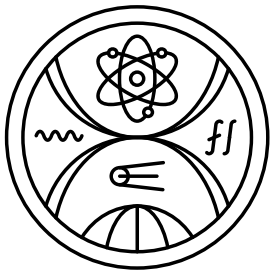
UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY



Efektívne optimalizačné stratégie pri výrobe elektrickej energie vo vodnej elektrárni

DIPLOMOVÁ PRÁCA

2024 Bc. Andrej FUKAS

UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

Efektívne optimalizačné stratégie pri výrobe elektrickej energie vo vodnej elektrárni

DIPLOMOVÁ PRÁCA

Študijný program:	Ekonomická a finančná matematika
Študijný odbor:	1114 Aplikovaná matematika
Školiace pracovisko:	Katedra aplikovanej matematiky a štatistiky
Vedúci práce:	doc. RNDr. Zuzana Chladná, Dr.

Bratislava 2024 **Andrej Fukas**





Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Bc. Andrej Fukas
Študijný program: ekonomicko-finančná matematika a modelovanie (Jednoodborové štúdium, magisterský II. st., denná forma)
Študijný odbor: matematika
Typ záverečnej práce: diplomová
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Efektívne optimalizačné stratégie pri výrobe elektrickej energie vo vodnej elektrárni
Effective optimization strategies in hydropower energy generation

Anotácia: Hlavnou úlohou obchodného oddelenia v spoločnosti Slovenské Elektrárne(SE) a.s. je riadenie portfólia výrobných zdrojov za účelom maximalizácie zisku z predaja elektriny pri zachovaní všetkých technických požiadaviek. V posledných rokoch boli v SE vyvinuté matematické modely, ktoré sa využívajú v každodennom rozhodovaní pri nasadení elektrární a zobchodovaní vyrobeného objemu (tzv. unit commitment problem). Vo všeobecnosti ide o optimalizačné problémy s veľkým množstvom binárnych rozhodovacích premenných. Pre praktické využitie týchto matematických modelov je rozhodujúci prijateľný výpočtový čas. Dosiahnuť požadovanú rýchlosť výpočtu je vzhľadom na NP úplnosť týchto problémov netriviálny problém. Diplomová práca bude spočívať v hľadaní alternatívnych formulácií daných optimalizačných úloh s cieľom znížiť výpočtový čas, ktorý je potrebný na nájdenie optimálneho riešenia s prijateľnou mierou presnosti pri zachovaní dostatočnej presnosti aproximácií reálnych fyzikálnych vzťahov ovplyvňujúcich rozhodovací proces.

Vedúci: doc. RNDr. Zuzana Chladná, Dr.
Katedra: FMFI.KAMŠ - Katedra aplikovanej matematiky a štatistiky
Vedúci katedry: doc. Mgr. Igor Melicherčík, PhD.

Dátum zadania: 10.01.2023

Dátum schválenia: 11.01.2023 prof. RNDr. Daniel Ševčovič, DrSc.
garant študijného programu

.....
študent

.....
vedúci práce

Poďakovanie Na tomto mieste by som chcel vyjadriť úprimnú vďaku svojej vedúcej diplomovej práce doc. RNDr. Zuzane Chladnej, Dr, za jej podporu, odborné vedenie a trpezlivosť. Rovnako by som sa chcel poďakovať kolegom zo Slovenských elektrární za odbornú pomoc a konzultácie. Nakoniec, chcem vyjadriť svoju hlbokú vďačnosť svojim rodičom a mojej priateľke za emocionálnu podporu.

Abstrakt v štátnom jazyku

FUKAS, Andrej: Efektívne optimalizačné stratégie pri výrobe elektrickej energie vo vodnej elektrárni [Diplomová práca], Univerzita Komenského v Bratislave, Fakulta matematiky, fyziky a informatiky, Katedra aplikovanej matematiky a štatistiky; školiteľ: doc. RNDr. Zuzana Chladná, Dr., Bratislava, 2024, 61s.

Táto práca sa zaoberá optimalizáciou prevádzky vodných elektrární Vážskej kaskády. Cieľom je minimalizovať odchýlky celkovej výroby elektrickej energie od vopred daného plánu výroby pre jednotlivé hodiny obchodovacieho dňa. V práci predstavíme dve triedy numerických prístupov na riešenie nastolenej úlohy. Prvou je formulácia problému ako úlohy zmiešaného celočíselného lineárneho programovania (MILP) a zmiešaného celočíselného kvadratického programovania (MIQCP). Druhou testovanou triedou optimalizačných techník sú heuristické prístupy. Spomedzi nich sme sa zamerali na využitie genetických algoritmov a simulovaného žíhania. V práci predstavíme aj vlastný prístup: hybrid medzi MILP formuláciou a heuristikami, ktorý sme nazvali lineárny iteratívny model. Jednotlivé výpočtové techniky porovnávame na základe presnosti a dĺžky výpočtového času.

Kľúčové slová: Optimalizácia prevádzky, MILP, MIQCP, genetické algoritmy, simulované žíhanie

Abstract

FUKAS, Andrej: Effective optimization strategies in hydropower energy generation [Master Thesis], Comenius University in Bratislava, Faculty of Mathematics, Physics and Informatics, Department of Applied Mathematics and Statistics; Supervisor: doc. RNDr. Zuzana Chladná, Dr, Bratislava, 2024, 61p.

This paper addresses various approaches towards optimising the operation of hydroelectric power plants in the Váh River cascade. The aim is to minimize the deviations of the total power plant production from a predetermined production plan for respective hours, while the problem is solved over a time horizont of one day. Optimisation involves determining specific water flows through each power plant for every hour of the day. The problem is addressed using two different approaches in the work. The first is the formulation of the problem as a Mixed Integer Linear Programming (MILP) problem and Mixed Integer Quadratic Constrained Programming (MIQCP) problem. The second type of approach to solving the problem is the application of heuristic algorithms. Among them, we focused on the use of genetic algorithms and simulated annealing. The paper also introduces our own approach: a hybrid between MILP formulation and heuristics, which we named the linear iterative model. We compare individual computational techniques based on accuracy and computation time length.

Keywords: Unit commitment problem, MILP, MIQCP, genetic algorithms, simulated annealing

Obsah

Úvod **9**

1 Vážska kaskáda **11**

1.1 Matematický model 12

1.1.1 Funkcia výkonu v priestore spádu a prietoku 15

1.1.2 Funkcia výkonu v priestore prietoku a objemu hornej a dolnej nádrže 18

2 Metódy riešenia **20**

2.1 Lineárna a kvadratická aproximácia 21

2.1.1 Trojuholníková metóda 22

2.1.2 MILP formulácia modelu Vážskej kaskády 24

2.1.3 Výpočtové techniky 26

2.2 Heuristiky 29

2.2.1 Genetické algoritmy 30

2.2.2 Simulované žíhanie 31

3 Priebežné porovnanie optimalizačných metód **34**

3.1 Zhodnotenie MILP prístupu 35

3.1.1 Model v priestore prietoku a spádu s lineárnou aproximáciou 36

3.1.2 Model v priestore prietoku a objemu hornej a dolnej hladiny s lineárnou aproximáciou 38

3.1.3 Model v priestore prietoku a spádu s kvadratickou aproximáciou . . 38

3.1.4 Model v priestore prietoku a objemu hornej a dolnej hladiny s kvadratickou aproximáciou 40

3.1.5 Porovnanie modelov MIP modelov v čase 41

3.2 Lineárny iteratívny model v priestore prietoku 43

3.3 Zhodnotenie heuristických prístupov 45

3.3.1 Aplikácia genetického algoritmu 46

3.3.2 Aplikácia simulovaného žíhania 49

3.4 Vzájomné porovnanie 52

4 Finálne zhrnutie a porovnanie **54**

Záver **58**

Literatúra **60**

Úvod

Elektrina patrí medzi komodity, bez ktorých si nevieme predstaviť svoj moderný život. V minulosti sa elektrická energia vyrábala prevažne z fosílnych palív. Avšak narastajúci dopyt a zvyšujúce sa množstvo emisií CO_2 v atmosfére si vynútilo hľadanie alternatív, ktoré sú z dlhodobého hľadiska udržateľné a súčasne šetrné k životnému prostrediu.

Kľúčovú úlohu pri zmierňovaní klimatických zmien zohráva využitie obnoviteľných zdrojov energie, akými sú napr. solárne, veterné či vodné elektrárne. Slovenská republika sa zaviazala do roku 2030 zvýšiť podiel obnoviteľných zdrojov pri výrobe elektrickej energie na 22% zo súčasných 17.5% ([4]). Najväčší podiel na výrobe tzv. čistej elektrickej energie na Slovensku majú vodné elektrárne. Najvýznamnejším prevádzkovateľom vodných elektrární v Slovenskej republike sú Slovenské elektrárne (SE).

Do portfólia vodných elektrární pod správou SE patrí aj systém 22 vodných elektrární vybudovaných na rieke Váh. Celkový inštalovaný výkon Vážskej kaskády je 1515MW. Pre lepšiu predstavu o význame tejto kaskády vodných elektrární spomeňme, že v roku 2023 bolo z Vážskej kaskády do elektrickej siete dodané množstvo 1.9TWh, čo predstavuje približne 10% z celkovej dodanej elektriny od Slovenských elektrární a 7% celkovej vyrobenej elektriny na Slovensku.

Narozdiel od iných typov obnoviteľných zdrojov (solárne, veterné) výroba elektrickej energie vo vodnej elektrárni je dostatočne flexibilná. To znamená, že množstvo vyrobenej energie je možné prispôsobovať požiadavkám na výrobu a nie je závislé od poveternostných podmienok. Nakoľko elektrickú energiu nie je možné uskladniť, flexibilita vodných elektrární zohráva dôležitú úlohu pri obchodovaní.

Cieľom trhu s elektrinou je zabezpečiť spoľahlivú dodávku elektrickej energie a uspokojiť dopyt. Elektrická energia sa predáva dopredu na jednotlivé hodiny v rámci dňa.

Ako sme už spomenuli, elektrická energia je takmer neuskladniteľná. Preto, ak sa na daný časový úsek zobchoduje určitý objem elektrickej energie, je potrebné, aby bol v daný čas aj doručený presný objem elektrickej energie do siete. Neočakávané výchylky vo výrobe, resp. v dopyte, by mohli narušiť stabilitu siete. V takýchto prípadoch je potrebné flexibilne prispôsobiť výrobu zmeneným podmienkam. Efektívna optimalizácia prevádzky vodných elektrární v tomto procese tak zohráva kľúčovú úlohu.

Predložená diplomová práca nadväzuje na sériu diplomových prác, ktoré sa zaoberali

optimalizáciou prevádzky vodných elektrární na Slovensku. K problému pristupovali z rôznych hľadísk. V práci [2] sa autor zaoberal manažmentom prečerpávacej elektrárne Čierny Váh. Táto elektráreň nie je súčasťou Vážskej kaskády, a preto optimalizáciu jej prevádzky bolo možné realizovať bez ohľadu na prevádzku iných vodných elektrární. Za predprípravu optimalizácie Vážskej kaskády možno považovať prácu [3]. Autor sa v nej venoval optimalizácii prevádzky vodnej elektrární Kráľová, poslednej z elektrární Vážskej kaskády. Poslednou z tejto série prác je diplomová práca [1]. Autor v nej sa zaoberal optimalizáciou prevádzky celej Vážskej kaskády s cieľom dosiahnuť maximálny zisk z predaja elektriny na spotovom trhu, pri dodržaní všetkých prevádzkových a technických obmedzení. Jeho cieľom bolo nájsť taký spôsob výpočtu, ktorý by spĺňal požiadavku na presnosť a rýchlosť výpočtu.

