. V současné době je komunikace mezi různými SW neefektivní, a proto chceme vyvinout integrovaný systém, který umožní provádět všechny výpočty najednou a minimalizuje lidský zásah a možnost chyby. Rovněž jsme probírali možnosti rozšíření exportu.

V současnosti exportujeme pouze finální rozložení painting trainu, i když v modelu existuje mnoho užitečných informací nejen pro plánovače, ale i pro operátory výroby. Pokusili jsme se také nahradit solver Highs komerčním solverem Gurobi, který našel řešení rychleji, ale rozdíl byl pouze v jednotkách sekund. Solver Highs tedy dosahuje uspokojivých výsledků pro naše potřeby. Nakonec jsme diskutovali o možném zdokonalení modelu v oblasti konkrétního plánování pozic skidů a následném rozmístění položek na skidy, a také o dalším využití lineárního programování v oblasti plánování a optimalizace výroby. Konkrétně jsme probírali další druhy lakoven, loop conveyor systémy a plánování výroby produktů s různými variantami.

Cílem této práce bylo poskytnout ucelený pohled na plánování průmyslové lakovny a jeho optimalizaci, a to na konkrétním příkladu výrobce dílů pro automotive v Německu. Závěrem lze konstatovat, že se nám podařilo úspěšně vyřešit problém plánování lakovny a navrhnout optimalizovaný model, který může být využit v praxi pro zlepšení výrobního procesu.

Zdroje

- REIF, Jiří. Metody matematické statistiky. Plzeň: Západočeská univerzita, 2000. ISBn 80-7040-302-7.
- VANDERBEI, Robert J. Linear programming : foundations and extensions. 5. vyd. Springer, 2020. ISBn 978-3-030-39414-1.
- LAMOŠ, F.; POTOCKÝ, R. Pravdepodobnosť a matematická štatistika, štatistické analýzy. Bratislava: Alfa, 1989.
- Plastics Painting, 2005. eisenmann.com, 2023-04-21. Dostupné z: https://cdn2.hubspot.net/hub/133998/file-17444811-pdf/docs/plastic paint systems.pdf.
- SHAPIRO, S. S.; WILK, M. B.; CHEN, H. J. A comparative study of various tests for normality. J. Amer. Statist. Assoc. 63, 1968.
- WOLSEY, Laurence A. Integer Programming. John Wiley Sons, Inc, 1998.
- LIKEŠ, J.; MACHEK, J. Matematická statistika. Praha: SNTL, 1983.
- JABLONSKÝ, J. Programy pro matematické modelování. Praha: Oeconomica, 2011. ISBn 978-80-245-1810-7.
- R Core Team. (2021). R: A Language and Environment for Statistical Computing [software]. R Foundation for Statistical Computing. Vídeň, Rakousko. URL: https://www.R-project.org/.
- AMPL Optimization LLC. AMPL (Verze: 3.6.10.202208131811) [Software]. (2019). Dostupné z https://ampl.com/.
- Huangfu, Q. a Hall, J. A. J. (2018). Paralelizace revidované simplexové metody. Mathematical Programming Computation, 10(1), 119-142. DOI: 10.1007/s12532-017-0130-5.

Seznam obrázků

3.1 Lakovací proces lakovací linky. 18
3.2 Lakování zadních nárazníků. Zdroj: Plastics Painting (2005). 19

5.1 Histogram počtu požadavků na položky za den. 29
5.2 Boxploty počtu požadavků na položky v jednotlivých dnech v týdnu. . 30
5.3 Boxploty počtu požadavků na skidy v jednotlivých dnech v týdnu. . . 31
5.4 Boxploty počtu požadovaných skidů v jednotlivých dnech v týdnu. . . 32
5.5 Boxploty počtu požadovaných skidů v jednotlivých dnech v týdnu. . . 33

7.1 Počet skidů v painting trainu a požadavky na skidtype *Skid*002. 56
7.2 Počet skidů v painting trainu a požadavky na skidtype *Skid*065. 56
7.3 Počet skidů v painting trainu a požadavky na skidtype *Skid*001. . . . 57
7.4 Počet skidů v painting trainu a požadavky na skidtype *Skid*169. 58

Seznam tabulek

5.1 Statisticky významné skidtypy a p-hodnota Wilcoxonova testu. 35
5.2 Statisticky významné skidtypy, průměrný požadovaný počet a minimální počet v painting trainu. 36
5.3 Statisticky významné skidtypy, průměrný požadovaný počet na lakovací kolo a minimální počet v painting trainu. 37

6.1 Maximální počet navýšení maximálního počtu výměn. 51
6.2 Čas výpočtu pro různé kombinace koeficientů. 51

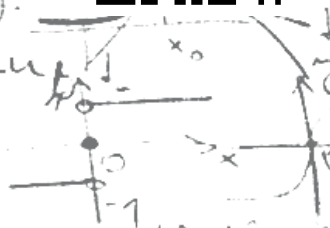
7.1 Celkový počet vložení a vyjmutí skidů v jednotlivých lakovacíhc kolech 14. dne. 58

8.1 Maximální počet navýšení maximálního počtu výměn. 61

za jednotku času pro $x \in \langle x_0, x_0+h \rangle$
okamžitá změna veličiny y v čase $x = x_0$. Př. a) Nechť
vztahem $s = s(t)$ — $s(5)$ — $s(6)$ — t_0 — t_1. Pak $\frac{s(t_0+h)-s(t_0)}{h}$ je prů
hmotného bodu v čase t_0. b) Nechť vstah $q = q(t)$, u
čase t ⟹. Pak $\frac{q(t+\Delta t)-q(t_0)}{\Delta t}$ je průměrná změna te

$i(t_0)$. c) Hmotnost radioak
jednotku času je

existuje. Pokud existuje jen
v bodě $A = [x_0, f(x_0)]$.



Normála grafu funk
$k_n = -\frac{1}{k_t}$, pro
$\frac{1}{f'x}$
x), je tečna rovno

A a f je spojitá v x_0)
τ (resp. x).
f nabývá v bodě $x_0 \in D(f)$ lokálního minima (resp. maxim
—"— ostrého lokálního minima (resp. maxima) $\overset{\text{def}}{\Longleftrightarrow} \exists \delta > 0$