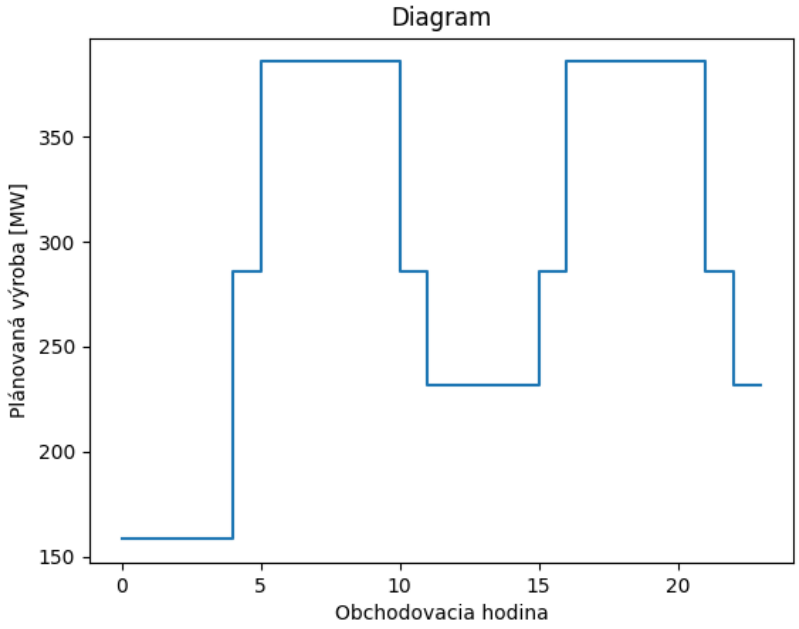
Aj táto diplomová práca sa zaoberá problematikou optimálnej prevádzky Vážskej kaskády. Narozdiel od práce [1] pod pojmom optimálna prevádzka nebudeme uvažovať plán výroby, ktorý vedie k maximálnemu zisku počas daného dňa, ale taký, ktorý minimalizuje odchýlky agregovanej výroby od dopredu daného plánu výroby. Optimalizačný problém tohto typu je v anglickej literatúre známy po pojmom *hydro unit commitment and loading (HUCL) problem*. V práci [5] bol takýto optimalizačný problém využitý na naplánovanie prevádzky Brazílskych sériovo zapojených vodných elektrární na deň dopredu. V Brazílii sa nachádzajú pomerne veľké nádrže oproti Vážskej kaskáde, preto v tejto práci postačoval zjednodušený model, kde nebolo potrebné modelovať výšky hladín a dotokové časy s veľkou presnosťou.

Práca je rozdelená do štyroch kapitol. V prvej kapitole predstavíme matematický model Vážskej kaskády. Z matematického hľadiska ide o deterministický dynamický model s viacerými stavovými a riadiacimi premennými. Jadro modelu prevezmeme z práce [1] a modifikujeme ho do požadovaného tvaru. V druhej kapitole predstavíme relevantné numerické techniky, ktoré sa dajú považovať za vhodných kandidátov na riešenie nastoleného problému. V tretej kapitole zhrnieme výsledky realizovaných numerických experimentov. V štvrtej kapitole porovnáme presnosť a rýchlosť uvažovaných prístupov pri stanovaní optimálnej prevádzky Vážskej kaskády na vybraných obchodovacích dňoch.

1 Vážska kaskáda

V prvej Kapitole si stručne predstavíme optimalizačný model, ktorým reprezentujeme problém optimálnej prevádzky Vážskej kaskády. Pod pojmom optimálna prevádzka budeme rozumieť takú stratégiu výroby, ktorá minimalizuje odchýlky agregovaného výkonu elektrární od vopred daného plánu výroby, tzv. diagramu. Pre lepšiu predstavu sme na Obr. 1 graficky znázornili príklad diagramu pre jeden obchodovací deň. Diagramy majú zväčša v danom dni dva vrcholy v časoch, keď je dopyt po elektrickej energii vysoký. Dopyt je typicky nízky v neskorých nočných a skorých ranných hodinách.

Prevádzku elektrární optimalizujeme na jeden deň vopred pre každú hodinu dňa. Keďže sú jednotlivé elektrárne Vážskej kaskády sériovo zapojené, tak prietok cez hornú elektráreň ovplyvní hladinu, a teda aj efektivitu produkcie nasledujúcej elektrárne. Z dôvodu prepojenia je potrebné stanoviť výrobu všetkých elektrární naraz. Pri plánovaní optimálnej prevádzky je potrebné zohľadniť všetky špecifiká jednotlivých elektrární. Navyše, výroba musí byť naplánovaná tak, aby spĺňala všetky prevádzkové a technické obmedzenia. Zohľadnením všetkých podmienok dostaneme komplexný optimalizačný problém, ktorý si predstavíme v nasledujúcej časti.



Obr. 1: Diagram výroby

1.1 Matematický model

Model optimálneho riadenia prevádzky Vážskej kaskády bol už predstavený v práci [1]. Nakoľko však v práci [1] bolo cieľom maximalizovať zisk z predaja a nie minimalizovať odchýlku od diagramu, pre účely tejto práce bolo potrebné upraviť niektoré časti modelu.

V prvom rade bolo potrebné modifikovať účelovú funkciu. Upravená účelová funkcia bude mať tvar

$$\begin{aligned}\max \quad & \sum_{t=0}^{T-1} -\left|\left(\sum_{p=0}^{plant-1} P_{t,p}\right) - D_t\right| - \sum_{t=0}^{T-1}\sum_{p=0}^{plant-1} Q_spill_pen_{t,p} Qspill_plant_{t,p} \\ & - \sum_{t=0}^{T-1}\sum_{r=0}^{res-1} spill_pen_{t,r}(Qspill_{t,r} - outflows_bio_{t,r}) \\ & - \sum_{t=0}^{T-1}\sum_{g=0}^{gen-1} tg_startup_cost_g \; TG_startup_{t,g}\end{aligned} \tag{1}$$

Prvý výraz v účelovej funkcii (1) reprezentuje odchýlku celkovej výroby Vážskej kaskády ($\sum_{p=0}^{plant-1} P_{t,p}$) od požadovaného diagramu (D_t) v čase t . Do účelovej funkcie sme ďalej zahrnuli aj penalizácie za jalový prietok, keďže od optimálneho riadenia prevádzky očakávame, že nebude plytvať vodou. Tieto penalizácie sú zachytené pomocou parametrov $Q_spill_pen_{t,p}$ a $spill_pen_{t,p}$. Ďalej treba zahrnúť ešte penalizáciu za spúšťanie generátorov, keďže častým zapínaním a vypínaním sa generátory opotrebúvajú. Parameter $tg_startup_cost_g$ vyjadruje túto cenu zapnutia generátora.

Optimalizácia prevádzky podlieha viacerým prevádzkovým a technickým obmedzeniam. Navyše je potrebné zohľadniť dynamiku vývoja niektorých premenných v čase (napr. objemu v nádržiach a prevádzkovému stavu generátora). Tieto podmienky môžeme vyjadriť nasledovne

$$P_{t,p} = \begin{cases} F_1^p(Q_{t,p}, H_{t,p}) \\ F_2^p(Q_{t,p}, HH_V_{t,p}, DH_V_{t,p}) \end{cases} \quad t=0,..,T\text{-}1;\ p=0,..,plant\text{-}1 \tag{2}$$

$$\begin{aligned} V_{t+1,r} &= V_{t,r} + z(-Q_{t,r}^{out} - Q_spill_{t,r} + infl_{t,r} + Q_{t,r}^{in}) \\ & t=0,..,T\text{-}1;\ p=0,..,plant\text{-}1, r = 0,..,res\text{-}1 \end{aligned} \tag{3}$$

$$HH_{t,p} = F_5^p(V_{t,r}) \quad t=0,..,T\text{-}1;\ p=0,..,plant,\ r = 0,..,res\text{-}1 \tag{4}$$

$$DH_{t,p} = F_6^p(Q_{t,p}, HH_{t,p+1}) \quad t = 0, 1, \ldots, T-1;\ p = 0, 1, \ldots, plant-1, \tag{5}$$

$$V_{0,r} = v_{0,r} \quad r=0,..,res\text{-}1 \tag{6}$$

$$TG_{t,g} = \begin{cases} 1 & \text{ak } TG_{t-min(min_TG_up_time,t),g} = 1 \\ 1 & \text{ak } TG_{t-1,g} = 1 \text{ a } TG_shutdown_{t,g} = 0 \\ 1 & \text{ak } TG_{t-1,g} = 0 \text{ a } TG_startup_{t,g} = 1 \\ 0 & \text{ak } TG_{t-1,g} = 0 \text{ a } TG_shutup_{t,g} = 0 \\ 0 & \text{ak } TG_{t-1,g} = 1 \text{ a } TG_shutdown_{t,g} = 1 \end{cases} \tag{7}$$

t=1,..,T-1; g=0,..,gen-1

$$TG_{0,g} = \begin{cases} 1 & \text{ak } TG_startup_{0,g} = 1 \\ 0 & inak \end{cases} \quad g=0,..,gen\text{-}1 \tag{8}$$

$$TG_startup_{t,g} = 0 \text{ ak } TG_{t-1,g} = 1 \quad t=1,..,T\text{-}1,\ g=0,..,gen\text{-}1 \tag{9}$$

$$TG_shutdown_{t,g} = 0 \text{ ak } TG_{t-1,g} = 0 \quad t=1,..,T\text{-}1,\ g=0,..,gen\text{-}1 \tag{10}$$

$$Q_{t,p} \leq \sum_{g=firstTG_p}^{lastTG_p} TG_{t,g} qtg_{t,g}^{max} \quad t=0,..,T\text{-}1;\ p=0,..,plant\text{-}1 \tag{11}$$

$$Q_{t,p} \geq \sum_{g=firstTG_p}^{lastTG_p} TG_{t,g} qtg_{t,g}^{min} \quad t=0,..,T\text{-}1;\ p=0,..,plant\text{-}1 \tag{12}$$

$$Q_{t,r}^{out} = Q_{t,p} + Q_spill_plant_{t,p} \quad t=0,..,T\text{-}1;\ r=0,..,res\text{-}1,\ p \text{ pre elektráreň} \tag{13}$$

pod nádržou r

$$Q^{out}_del_{t,r} = \begin{cases} Q_{t,r}^{out} + F_3(Q_spill_{\tau,r}) & \text{ak } r \in SK_w_delay \\ Q_{t,r}^{out} + Q_spill_{t,r} & \text{inak} \end{cases} \tag{14}$$

t=0,..,T-1; r= 0,..,res-1, τ*=0,..,t*

$$Q_{t,r+1}^{in} = \begin{cases} F_4(Q^{out}_del_{\tau,r}) & \text{ak } r \in res_w_delay \\ Q^{out}_del_{t,r} & \text{inak} \end{cases} \tag{15}$$

t=0,..,T-1; r=0,..,res-2, τ*=0,..,t*

$$Q_{t,p} + Q_spill_plant_{t,p} = Q_{t,p+1} + Q_spill_plant_{t,p+1} \quad t=0,..,T\text{-}1;\ p \in canal_scheme \tag{16}$$

$$\sum_{t=0}^{T-1} Q_{t,r}^{out}/T \leq max_avg_out_r \quad r=0,..,res\text{-}1 \tag{17}$$

$$\sum_{t=0}^{T-1} Q_{t,r}^{out}/T \geq min_avg_out_r \quad r=0,..,res\text{-}1 \tag{18}$$

$$q_p^{diff} \geq |Q_{t,p} - Q_{t-1,p}| \quad t=0,..,T\text{-}1;\ p=0,..,plant\text{-}1 \tag{19}$$

$$q^{max}_canal_p \geq Q_{t,p} + Q_spill_plant_{t,p} \quad t=0,..,T\text{-}1;\ p=0,..,plant\text{-}1 \tag{20}$$

$$V_{T,r} \in \langle \hat{V}_r^{min}, \hat{V}_r^{max} \rangle \quad t=0,..,T\text{-}1;\ r=0,..,res\text{-}1 \tag{21}$$

$$Q_{t,p} \in \{0\} \cup \langle Q_p^{min}, Q_p^{max} \rangle \quad t=0,..,T\text{-}1;\ p=0,..,plant\text{-}1 \tag{22}$$

$$P_{t,p} \in \{0\} \cup \langle P_p^{min}, P_p^{max} \rangle \quad t=0,..,T\text{-}1;\ p=0,..,plant\text{-}1 \tag{23}$$

$$V_{t,r} \in \langle V_r^{min}, V_r^{max} \rangle \quad t=0,..,T\text{-}1;\ r=0,..,res\text{-}1 \tag{24}$$

Kvôli veľkému počtu premenných a parametrov sú názvy všetkých premenných a parametrov uvedené v Tabuľkách 1 až 3.

Rovnica (3) popisuje zmenu objemu danej nádrže medzi hodinami. V ohraničení (2) vidíme, že funkcia výkonu môže byť vyjadrená dvoma spôsobmi. Buď ako funkcia závislá od prietoku Q a spádu H, alebo ako funkcia závislá od prietoku Q, objemu hornej nádrže HH_V a objemu dolnej nádrže DH_V. Zároveň si môžeme všimnúť, že každá z elektrární má špecifickú funkciu výkonu. Tento jav vychádza z technických špecifikácií jednotlivých elektrární.

Rovnice (4) a (5) popisujú prevod objemu nádrže na výšku hladiny. Podmienky (7) a (8) vyjadrujú, či je daný generátor v daný čas zapnutý. Ohraničenia (9) a (10) zabezpečujú, aby nebolo možné v čase t zapnúť generátor, ktorý už je zapnutý, resp. vypnúť generátor, ktorý je už vypnutý. Ďalšie ohraničenia (11) a (12) zabezpečujú, aby prietok cez jednotlivé zapnuté generátory elektrárne pokryl celkový prietok cez elektráreň. Ohraničenie (13) popisuje celkový odtok z nádrže, teda súčet prietoku cez elektráreň $Q_{t,p}$ a jalového prietoku cez elektráreň $Q_spill_plant_{t,p}$. Ohraničenie (14) zavádza pomocnú premennú $Q^{out}_del_{t,r}$, ktorá reprezentuje súčet toku do kanálu a oneskoreného jalového odtoku. Množina SK_w_delay pozostáva z indexov nádrží, ktoré majú staré koryto, pri ktorých nastáva oneskorenie jalového odtoku $Q_spill_{t,r}$. Funkcia F_3 je nelineárnou funkciou dotoku oneskoreného jalového odtoku. Ohraničenie (15) vyjadruje sumárny prítok do nádrže. Množina *canal_scheme* obsahuje indexy kanálových elektrární, medzi ktorými sa nemá kde zachytiť voda. Pri takýchto typoch elektrární musí byť objem vody, ktorý pritečie do elektrárne, rovnaký ako objem prietoku cez elektráreň. Tento vzťah zachytáva podmienka (16). Ohraničenia (17) a (18) zabezpečujú, aby boli dodržané maximálne a minimálne priemerné prietoky cez elektráreň. Nerovnosť (19) vyjadruje maximálnu možnú zmenu v prietoku medzi dvoma za sebou idúcimi hodinami. Nerovnosť (20) stanovuje

maximálny prietok kanálom. Podmienka (21) nám udáva ohraničenie na koncový stav objemu nádrže. Nakoniec ohraničenia (22)–(24) hovoria o rozsahu prípustných hodnôt pre dané premenné.

Bližšie vysvetlenie jednotlivých vzťahov a premenných v modeli možno nájsť v [1].

1.1.1 Funkcia výkonu v priestore spádu a prietoku

Kľúčovou časťou modelu je spôsob vyjadrenia funkcie výkonu. Ako vidíme vo vzťahu (2), v modeli sa vyskytujú dva rôzne spôsoby vyjadrenia výkonu. V tejto časti reprezentujeme funkciu výkonu v závislosti od dvoch premenných prietoku cez elektráreň a spádu, tj.

$$P_{t,p}(Q_{t,p}, H_{t,p}) = F_1^p(Q_{t,p}, H_{t,p}) \quad t = 0, 1, \ldots, T-1; p = 0, 1, \ldots, plant - 1, \tag{25}$$

kde $Q_{t,p}$ je prietok elektrárne a $H_{t,p}$ je čistý spád elektrárne p v čase t. Konkrétny tvar funkcie F_1^p uvedieme v Kapitole 3. Pod pojmom spád rozumieme kladný rozdiel medzi výškou hornej hladiny $HH_{t,p}$ a výškou dolnej hladiny $DH_{t,p}$, tj.

$$H_{t,p} = HH_{t,p} - DH_{t,p}$$

Výšky horných a dolných hladín možno vyjadriť v závislosti od objemu vody v daných nádržiach nasledovne

$$HH_{t,p} = F_5^p(V_{t,r}) \quad t = 0, 1, \ldots, T-1;\ p = 0, \ldots, plant - 1;\ r = 0, 1, \ldots, res - 1$$

$$DH_{t,p} = F_6^p(Q_{t,p}, HH_{t,p+1}) \quad t = 0, 1, \ldots, T-1;\ p = 0, 1, \ldots, plant - 1$$

Funkcie F_5^p a F_6^p sú nelineárnymi funkciami, ktoré prevádzajú objem nádrže na výšku hladiny. Všimnime si, že funkcie F_5^p a F_6^p sú špecifické pre jednotlivé elektrárne. Tento jav vyplýva z technických špecifikácií jednotlivých elektrární.

Kvôli technickým špecifikáciám jednotlivých elektrární na Vážskej kaskáde je potrebné pri modelovaní funkcie výkonu v závislosti od spádu rozlišovať elektrárne na základe ich polohy k prislúchajúcej vodnej nádrži. Pre účely tejto práce prevezmeme klasifikáciu predstavenú v práci [1]:

Názov parametra	Rozmer	Stručný popis
T	1×1	časový úsek, na ktorom optimalizujeme (24h)
$plant$	1×1	počet elektrární (22)
res	1×1	počet nádrží (12)
gen	1×1	počet generátorov (52)
z	1×1	prevod zo sekúnd na hodiny (3600)
$spill_pen$	$T \times res$	pokuta za jalový prietok a odtok
$outflows_bio$	$T \times res$	nutné biologické odtoky
$tg_startup_cost$	$gen \times 1$	cena zapnutia generátora
min_TG_uptime	$gen \times 1$	minimálny čas behu generátora
$firstTG$	$plant \times 1$	poradové čísla prvých generátorov jedn. elektrární
$lastTG$	$plant \times 1$	poradové čísla posledných generátorov jedn. elektrární
$infl$	$T \times res$	prítoky z vedľajších tokov, dažďa
min_avg_out	$res \times 1$	minimálne priemerné odtoky
max_avg_out	$res \times 1$	maximálne priemerné odtoky
q^{max}_canal	$p \times 1$	maximálny tok kanálom
q^{diff}	$plant \times 1$	maximálna zmena prietoku cez elektráreň
v_0	$res \times 1$	počiatočný objem v nádrži
$\hat{V}^{max}, \hat{V}^{min}$	$res \times 1$	maximálny a minimálny koncový objem
Q^{max}, Q^{min}	$T \times plant$	maximálny a minimálny prietok elektrárňou
P^{max}, P^{min}	$T \times plant$	maximálny a minimálny výkon elektrárne
V^{max}, V^{min}	$T \times res$	maximálny a minimálny objem nádrže
H^{max}, H^{min}	$T \times plant$	maximálny a minimálny spád
qtg^{max}, qtg^{min}	$T \times gen$	maximálny a minimálny prietok generátorom
D	$T \times 1$	požadovaný výkon, hodnota diagramu

Tabuľka 1: Vstupné parametre modelu

Názov stavovej premennej	Rozmer	Stručný popis
P	$T \times plant$	výkon
H	$T \times plant$	spád
V	$T \times res$	objem nádrže
Q^{out}	$T \times res$	sumárny odtok z nádrže do kanála
Q^{out}_del	$T \times res$	súčet odtoku z nádrže do kanála s oneskoreným jalovým odtokom
Q^{in}	$T \times res$	sumárny prítok do nádrže
HH	$T \times plant$	výška hornej hladiny
DH	$T \times plant$	výška dolnej hladiny
HH_V	$T \times plant$	objem hornej nádrže
DH_V	$T \times plant$	objem dolnej nádrže
TG	$T \times gen$	stav generátora (1-zapnutý, 0-vypnutý)

Tabuľka 2: Stavové premenné

Názov riadiacej premennej	Rozmer	Stručný popis
Q	$T \times plant$	prietok
Q_spill_plant	$T \times plant$	jalový prietok
Q_spill	$T \times res$	jalový odtok
TG_starup	$T \times gen$	zapnutie generátora
$TG_shutdown$	$T \times gen$	vypnutie generátora

Tabuľka 3: Riadiace premenné

- **Typ 0**: Elektrárne, ktoré sa nachádzajú medzi dvoma nádržami. V tomto prípade výška hornej nádrže závisí od objemu hornej nádrže a výška dolnej hladiny závisí od výšky hornej hladiny nasledujúcej nádrže v kaskáde a od prietoku cez elektráreň. Horná hladina bola aproximovaná lineárnou po častiach spojitou funkciou a dolná hladina pomocou lineárnej regresie v tvare

$$DH_{t,p} = a_p Q_{t,p} + b_p HH_{t,p} + c_p$$

.

- **Typ 1**: Elektrárne, ktoré sú hneď pod nádržou, ale za nimi sa nenachádza hneď ďalšia nádrž. V takomto prípade výška hornej nádrže závisí od objemu hornej nádrže a výška dolnej hladiny iba od prietoku cez elektráreň. Výšku hornej hladiny aproximujeme lineárnou po častiach spojitou funkciou a výšku dolnej hladiny lineárnou regresiou v tvare

$$DH_{t,p} = a_p Q_{t,p} + c_p$$

.

- **Typ 2**: Kanálové elektrárne. V tomto prípade sa nádrž nenachádza ani pod, ani nad elektrárňou. Výkon elektrárne potom závisí iba od prietoku.

- **Typ 3**: Elektrárne, ktoré sú hneď nad nádržou, ale pred nimi sa hneď nenachádza nádrž. Výška hornej hladiny sa potom nemení. Výška dolnej hladiny závisí od výšky hornej hladiny následujúcej nádrže a od prietoku cez elektráreň. Výšku dolnej hladiny aproximujeme rovnakým spôsobom ako v prípade Elektrárne typu 0.

- **Typ 4**: Elektrárne, ktoré sú hneď pod nádržou a na začiatku sústavy kanálových elektrární. Výška dolnej hladiny je považovaná za fixnú a výška hornej hladiny závisí od objemu hornej nádrže.

Poznamenajme, že koeficienty a_p, b_p a c_p boli odhadnuté z reálnych dát.

1.1.2 Funkcia výkonu v priestore prietoku a objemu hornej a dolnej nádrže

Druhou možnosťou je popísať výkon ako funkciu závislú od prietoku $Q_{t,p}$, objemu hornej hladiny $HH_V_{t,p}$ a objemu dolnej hladiny $DH_V_{t,p}$. Podobne ako v časti 1.1.1 presný

tvar funkcie výkonu závisí od individuálnych charakteristík elektrárne. Opäť rozlíšime nasledujúcich päť typov elektrární:

- **Typ 0**: Elektrárne, ktoré sú priamo pod nádržou, ale za nimi sa nenachádza hneď ďalšia nádrž. V takomto prípade do výpočtu výkonu elektrárne bude vstupovať okrem prietoku aj objem hornej a dolnej hladiny. Aproximovaný výkon je potom závislý od $Q_{t,p}$ a $(HH_V_{t,p} - DH_V_{t,p})$

- **Typ 1 a 4**: Elektrárne, ktoré sú priamo pod nádržou, ale hneď za nimi sa nenachádza ďalšia nádrž. To znamená, že máme k dispozícií iba informáciu o objeme hornej nádrže a výkon budeme modelovať len ako funkciu prietoku $Q_{t,p}$ a objemu hornej hladiny $HH_V_{t,p}$.

- **Typ 2**: Kanálové elektrárne. Výkon bude závislý iba od prietoku.

- **Typ 3**: Elektrárne, ktoré sú priamo nad nádržou, ale pred nimi sa hneď nenachádza nádrž. Informácia o objeme hornej hladiny nie je teda dostupná a výkon budeme modelovať len v závislosti od prietoku $Q_{t,p}$ a objemu dolnej hladiny $DH_V_{t,p}$.

2 Metódy riešenia

Model sformulovaný v Kapitole 1 možno zaradiť medzi problémy deterministického dynamického programovania. Úlohy daného typu sa štandardne riešia s využitím rovnice dynamického programovania (viď. [6]). Avšak pre úlohy s väčším počtom riadiacich a stavových premenných je tento prístup značne neefektívny. Dokumentujú to aj výsledky práce [3]. V tejto diplomovej práci sa autor pokúsil optimalizovať prevádzku jednej vodnej elektrárne štandardným numerickým prístupom vychádzajúci z RDP. Výsledky numerickej analýzy ukázali, že tento prístup je časovo náročný. Požiadavku z praxe, tj. získať riešenie v krátkom časovom horizonte, nebolo možné dosiahnuť už pri rozmere úlohy, ktorá sa týkala len jednej elektrárne.

Ako sme videli v Kapitole 1, rozmer úlohy optimálneho riadenia je pomerne veľký. Presnejšie ide o model, ktorý pozostáva z 24 časových periód. V každej časovej perióde je potrebné stanoviť hodnotu 56 spojitých a 106 binárnych riadiacich premenných. Súbežne s riadiacimi premennými v modeli vystupuje 233 stavových premenných.

Je zrejmé, že vzhľadom na takýto rozmer úlohy by nebolo efektívne problém optimálneho riadenia Vážskej kaskády riešiť štandardnými numerickými technikami, ktoré sú dostupné na riešenie úloh optimálneho riadenia. Okrem pamäťovej náročnosti týchto techník by totiž problémom bola aj dĺžka výpočtového času: v praxi je potrebné realizovať výpočet mnohokrát denne, aby sa výroba prispôsobila zmenám v vstupných parametroch. Slovenské elektrárne požadujú, aby výpočet optimálne nepresiahol jednu minútu.

V tejto Kapitole si predstavíme niekoľko alternatívnych spôsobov, ako riešiť problém popísaný v Kapitole 1. Namiesto riešenia dynamickej úlohy pomocou RDP sa na problém pozrieme ako na úlohu nelineárneho programovania, tj. pokúsime sa stanoviť hodnotu všetkých riadiacich premenných naraz. Prirodzene, nakoľko celý výpočet prebehne pre všetky riadiace premenné naraz, rozmer úlohy narástol 24−krát oproti jednej čiastkovej úlohe pri RDP. Podobne narástol aj počet ohraničení, ktoré treba naraz zohľadniť. Výhodou spomínaného prístupu je však väčšia množina dostupných optimalizačných techník a solverov, z ktorých viaceré sú v súčasnosti prispôsobené aj na riešenie úloh veľkého rozmeru. Keďže ide o deterministickú úlohu, navrhnutý pristúp je ekvivalentný riešeniu pôvodnej dynamickej úlohy.

V práci využijeme dve triedy výpočtových metód. Prvým prístupom je využitie do-

stupných solverov, ktoré sú uspôsobené na riešenie úloh lineárneho, resp. kvadratického programovania. Nakoľko však v probléme optimálnej prevádzky Vážskej kaskády vystupuje niekoľko nelineárnych vzťahov (viď (2), (14), (15)), pred použitím týchto solverov bude potrebné nájsť vhodnú aproximáciu daných vzťahov.

Druhú triedu výpočtových metód predstavujú heuristické prístupy. V tomto prípade sa na problém z Kapitoly 1 stále budeme pozerať ako na úlohu nelineárneho programovania. Keďže heuristicke prístupy majú všeobecné použitie, problém aproximácie nelineárnych vzťahov nie je v tomto prípade potrebné riešiť. Jedinou vynútenou zmenou bude modifikácia účelovej funkcie. V nasledujúcich častiach si jednotlivé techniky stručne priblížime.

2.1 Lineárna a kvadratická aproximácia

Problém optimálneho riadenia Vážskej kaskády popísaný v Kapitole 1 môžeme preformulovať do tvaru úlohy zmiešaného celočíselného programovania (MIP). Presnejšie, po aproximácii nelineárnych vzťahov dospejeme k úlohe zmiešaného lineárneho (resp. kvadratického) celočíselného programovania (MILP, resp. MIQCP). Výhodou MILP a MIQCP formulácií je možnosť využitia pokročilých solverov, ktoré sú schopné veľmi efektívne riešiť dané typy úloh. V MILP formulácii sa okrem spojitých premenných nachádzajú aj premenné, ktoré sú definované iba na podmnožine celých čísel. Štandardná formulácia MILP (viď. [7]) je v tvare

$$\begin{aligned} \min\ & c^T x + d^T y \\ & Ax + By \leq b \\ & x \in X \cap \mathbb{Z} \\ & y \in Y. \end{aligned} \tag{26}$$

V úlohe (26) symboly A a B reprezentujú matice koeficientov, pričom $A \in \mathbf{R}^{m \times n}$ a $B \in \mathbf{R}^{m \times s}$. Ďalej b je m dimenzionálny vektor, c je n dimenzionálny vektor a d je s dimenzionálny vektor.

Rozšírením úlohy zmiešaného celočíselného lineárneho programovania je úloha zmiešaného celočíselného kvadratického programovania (MIQCP), ktorého formuláciu možno vyjadriť v tvare

$$\min_x x^T Q x + c^T x$$

$$
\begin{aligned}
Ax &= b \\
x^T W x + w^T x &\leq 0 \\
l \leq x &\leq u \\
x_i \in \mathbb{Z} \text{ pre i} &\in I,
\end{aligned}
$$

kde I je množina indexov celočíselných zložiek vektora x. Matice $Q \in \mathbf{R}^{n \times n}$ a $W \in \mathbf{R}^{n \times n}$ sú kladne semidefinitné symetrické matice a vektory $c, w, l, u \in \mathbf{R}^n$.

Nevýhodou prístupu pri riešení úlohy optimálnej prevádzky Vážskej kaskády je potreba aproximácie nelineárnych funkcií. Kľúčovou funkciou je nelineárna dvojrozmerná funkcia výkonu (2). V nasledujúcej časti si popíšeme jeden spôsob jej linearizácie, tzv. trojuholníkovú metódu. Daná metóda bola použitá aj v práci [1]. Spomedzi testovaných alternatívnych spôsobov sa pri jej použití dosiahol najlepší pomer presnosti a rýchlosti výpočtu. Tieto výsledky nás motivovali k použitiu trojuholníkovej aproximácie aj pri riešení alternatívnej formulácie problému optimálnej prevádzky Vážskej kaskády predstavenej v Kapitole 1.

2.1.1 Trojuholníková metóda

Kľúčovým problémom pri aplikácii MILP je spôsob, akým linearizujeme nelineárne vzťahy v danom probléme. V tejto časti si predstavíme trojuholníkovú metódu, ktorá bola použitá aj v práci [1] a jej podrobnú špecifikáciu možno nájsť v práci [8].

Ide o dvojrozmernú po častiach lineárnu aproximáciu v dvojrozmernom priestore. Uvažujme body $x_1, \ldots, x_n$ a $y_1, \ldots, y_m$ usporiadané tak, že $x_1 \leq x_2 \leq \ldots \leq x_n$ a $y_1 \leq y_2 \leq \ldots \leq y_m$. Majme ďalej funkciu $f(x, y)$, ktorej hodnoty poznáme vo všetkých bodoch (x_i, y_j) pre $i = 1, \ldots, n$ a $j = 1, \ldots, m$. Potom pre ľubovoľný bod $\hat{x} \in \langle x_1, x_n \rangle$ vieme nájsť taký index i, že platí $x_i \leq \hat{x} \leq x_{i+1}$. Podobne pre ľubovoľný bod $\hat{y} \in \langle y_1, y_m \rangle$ vieme nájsť taký index j, že platí $y_j \leq \hat{y} \leq y_{j+1}$. Bod $(\hat{x}, \hat{y})$ sa potom nachádza v obdĺžniku danom bodmi (x_i, y_j), (x_{+1}, y_j), (x_i, y_{j+1}) (x_{i+1}, y_{j+1}). Hodnotu $f(\hat{x}, \hat{y})$ následne aproximujeme ako konvexnú kombináciu troch hodnôt. Dve z týchto hodnôt budú $f(x_i, y_j)$ a $f(x_{i+1}, y_{j+1})$. Tretia hodnota bude buď $f(x_i, y_{j+1})$, alebo $f(x_{i+1}, y_j)$ podľa toho, či sa bod $(\hat{x}, \hat{y})$ nachádza nad alebo pod uhlopriečkou obdĺžnika. Aproximovanú hodnotu funkcie f

v bode $(\hat{x}, \hat{y})$ potom zapíšeme ako

$$f^a(\hat{x}, \hat{y}) = \lambda f(x_i, y_j) + \mu f(x_{i+1}, y_{j+1}) + (1 - \lambda - \mu)\overline{f}, \tag{27}$$

kde $\lambda \in [0, 1]$, $\mu \in [0, 1]$, $(1 - \lambda - \mu) \in [0, 1]$, $\overline{f}$ je funkčná hodnota v treťom vrchole obdĺžnika a hodnota $\overline{f}$, ktorú definujeme nasledovne

$$\overline{f} = \begin{cases} f(x_{i+1}, y_j) & \text{ak } \hat{y} \leq y_j + (\hat{x} - x_i)\frac{y_{j+1} - y_j}{x_{i+1} + x_i} \\ f(x_i, y_{j+1}) & \text{inak.} \end{cases} \tag{28}$$

V ďalšom kroku je potrebné takto naformulovanú aproximáciu v tvare po častiach lineárnej funkcie zakomponovať do MILP formulácie. To znamená, že je potrebné ju zapísať pomocou lineárnych ohraničení. Pri tejto formulácii sa inšpirujeme postupom uvedeným v práci [8]. Zavedieme $n \times m$ nových pomocných spojitých premenných $\alpha_{i,j} \in [0, 1]$, pre $i = 1, \ldots, n$ a $j = 1, \ldots, m$. Po častiach spojitú lineárnu funkciu (27) zapíšme pomocou lineárnych ohraničení nasledovne

$$1 = \sum_{i=1}^{n} \sum_{j=1}^{m} \alpha_{i,j} \tag{29}$$

$$\hat{x} = \sum_{i=1}^{n} \sum_{j=1}^{m} \alpha_{i,j} x_i \tag{30}$$

$$\hat{y} = \sum_{i=1}^{n} \sum_{j=1}^{m} \alpha_{i,j} y_j \tag{31}$$

$$f^a(\hat{x}, \hat{y}) = \sum_{i=1}^{n} \sum_{j=1}^{m} \alpha_{i,j} f(x_i, y_j) \tag{32}$$

Formulácia ohraničení (29)–(32) ešte nie je dostačujúca. Potrebujeme totiž zabezpečiť nenulovosť práve troch premenných α_{ij} vymedzujúcich trojuholník, v ktorom je obsiahnutý bod $(\hat{x}, \hat{y})$. To dosiahneme zavedením ďalších pomocných binárnych premenných $h^u_{i,j}$ a $h^l_{i,j}$, ktoré udávajú, či sa bod $(\hat{x}, \hat{y})$ nachádza v hornom alebo dolnom trojuholníku. Pomocné ohraničenia zapíšeme v nasledujúcom tvare

$$1 = \sum_{i=1}^{n-1} \sum_{j=1}^{m-1} (h^u_{i,j} + h^l_{i,j}) \tag{33}$$

$$\begin{gathered} \alpha_{i,j} \leq h^u_{i,j} + h^l_{i,j} + h^u_{i,j-1} + h^l_{i-1,j-1} + h^u_{i-1,j-1} + h^l_{i-1,j} \\ i = 1, \ldots, n; j = 1, \ldots, m \end{gathered} \tag{34}$$

$$\begin{gathered} h^u_{i,0} = h^l_{i,0} = h^u_{n,j} = h^l_{n,j} = h^u_{i,m} = h^l_{i,m} = 0 \\ i = 1, \ldots, n; j = 1, \ldots, m \end{gathered} \tag{35}$$

Ohraničenie (33) hovorí o tom, že spomedzi všetkých trojuholníkov sa iba jeden využíva na konvexnú kombináciu bodov. Ohraničenie (34) zaručuje, že jediné nenulové hodnoty premenných $\alpha_{i,j}$ budú pre tie premenné, ktoré sú asociované s daným trojuholníkom. Napríklad, ak by bola premenná $h^u_{i,j} = 1$, znamená to, že sa nachádzame v trojuholníku danom bodmi (x_i, y_j),(x_{i+1}, y_j) a (x_{i+1}, y_{j+1}), keďže jediné nenulové hodnoty budú $\alpha_{i,j}$, $\alpha_{i+1,j}$ a $\alpha_{i+1,j+1}$. Posledné ohraničenie (35) definuje krajné hodnoty premenných $h^u_{i,j}$ a $h^l_{i,j}$. Prejdením k takejto formulácii v modeli celkovo pribudne $3 \times m \times n \times plant$ premenných, keďže každá elektráreň v kaskáde má svoju vlastnú funkciu výkonu, ktorú je potrebné aproximovať osobitne.

2.1.2 MILP formulácia modelu Vážskej kaskády

V časti 2.1.1 časti sme prezentovali spôsob, ako vo všeobecnosti pomocou trojuholníkovej metódy možno aproximovať nelineárnu dvojrozmernú funkciu. Pomocou tohto postupu možno prepísať rovnicu výkonu F^p_1, resp. F^p_2 (viď (2)) v modeli Vážskej kaskády. Ďalšími nelineárnymi vzťahmi v modeli sú funkcie dotokových časov F_3, resp. F_4 v rovnici (14), resp. (15). Nakoľko ide o funkcie jednej premennej, môžme ich jednoducho aproximovať pomocou jednorozmernej po častiach lineárnej funkcie.

Posledným problémom, ktorý je potrebné vyriešiť, je formulácia podmienok (7)–(10), (22) a (23). Tvar týchto podmienok nie je v súlade s MILP formuláciou. Podmienky (7)–(10) hovoria o binárnych premenných, určujúcich či je daný generátor zapnutý. Podmienky (22) a (23) vymedzujú prípustné hodnoty pre premenné $Q_{t,p}$ a $P_{t,p}$. Ohraničenia (7)–(10) môžeme ekvivalentne zapísať ako

$$TG_startup_{t,g} \leq TG_{t,g} \quad t=0,\ldots,T\text{-}1,\ g=0,\ldots,gen\text{-}1 \tag{36}$$

$$TG_shutdown_{t,g} \leq 1 - TG_{t,g} \quad t=0,\ldots,T\text{-}1,\ g=0,\ldots,gen\text{-}1 \tag{37}$$

$$TG_{t,g} - TG_{t-1,g} = TG_startup_{t,g} - TG_shutdown_{t,g} \quad t=1,\ldots,T\text{-}1,\ g=0,\ldots,gen\text{-}1 \tag{38}$$

$$TG_{0,g} = TG_startup_{0,g} \quad g=0,\ldots,gen\text{-}1 \tag{39}$$

$$TG_{t,g} \geq \sum_{i=t-min_TG_uptime_g+1}^{t} TG_startup_{i,g} \quad g=0,\ldots,gen\text{-}1,\ t = min_TG_uptime_g\ \text{-}1,\ldots,T\text{-}1 \tag{40}$$

Ohraničenie (36) hovorí o tom, že ak je hodnota premennej $TG_startup_{t,g}$ nastavená na hodnotu jedna (tj. zapíname generátor), tak aj premenná $TG_{t,g}$ musí byť rovná jednej (teda generátor je zapnutý). Podobne ohraničenie (37) hovorí o tom, že ak bude premenná $TG_shutdown_{t,g} = 1$, tak generátor bude v čase t vypnutý, teda $TG_{t,g} = 0$.

Ohraničenie (38) zabezpečuje správne zapínanie a vypínanie generátorov. Ak by bol generátor g zapnutý v časoch t a $t-1$ ($TG_{t,g} = TG_{t-1,g} = 0$), tak toto ohraničenie spolu s ohraničením (37) stanovuje, že v čase t generátor nezapíname a ani nevypíname, tj. $TG_startup_{t,g} = TG_shutdown_{t,g} = 0$. V prípade, že generátor g je zapnutý v čase t, ale v čase $t-1$ bol vypnutý, tak nutne musí platiť, že $TG_startup_{t,g} = 1$ a $TG_shutdown_{t,g} = 0$. Naopak, ak je generátor g v čase t vypnutý a v čase $t-1$ bol zapnutý, tak vďaka ohraničeniu (38) bude nutne $TG_startup_{t,g} = 0$ a $TG_shutdown_{t,g} = 1$. Poslednou možnosťou je, že generátor g je vypnutý v časoch t a $t-1$ ($TG_{t,g} = TG_{t-1,g} = 1$). Vtedy z ohraničení (38) a (36) vyplýva, že $TG_startup_{t,g} = TG_shutdown_{t,g} = 0$.

Vzťah (40) zabezpečuje, že po spustení generátora musí daný generátor ostať zapnutý aspoň čas $min_TG_uptime_g$. Ak by bol generátor spustený v čase s, tak ohraničenie (40) zabezpečuje, že generátor bude zapnutý aspoň do času $s + min_TG_uptime_g - 1$. Ak by totiž to bol generátor vypnutý skôr, tak vzťah (40) by nebol splnený pre $t = s + min_TG_uptime_g - 1$. Ohraničenie (39) nastavuje počiatočný stav premennej $TG_{0,g}$

Poslednými ohraničeniami, ktoré treba nahradiť lineárnymi vzťahmi, sú ohraničenia (22) a (23). Tie môžeme zapísať ekvivalentne ako

$$Q_{t,p} \leq Bplant_{t,p} Q_p^{max} \quad t=0,\ldots,T\text{-}1,\ p=0,..,plant\text{-}1 \tag{41}$$

$$Q_{t,p} \geq Bplant_{t,p} Q_p^{min} \quad t=0,\ldots,T\text{-}1,\ p=0,\ldots,plant\text{-}1 \tag{42}$$

$$Q_{t,p} \in \langle 0, Q_p^{max} \rangle \quad t=0,\ldots,T\text{-}1,\ p=0,\ldots,plant\text{-}1 \tag{43}$$

$$P_{t,p} \leq Bplant_{t,p} P_p^{max} \quad t=0,\ldots,T\text{-}1,\ p=0,..,plant\text{-}1 \tag{44}$$

$$P_{t,p} \geq Bplant_{t,p} P_p^{min} \quad t=0,\ldots,T\text{-}1,\ p=0,\ldots,plant\text{-}1 \tag{45}$$

$$P_{t,p} \in \langle 0, P_p^{max} \rangle \quad t=0,\ldots,T\text{-}1,\ p=0,\ldots,plant\text{-}1 \tag{46}$$

$$Bplant_{t,p} \in \{0,1\} \quad t=0,\ldots,T\text{-}1,\ p=0,\ldots,plant\text{-}1 \tag{47}$$

$$Bplant_{t,p} \leq \sum_{g=firstTG_p}^{lastTG_p} TG_{t,g} \quad t=0,\ldots,T\text{-}1,\ p=0,\ldots,plant\text{-}1 \tag{48}$$

$$Bplant_{t,p} \geq TG_{t,g} \quad t=0,\ldots,T\text{-}1,\ p=0,\ldots,plant\text{-}1,\ g = firstTG_p,\ldots,lastTG_p \tag{49}$$

Zaviedli sme novú binárnu premennú $Bplant_{t,p}$, ktorá vyjadruje, či je v čase t spustená elektráreň p. Ohraničenia (41)–(43) zabezpečujú, že ak je elektráreň p v čase t vypnutá ($Bplant_{t,p} = 0$), tak prietok cez elektráreň $Q_{t,p} = 0$. Naopak, ak je elektráreň p v čase t zapnutá ($Bplant_{t,p} = 1$), tak prietok cez elektráreň musí ležať v intervale $\langle Q_p^{min}, Q_p^{max} \rangle$.

Ohraničenia (44)–(46) rovnako zabezpečujú, že ak je elektráreň p v čase t vypnutá, tak výkon elektrárne $P_{t,p} = 0$ a naopak, ak je zapnutá, tak výkon elektrárne leží v intervale $\langle P_p^{min}, P_p^{max} \rangle$.

Ohraničenie (47) definuje premennú $Bplant_{t,p}$ ako binárnu premennú. Hodnota $firstTG_p$ (resp. $lastTG_p$) predstavuje poradové číslo prvého (resp. posledného) generátora elektrárne p. Vzťah (48) zabezpečuje, že ak sú všetky generátory prislúchajúce k elektrárni p v čase t vypnuté, tak nutne aj elektráreň p musí byť vypnutá ($Bplant_{t,p} = 0$). Podobne ohraničenie (49) hovorí o tom, že ak aspoň jeden z generátorov prislúchajúcich elektrárni p je zapnutý ($TG_{t,g} = 1$), tak aj elektráreň p je zapnutá ($Bplant_{t,p} = 1$).

2.1.3 Výpočtové techniky

Na riešenie MILP (resp. MIQCP) úloh prezentovaných v častiach 2.1.1 a 2.1.2 využijeme dostupné solvery. V centre našej pozornosti stoja tie softvérové balíky, ktoré sú uspôsobené na riešenie úloh veľkého rozmeru. V práci sme aplikovali tri známe solvery: Gurobi, Cplex a Knitro. V práci [14] boli porovnávané solvery Gurobi a Cplex. Na základe numerických experimentov, ktoré autori v práci [14] realizovali, nemožno spomedzi solverov vybrať jeden solver, ktorý by bol univerzálne presnejší alebo rýchlejší, naprieč spektrom rôznych úloh.

Všetky tri solvery používajú ako základ riešenia metódy *Branch-and-bound* a *Cutting-planes*. Zároveň treba podotknúť, že solver Cplex je navrhnutý len na riešenie MILP úloh, zatiaľ čo solvery Gurobi a Knitro sú schopné riešiť aj MIQCP úlohy.

Na nájdenie optimálneho riešenia uvedené solvery využívajú kombináciu viacerých techník. Cieľom je zvýšiť efektivitu výpočtu, tj. nájsť dostatočne presné riešenie v čo najkratšom čase. Rýchlosť výpočtu je možné zlepšiť aj pridaním ďalších doplnkových techník. Jednou z nich je tzv. preprocesing (predspracovanie). V nasledujúcich častiach si uvedené výpočtove techniky stručne priblížime.

Branch-and-bound

Algoritmus *Branch-and-bound* možno popísať nasledovne. Na začiatku máme MILP problém, ktorý nie sme schopný riešiť priamo, z dôvodu celočíselných ohraničení na niektoré premenné. Z ohraničení odstránime podmienky celočíselnosti a dostaneme problém lineárneho programovania, ktorý sa označuje ako relaxácia pôvodnej MILP úlohy. Takúto úlohu lineárneho programovania už vieme jednoducho riešiť. Ak riešenie relaxovanej úlohy bude spĺňať aj odstránené podmienky celočíselnosti, tak algoritmus našiel optimálne riešenie pôvodnej MILP úlohy. V prípade, ak riešenie nespĺňa túto podmienku, tak algoritmus zvolí jednu premennú v riešení, ktorá nespĺňa pôvodnú podmienku celočíselnosti.

Označme pôvodnú MILP úlohu ako P_0 a bez ujmy na všeobecnosti predpokladajme, že ide o minimalizačnú úlohu. Majme napríklad premennú $x \in \mathbb{Z}$, ktorá nadobudla v riešení relaxovanej úlohy hodnotu 9,1. Je zrejmé, že takéto riešenie nie je prípustným riešením pre úlohu P_0. Aby sme túto neprípustnú hodnotu vylúčili, rozvetvíme úlohu P_0 na dve podúlohy P_1 a P_2. Úloha P_1 rieši rovnaký problém, ako riešila pôvodná úloha P_0, ale s pridaným ohraničením $x \leq 9$. Úloha P_2 tiež rieši rovnaký problém ako P_0, ale s pridaním ohraničenia $x \geq 10$. Premennú x potom nazveme vetviacou (branching) premennou. Týmto spôsobom sme pôvodnú úlohu P_0 nahradili dvoma jednoduchšími (resp. lepšie ohraničenými) úlohami. Rovnaký postup relaxácie a prípadného vetvenia môžeme potom aplikovať na podúlohy P_1 a P_2. Vetvením podúloh bude postupne vznikať takzvaný prehľadávací strom. Jednotlivé MILP úlohy v tomto strome sa nazývajú uzly (nodes), pričom ak hodnota účelovej funkcie bola pre nové riešenie relaxovanej úlohy menšia ako pre doteraz najlepšie nájdene riešenie, algoritmus si zapamätá novú minimálnu hodnotu p_{min}.

Ak sa pri vetvení stromu dostaneme do uzla, v ktorom riešenie relaxovanej úlohy dáva hodnotu účelovej funkcie $p_k \geq p_{min}$, tak tento uzol ukončíme. Pridanie ďalších podmienok, tj. ďalšie vetvenie stromu, by totiž prirodzene viedlo k väčšej hodnote účelovej funkcie (minimalizácia cez menšiu množinu prípustných riešení), preto uzol ukončíme. Druhá možnosť ukončenia uzlu nastáva v prípade, ak pridaním ohraničenia sa úloha stala neprípustnou. Ak pri úlohe P_k platilo, že optimálna hodnota účelovej funkcie $p_k \leq p_{min}$, ale riešenie nespĺňa podmienku celočíselnosti, tak je potrebné uzol opäť vetviť.

Kľúčovým krokom v tomto algoritme je určenie, ktorá premenná bude vetviaca. Od tohto rozhodnutia závisí rýchlosť, s akou *Branch-and-bound* algoritmus nájde optimálne

riešenie. V dostupnej literatúre (viď napr. [15]) možno nájsť viacero pravidiel na stanovenie vhodnej vetviacej premennej. Moderné solvery používajú kombinácie mnohých pravidiel v závislosti od vlastností daného uzla ([9]).

Cutting-planes (rezné roviny)

Inou možnosťou, ako riešiť úlohy zmiešaného celočíselného programovania, je pridanie dodatočných ohraničení k pôvodnej úlohe. Cieľom pridaných ohraničení je zabezpečiť splnenie nasledujúcich dvoch podmienok. Prvou je, že pridaním danej podmienky nevylúčime (neodrežeme) žiadne prípustné riešenia pôvodného MILP problému. Druhou podmienkou je, že dané ohraničenie vylúči (odreže) aspoň jedno neceločíselné riešenie MILP, ktoré by mohlo byť potenciálnym riešením relaxovanej úlohy. Algoritmy, ktoré aplikujú takéto dodatočné ohraničenia sa nazývajú *cutting planes* algoritmy. *Cutting planes* algoritmy samé o sebe zväčša nie sú schopné vyriešiť MILP problémy v praxi, ale v kombinácii s *Branch-and-bound* algoritmom sú veľmi úspešné. Aplikovaním rezných rovín vnútri *Branch-and-bound* algoritmu často dochádza k zníženiu potrebných vetvení ([12]). Takýto kombinovaný prístup sa nazýva *Branch-and-cut*. Existuje veľa rôznych metód, ako pridať do formulácie úlohy metodiku rezných rovín. Popis jednotlivých spôsobov možno nájsť v [13].

Názorne si ukážeme danej metódy na jednoduchom príklade. Predpokladajme, že máme MILP úlohu s nasledujúcim ohraničením

$$6x_1 + 5x_2 + 6x_3 + 8x_4 + 7x_5 \leq 20 \tag{50}$$

$$x_1, x_2, x_3, x_4, x_5 \in \{0, 1\} \tag{51}$$

,

Následne prepokladajme, že sme vyriešili relaxovanú úlohu k pôvodnej úlohe a riešenie relaxácie vyšlo: $x_1 = 0$, $x_2 = 1, x_3 = x_4 = x_5 = \frac{4}{5}$. Vidíme však, že riešenie nie je prípustným riešením pôvodnej úlohy. Toto riešenie vieme vylúčiť pridaním jedného ohraničenia do pôvodnej úlohy. Ľahko overíme, že nie je možné, aby všetky tri neceločíselné zložky riešenia boli rovné jednej. Teda nemôže platiť $x_3 = x_4 = x_5 = 1$, keďže $6 + 8 + 7 = 21 > 20$. Potom pridaním nerovnosti $x_3 + x_4 + x_5 \leq 2$ vylúčime súčasné riešenie, keďže $\frac{4}{5} + \frac{4}{5} + \frac{4}{5} = \frac{12}{5} > 2$.

Viac podrobnosti k tejto metóde možno nájsť v [11].

Preprocessing (predspracovanie)

Pred začatím samotného algoritmu *Branch-and-bound* (resp. *Branch-and-cut*) väčšina solverov vsúva ešte krok preprocessingu. Úlohou tohto kroku je odstrániť zbytočné informácie z užívateľovej formulácie problému a zároveň vylepšiť ohraničenia modelu skrz logické implikácie. Preprocessing dokáže byť veľmi efektívny a často bez neho nie je možné niektoré problémy ani riešiť klasickými metódami.

Typicky sa preprocessing aplikuje väčšinou na začiatku optimalizácie, avšak je možné ho použiť aj vo vnútorných uzloch v *Branch-and-cut* strome. Dôležité je aj rozdeliť dostupný čas na optimalizáciu medzi preprocessing a samotné hľadanie optimálneho riešenia. Žiaľ pre tento problém neexistuje všeobecné riešenie, koľko času sa oplatí investovať do preprocessingu. Zväčša sa však odporúča pre zložité modely na začiatku investovať dlhší čas do preprocessingu. Bližšie informácie k preprocessingu a popis konkrétnych metód, ktoré využíva preprocessing, možno nájsť v [16].

2.2 Heuristiky

Heuristické metódy optimalizácie predstavujú významnú oblasť výskumu v oblasti

umelej inteligencie a výpočtovej matematiky. Tieto metódy sú zvlášť cenné v situáciách, kde klasické analytické prístupy zlyhávajú alebo sú neefektívne. Napríklad pri riešení veľmi komplexných nelineárnych problémov optimalizácie.

Pod pojmom heuristika rozumieme prístup, ktorý je založený na prehľadávaní priestoru riešení. Cieľom je efektívne nájsť dostatočne dobré riešenia pre optimalizačný problém s prijateľnými výpočtovým časom. Charakteristickým znakom heuristík je ich schopnosť uniknúť z lokálnych optím a priblížiť sa ku globálnemu optimu.

Úspech heuristických metód spočíva v ich univerzálnosti a schopnosti riešiť širokú škálu problémov bez potreby významných úprav. Tieto metódy sú inšpirované prírodnými procesmi a využívajú koncepty ako evolúcia, termodynamika a kolektívne správanie sociálnych populácií.

Heuristiky využívajú rôzne pravidlá, stratégie a funkcie na určenie, ktoré riešenia majú byť skúmané, a ktoré majú byť zanedbané. Týmto spôsobom sa snažia minimalizovať počet skúmaných riešení a zamerať sa na tie, ktoré majú väčšiu pravdepodobnosť byť blízko optimálneho riešenia. Narozdiel od exaktných výpočtových metód predstavených v časti

2.1, heuristiky nevyžadujú linearizáciu, čo možno považovať za ich výhodu. Na druhej strane, pri aplikácií heuristík na úlohy s veľkým počtom ohraničení je potrebné zaručiť, aby prehľadávací algoritmus ostal v množine prípustných riešení. To sa docieli modifikáciou účelovej funkcie pridaním penalizácií, ak dané riešenie nie je prípustné. Zároveň pri aplikácií heuristík nie je zaručené, že algoritmus nájde globálne optimum problému.

V práci sme sa pokúsili aplikovať dva typy heuristík na optimalizáciu Vážskej kaskády: genetické algoritmy a simulované žíhanie.

2.2.1 Genetické algoritmy

Genetické algoritmy (GA) predstavujú jednu z kľúčových tried heuristických metód. Inšpiráciu čerpajú v princípe prirodzenej selekcie, ktorú pozorujeme v biologických procesoch. Genetické algoritmy sú založené na koncepte evolúcie, kde najvhodnejšie jedince prežívajú a reprodukujú sa, čím postupne vytvárajú stále lepšie prispôsobené populácie. V kontexte optimalizácie „jedinci" predstavujú potenciálne riešenia daného problému a „vhodnosť" (fitness) jedinca je meraná pomocou účelovej funkcie, ktorá hodnotí kvalitu riešenia. Každé riešenie je zakódované ako reťazec numerických hodnôt, ktorý sa nazýva chromozóm. Jednotlivé zložky chromozómu sa označujú ako gény.

Genetické algoritmy môžu byť vo všeobecnosti charakterizované vzťahom

$$x(t+1) = v(s(x(t))), \tag{52}$$

kde $x(t)$ je populácia v čase t, $v(.)$ je operátor kríženia a mutácie (taktiež sa nazýva operátor variácie) a $s(.)$ je operátor výberu (selekcie) jedincov na kríženie. Každá množina jednotlivcov v ľubovoľnej iterácií sa nazýva generácia. Bližší popis možno nájsť v [17].

Vo všeobecnosti genetický algoritmus môžeme reprezentovať schémou **Algorithm 1** (pre viac informácií viď. [17]).

Proces výberu rodičov je založený v prospech jedincov, ktorí zodpovedajú vhodnejším riešeniam. Dobré riešenia môžu byť vybrané aj viackrát. To zodpovedá myšlienke prežitia najschopnejších jedincov. Procesy mutácie a kríženia vnášajú malé náhodné zmeny do genetického materiálu jedincov, čím podporujú diverzitu populácie a pomáhajú algoritmu uniknúť z lokálnych optím [17]. Mutácia podporuje diverzitu populácie vnášaním nových hodnôt génov do populácie, zatiaľ čo kríženie kombinuje časti existujúcich riešení, aby bolo vytvorené nové riešenie.

Algorithm 1 Genetický algoritmus

Vytvorenie počiatočnej populácie jedincov a vyčíslenie účelovej funkcie každého z n riešení (jedincov)

repeat

Vyber n členov zo súčasnej generácie riešení ako rodičov (selekcia).

repeat

Vyber dvoch rodičov z množiny rodičov a pomocou operátora kríženia z nich vytvor dve nové riešenia (potomkov).

S pravdepodobnosťou p_{mut} aplikuj operátor mutácie na každú zložku (gén) potomkov.

until Nebolo vytvorených n nových riešení (potomkov).

Nahraď starú populáciu novo vytvorenou populáciou potomkov.

until Až kým nie je splnená ukončovacia podmienka

Vo všeobecnosti existuje veľa rôznych metód, ako konkrétne realizovať operátory selekcie, kríženia a mutácie. Popis jednotlivých metód možno nájsť v [20].

2.2.2 Simulované žíhanie

Podobne ako biologické procesy tak aj fyzikálne procesy dokážu byť inšpiráciou optimalizačných metód. Jedným z takýchto algoritmov je simulované žíhanie (SA). Simulované žíhanie je metóda inšpirovaná procesom žíhania v metalurgii, kde sa materiál zohrieva na vysokú teplotu a postupne sa ochladzuje, aby sa zlepšili jeho vlastnosti, akými sú pevnosť a odolnosť. Táto metóda je založená na princípe termodynamiky, konkrétne na procesoch, ktoré sa vyskytujú pri žíhaní kovov. Cieľom je dosiahnuť stav s minimálnou vnútornou energiou. Podrobnejšie informácie o fyzikálnych princípoch týchto metód možno nájsť v [17].

V kontexte optimalizácie simulované žíhanie začína s náhodným alebo preddefinovaným počiatočným bodom v priestore riešení, ktorý sa náhodne mení v každej iterácii algoritmu. Modifikácie, ktoré zlepšia účelovú funkciu algoritmus vždy príjme. Avšak algoritmus s nenulovou pravdepodobnosťou príjme aj modifikácie, ktoré vedú k zhoršeniu hodnoty účelovej funkcie. Hodnota pravdepodobnosti závisí od „teploty" systému. Vysoká teplota znamená vyššiu pravdepodobnosť, že algoritmus príjme aj horšie riešenie.

Toto poskytuje algoritmu možnosť uniknúť z lokálneho optima.

Jadro spomínaného prístupu predstavuje definovanie štruktúry susednosti riešení pre daný problém, tj. určiť, ktoré riešenia sú blízke k danému riešeniu. Základný algoritmus SA možno schematicky zapísať v tvare **Algorithm 2**.

Algorithm 2 Simulované žíhanie

Zadanie počiatočnej teploty, počiatočného riešenia a vyčíslenie účelovej funkcie počiatočného riešenia

repeat

Náhodné vygenerovanie nového riešenia blízkeho terajšiemu riešeniu a následne vyčíslenie účelovej funkcie

if hodnota nového riešenia $<$ hodnota terajšieho riešenia **then**

Príjmi nové riešenie

else

Príjmi nové riešenie s pravdepodobnosťou p.

end if

Uprav teplotu

until až kým nie je splnená ukončovacia podmienka

Na začiatku algoritmu je teplota nastavená na vyššie hodnoty, čo umožňuje algoritmu nájsť širokú oblasť v priestore riešení, kde by malo existovať globálne optimum. S ďalšími iteráciami teplota postupne klesá. Pri nižších teplotách sa objavia jemnejšie detaily skúmanej oblasti riešení, čo umožňuje algoritmu nájsť globálne optimum alebo riešenie blízke globálnemu optimu ([17]).

Jadrom prehľadávacích algoritmov je spôsob, akým sa priestor riešení prechádza. Vo všeobecnosti spôsob, ktorým simulované žíhanie prehľadáva priestor možno popísať nasledovne. Definujme X ako množinu všetkých riešení. Pre každé riešenie $x \in X$ existuje množina všetkých riešení N_x, do ktorých sa algoritmus môže dostať v jednom kroku z riešenia x. Množinu N_x potom nazveme množinou susedov x. Požiadavkou je, aby táto relácia susednosti bola symetrická, teda $x \in N_y$ práve vtedy, keď $y \in N_x$. Výkonnosť SA algoritmu vo veľkej miere závisí na voľbe relácie susednosti, ktorú vo všeobecnosti špecifikuje používateľ na mieru pre daný problém. Ak sú množiny susedov príliš malé, tak algoritmus nebude schopný prechádzať množinu riešení X dostatočne rýchlo, aby našiel

optimum v rozumnom čase. Naopak, ak sú množiny susedov príliš veľké, rádovo o veľkosti X, tak algoritmus v podstate vykonáva náhodné hľadanie cez celú množinu riešení X ([17]). Obvykle sa susedia generujú náhodnou malou zmenou terajšieho riešenia.

Kľúčovou časťou SA algoritmu je výpočet pravdepodobnosti akceptácie nových riešení, ktoré vedú k horšej hodnote účelovej funkcie. Hodnota takejto pravdepodobnosti sa obvykle vypočíta ako

$$p_i = e^{-\Delta f / T_i}, \tag{53}$$

kde T_i je teplota v iterácií i, Δf je zmena hodnoty účelovej funkcie pri zhoršení (rozdiel je kladný, ak došlo k zhoršeniu). Podľa vzťahu (53) akceptácia horšieho riešenia závisí iba od teploty systému a od toho, k akému veľkému zhoršeniu došlo. Teplota T_i bude postupne klesať podľa takzvaného plánu ochladzovania ([17]). Čím pomalší je proces ochladzovania, tým je väčšia pravdepodobnosť že algoritmus nájde optimálne riešenie. Na druhú stranu to ale predĺži výpočtový čas algoritmu. Najčastejšie je používané exponenciálne ochladzovanie, tj. $T_{i+1} = \alpha T_i$, kde $\alpha \in (0, 1)$. V literatúre ([17]) sa odporúča nastaviť hodnotu $\alpha = 0.9$ a počiatočnú teplotu T_1 na takú hodnotu, aby pravdepodobnosť akceptácie horších riešení bola na začiatku algoritmu približne 80%.

3 Priebežné porovnanie optimalizačných metód

V tejto Kapitole zhrnieme výsledky numerickej analýzy. Jednotlivé optimalizačné techniky predstavené v Kapitole 2 porovnáme na jednom zvolenom obchodovacom dni. Porovnávacím kritériom bude hodnota odchýlky výroby od diagramu a výpočtový čas. Požiadavka na rýchlosť výpočtu vzišla z praxe. Nakoľko riadenie Vážskej kaskády je nutné prispôsobovať náhle sa meniacim podmienkam v rámci dňa, je žiaduce, aby riešenie bolo nájdené do jednej minúty.

Všetky metódy predstavené v tejto práci sme implementovali pomocou programovacieho jazyka Python. V časti 2.1 sme predstavili optimalizačné metódy, pri ktorých bolo potrebné aproximovať funkciu výkonu. Presnosť týchto metód vyhodnotíme na základe porovnania aproximovaného výkonu s reálnym výkonom. Vzťah popisujúci reálny výkon generátora je uvedený v [10]. Ide o kubickú funkciu v tvare

$$P(Q,H) = (n - s(y-H))Q^2 + (o - u(y-H))Q + (r - w(y-H)), \tag{54}$$

kde P je výkon turbíny v MW. Premenná Q je prietok cez turbínu v m^3s^{-1}. Ďalej veličina H je čistý spád v metroch (rozdiel nadmorskej výšky hornej a dolnej hladiny) a n, s, y, o, u, r, w sú parametre popisujúce výkonový diagram turbíny. Poznamenajme, že hodnoty parametrov vystupujúcich vo vzťahu (54) sú rozdielne pre jednotlivé turbíny.

Zároveň si všimnime, že funkcia (54) popisuje výkon jedného generátora, ale v modeli z Kapitoly 1 sa uvažujú výkony po elektrárňach. Optimalizovaním prietokov po celých elektrárňach a nie po jednotlivých generátoroch sa významne redukuje počet premenných v modeli, keďže Vážská kaskáda obsahuje 22 elektrární a až 53 generátorov. V uvedenom zjednodušenom prístupe sa nájdený optimálny prietok cez elektráreň následne rozdelí rovnomerne na prietoky cez jednotlivé aktívne generátory. Toto zjednodušenie je pre potreby praxe akceptovateľné. Poznamenajme, že pri optimalizácii je stále potrebné určiť počet zapnutých generátorov. Počet zapnutých generátorov potom určuje maximálny a minimálny prietok cez elektráreň (viď. vzťahy (11) a (12)).

Ako sme už spomenuli, jedným z porovnávacích kritérií výpočtových techník bude presnosť výpočtu. Pod pojmom presnosť rozumieme odchýlku nájdeného riešenia od stanoveného diagramu výroby. Za týmto účelom definujeme mieru presnosti ako relatívnu

chybu nasledovne

$$error = \frac{\sum_{t=0}^{T-1} \sum_{p=0}^{plant-1} |P_{t,p}^{real} - D_t|}{\sum_{t=0}^{T-1} D_t}, \tag{55}$$

kde D_t je hodnota diagramu v čase t a $P_{t,p}^{real}$ je reálny výkon elektrárne p v čase t. Pod reálnym výkonom elektrárne p chápeme sumu výkonov jednotlivých generátorov prislúchajúcich k elektrárni p, pričom výkon generátora je daný vzťahom (54).

3.1 Zhodnotenie MILP prístupu

V tejto časti zhrnieme výsledky numerických experimentov pri použití lineárnej, resp. kvadratickej aproximácie, ktorú sme popísali v časti 2.1.

Pred samotnou implementáciou problému do vybraných solverov je potrebné ešte upraviť formuláciu účelovej funkcie (1). Problémom je prítomnosť absolútnej hodnoty, ktorá nespĺňa MILP formuláciu. Problém odstránime nasledujúcou úpravou

$$\begin{aligned} \max \quad & \sum_{t=0}^{T-1} -s_t - \sum_{t=0}^{T-1} \sum_{p=0}^{plant-1} Q_spill_pen_{t,p} Qspill_plant_{t,p} \\ & - \sum_{t=0}^{T-1} \sum_{r=0}^{res-1} spill_pen_{t,r}(Qspill_{t,r} - outflows_bio_{t,r}) \\ & - \sum_{t=0}^{T-1} \sum_{g=0}^{gen-1} tg_startup_cost_g \; TG_startup_{t,g} \end{aligned} \tag{56}$$

Absolútnu hodnotu sa nám podarilo odstrániť zavedením novej pomocnej premennej s_t do účelovej funkcie. Aby bola úprava korektná, je potrebné do formulácie úlohy doplniť nasledujúce ohraničenia

$$\begin{aligned} \left(\sum_{p=0}^{plant-1} P_{t,p} \right) - D_t \leq s_t, \quad t = 0, 1, \ldots, T-1 \\ \left(\sum_{p=0}^{plant-1} P_{t,p} \right) - D_t \geq -s_t, \quad t = 0, 1, \ldots, T-1 \\ s_t \geq 0, \quad t = 0, 1, \ldots, T-1 \end{aligned}$$

Podobne ako v práci [1] v numerických experimentoch budeme testovať dva prístupy vyjadrenia funkcie výkonu (54). V prvom z nich budeme modelovať výkon ako funkciu spádu a prietoku, tj. $P_{t,p} = F_1^p(H_{t,p}, Q_{t,p})$. Alternatívou bude vyjadrenie výkonu ako funkcie prietoku a rozdielu objemov hornej a dolnej nádrže, tj. $P_{t,p} = F_2^p(HH_V_{t,p} - DH_V_{t,p}, Q_{t,p})$.

Na riešenie problému optimálnej prevádzky Vážskej kaskády sme využili tri softvérové balíky: Gurobi, Cplex a Knitro. Ako sme spomenuli v Kapitole 2, solver Cplex je schopný riešiť iba MILP formulácie. Z tohto dôvodu nebolo možné použiť solver Cplex na modely s kvadratickou aproximáciou funkcie výkonu. Všetky výpočty sme realizovali s dodatočným ohraničením na dĺžku výpočtu. Horný časový limit sme nastavili na hodnotu 10 minút. Akákoľvek optimalizácia, ktorá by bola časovo náročnejšia, by totižto nebola pre Slovenské elektrárne použiteľná.

Ak solver nenájde optimálne riešenie v danom časovom horizonte, tak po jeho prekročení sa výpočet zastaví a solver vráti najlepšie riešenie, ktoré za daný čas našiel. Dodatočné obmedzenie na dĺžku výpočtu viedlo k vylúčeniu solvera Knitro z ďalších experimentov: ani pri jednej z možných formulácii úlohy nebol tento solver schopný nájsť prípustné riešenie úlohy v požadovanom čase.

3.1.1 Model v priestore prietoku a spádu s lineárnou aproximáciou

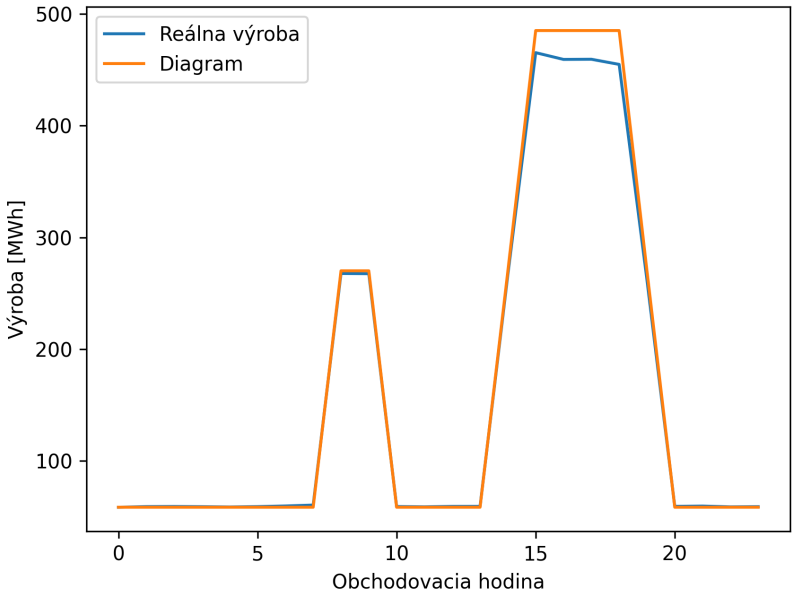
V tejto časti prezentujeme výsledky numerických experimentov v prípade použitia lineárnej aproximácie popísanej v časti 2.1.1. Riešenie úlohy v tvare MILP by malo byť vo všeobecnosti časovo menej náročné než riešenie úlohy v tvare MIQCP. Avšak aproximácia trojuholníkovou metódou si vynútila zavedenie veľkého počtu nových pomocných premenných do formulácie problému. (viď časť 2.1.1, resp. [1]). Výsledok tohto tradeoffu budeme mať možnosť pozorovať v nasledujúcich experimentoch, keď oba prístupy porovnáme z hľadiska dosiahnutej presnosti a výpočtového času.

Uvažujme výkon ako funkciu prietoku a spádu. Formuláciu problému riadenia Vážskej kaskády s takto uvažovaným výkonom, v ktorej sme použili lineárnu aproximáciu funkcie výkonu, označíme ako model $M1$. Posledným parametrom úlohy je počet deliacich bodov pri implementácii trojuholníkovej metódy. Vychádzajúc z výsledkov práce [1] sme priestor prietoku rozdelili tromi a priestor spádu dvomi bodmi. Takéto navýšenie počtu parametrov výrazne neovplyvní rozmer pôvodnej úlohy. Súčasne by toto nastavenie nemalo výrazne ovplyvniť presnosť výpočtu.

Takto pripravenú lineárnu úlohu sme riešili s využitím solverov Gurobi a Cplex. V prvom kroku výpočtu sme sa zaoberali otázkou predspracovania (viď časť 2.1.3). Solver Gurobi ako aj Cplex umožňujú zvoliť si úroveň predspracovania.

predspracovanie	čas optimalizácie s $[s]$
vypnuté	1246
konzervatívne	744
agresívne	686

Tabuľka 4: Efekt predspracovania



Obr. 2: Porovnanie diagramu a reálneho výkonu v priestore prietoku a spádu pri trojuholníkovej aproximácii

Presnejšie, solvery ponúkajú možnosť zvoliť si jednu z dvoch úrovní predspracovania: konzervatívne a agresívne. Rozdiel je v dĺžke času, ktorý sa vyhradí na predspracovanie. Efekt predspracovania sme testovali pri hľadaní riešenia úlohy pomocou solveru Gurobi. Poznamenajme, že solver Gurobi našiel rovnaké riešenie bez ohľadu na spôsob predspracovania. Ako však môžeme vidieť v Tabuľke 4, so zvyšujúcim sa časom stráveným na predspracovaní sa skrátil celkový čas potrebný na optimalizáciu solveru Gurobi, a to až o polovicu. Vzhľadom na tieto výsledky sme sa rozhodli pri všetkých experimentoch využívať agresívne predspracovanie.

Napriek voľbe agresívneho predspracovania sa ani jednému solveru nepodarilo nájsť globálne riešenie do 10 minút. Solver Cplex dokonca nebol schopný v časovom horizonte

10 minút nájsť ani prípustné riešenie.

Na Obr. 2 sme porovnali riešenie nájdené solverom Gurobi po 10 minútach s požadovaným výrobným diagramom. Ako vidíme, nájdené riešenie malo tendenciu preceňovať výkon v hodinách s vysokým požadovaným výkonom. Odchýlka od diagramu definovaná vzťahom (55) vykazovala hodnotu $error = 0.0324$.

3.1.2 Model v priestore prietoku a objemu hornej a dolnej hladiny s lineárnou aproximáciou

Tento prípad je modifikáciou prístupu z časti 3.1.1. Rozdiel je v reprezentácii funkcie výkonu, ktorá je v tomto prípade závislá od prietoku a rozdielu objemu hornej a dolnej hladiny. Priestor objemov bol opäť aproximovaný rôznymi spôsobmi pre rôzne typy elektrárni ako je uvedené v Kapitole 1. Takúto formuláciu úlohy označíme ako M2.

Uvažovaná zmena vo formulácii modelu neviedla k žiadanému výsledku: opäť ani jeden zo solverov nenašiel optimálne riešenie v nastavenom časovom limite 10 minút. Z tohto dôvodu sme sa pozreli aspoň na najlepšie doposiaľ nájdené riešenie. Nájdené riešenia z oboch solverov boli veľmi podobné: o trochu presnejšie riešenie našiel solver Cplex. Miera dosiahnutej presnosti dosiahla hodnotu $error = 0{,}0258$. Na porovnanie chyba pri riešení pomocou Gurobi bola na úrovni $error = 0{,}0271$.

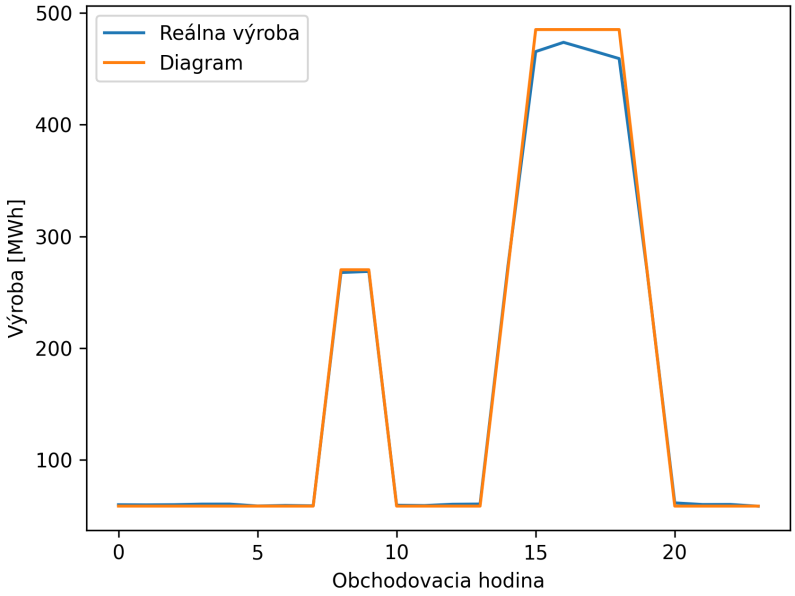
Na Obr. 3 sme graficky porovnali nájdené riešenie pomocou solveru Cplex a požadovaný diagram. Ako môžeme pozorovať, chyba pri aproximácii nastáva predovšetkým v hodinách, v ktorých sa požaduje vysoký výkon.

3.1.3 Model v priestore prietoku a spádu s kvadratickou aproximáciou

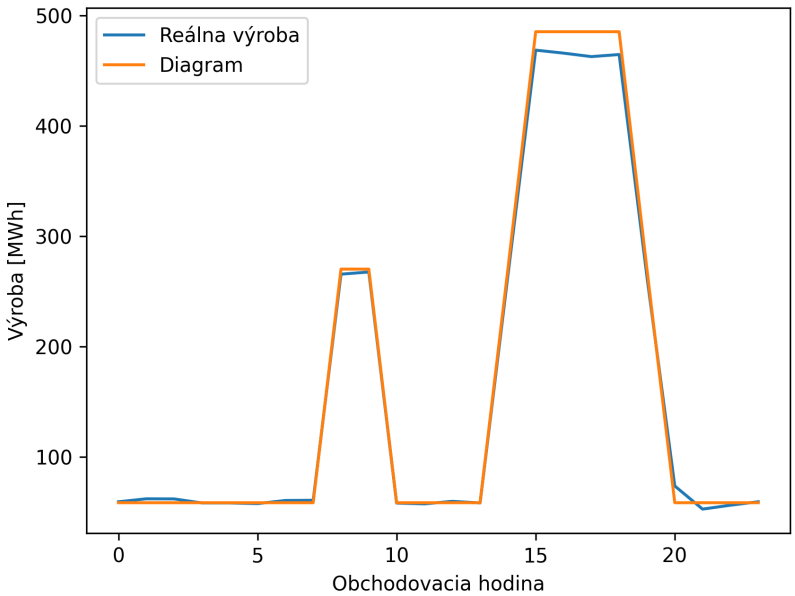
Ďalšou formuláciou, ktorú sme skúmali, bol model v tvare MIQCP. V tomto prístupe sme presnejšie použili kvadratickú aproximáciu funkcie výkonu v priestore prietoku a spádu v nasledujúcom tvare

$$P_{t,p} = a_p Q_{t,p}^2 + b_p Q_{t,p} + d_p H_{t,p} + f_p \tag{57}$$

Parametre a_p, b_p, d_p, f_p boli odhadnuté lineárnou regresiou z nameraných hodnôt. Aproximáciu (57) sme z dostupných variánt opäť zvolili na základe výsledkov experimentov v práci [1]. Tento model označíme ako $M3$.



Obr. 3: Porovnanie diagramu a reálneho výkonu v priestore prietoku a objemu pri trojuholníkovej aproximácii



Obr. 4: Porovnanie diagramu a reálneho výkonu v priestore spádu a prietoku s kvadratickou aproximáciou

Solver Gurobi nebol schopný nájsť optimálne riešenie v časovom horizonte 10 minút. Numerické experimenty ukázali, že pri MIQCP formulácii problému solveru Gurobi trvalo rádovo hodinu nájsť optimálne riešenie problému. Na Obr. 4 je zobrazené najlepšie riešenie nájdené v časovom limite 10 minút. Z obrázku vidíme, že ide o pomerne presnú aproximáciu. K najväčším odchýlkam opäť dochádzalo v hodinách s vysokou požadovanou výrobou. Chyba definovaná v (55) najlepšieho nájdeného riešenia pri tejto MIQCP formulácii bola $error = 0{,}035$.

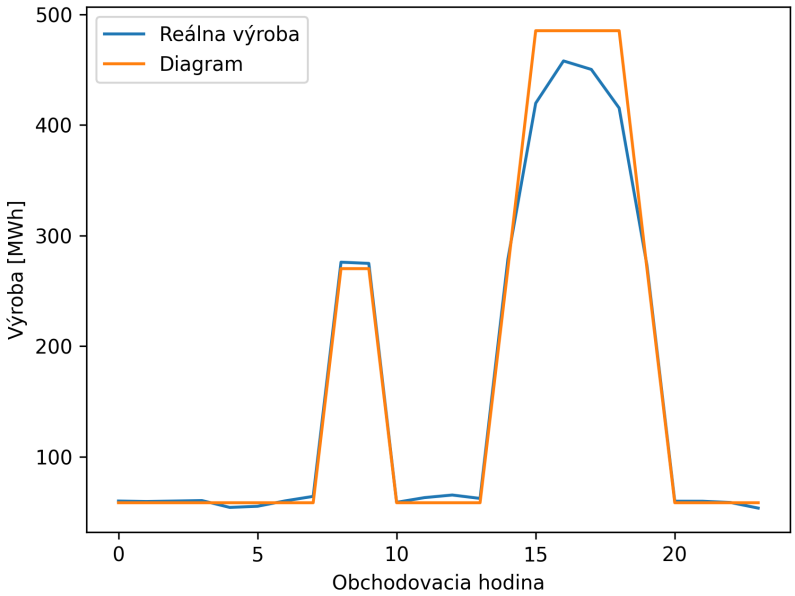
3.1.4 Model v priestore prietoku a objemu hornej a dolnej hladiny s kvadratickou aproximáciou

Poslednou skúmanou modifikáciou bola formulácia úlohy s kvadratickou aproximáciou výkonu. Funkcia výkonu bola v tomto prípade vyjadrená ako funkcia prietoku a rozdielu objemov hornej a dolnej hladiny. Podobne ako v časti 3.1.3 aj pri tomto prístupe musíme zohľadniť rozdielnu polohu elektrárne vzhľadom k vodnej nádrži. Pre jednotlivé typy elektrární definovaných v časti 1.1.2 aproximujeme výkon nasledovne

- **Typ 0:** $P_{t,p} = a_p Q_{t,p}^2 + b_p Q_{t,p} + g_p(HH_V_{t,p} - DH_V_{t,p}) + h_p(HH_V_{t,p} - DH_V_{t,p})^2 + \text{intercept}_p$
- **Typ 1 a 4:** $P_{t,p} = a_p Q_{t,p}^2 + b_p Q_{t,p} + c_p HH_V_{t,p} + d_p(HH_V_{t,p})^2 + \text{intercept}_p$
- **Typ 2:** $P_{t,p} = a_p Q_{t,p}^2 + b_p Q_{t,p} + \text{intercept}_p$
- **Typ 3:** $P_{t,p} = a_p Q_{t,p}^2 + b_p Q_{t,p} + e_p DH_V_{t,p} + f_p(DH_V_{t,p})^2 + \text{intercept}_p$

Pričom veličiny $a_p, b_p, c_p, d_p, e_p, g_p, h_p, \text{intercept}_p$ sú odhadnuté parametre z pozorovaných dát (pre viac informácií viď. [1]). Tento model označíme ako $M4$.

Solver Gurobi opäť nebol schopný nájsť globálne optimum do 10 minút. Na Obr. 5 je porovnanie najlepšieho doposiaľ nájdeného riešenia a požadovaného diagramu. Toto riešenie malo chybu na úrovni $error = 0{,}067$. Vidíme, že oproti lineárnym aproximáciám v tomto prípade dochádza aj k jemným nepresnostiam mimo hodín s vysokou požiadavkou. Zároveň aj celková odchýlka je podstatne vyššia (približne dvojnásobná) ako pri modeloch s lineárnou aproximáciou.



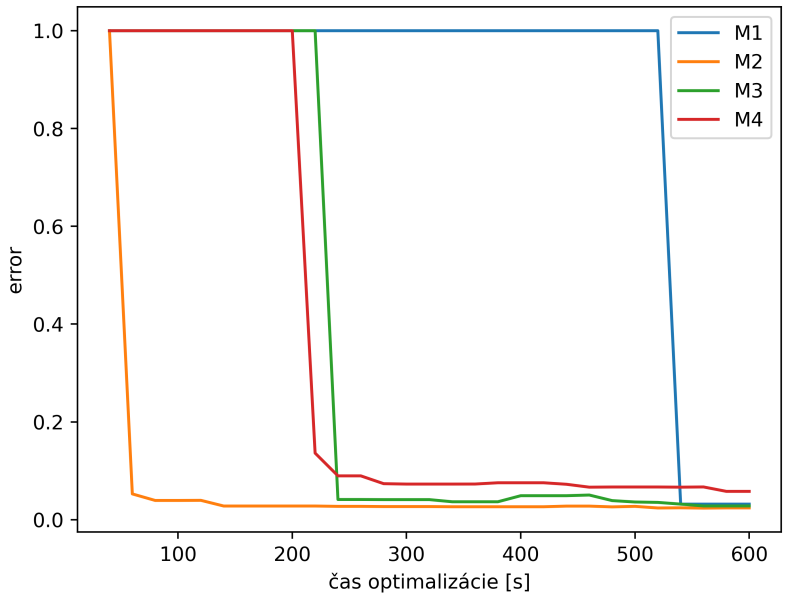
Obr. 5: Porovnanie diagramu a reálneho výkonu v priestore rozdielov objemov a prietoku s kvadratickou aproximáciou

3.1.5 Porovnanie modelov MIP modelov v čase

V predošlých sekciách sme videli, že ani jeden z modelov M1–M4 nebol schopný nájsť optimálne riešenie v časovom horizonte 10 minút. Pozorovali sme však rozdielnu rýchlosť nájdenia prípustného riešenia. Túto skutočnosť sme sa rozhodli bližšie preskúmať.

Na Obr. 6 sme graficky znázornili vývoj *error* jednotlivých modelov v čase. Hodnotu premennej *error* sme získali opakovanými spusteniami jednotlivých modelov s postupne narastajúcou povolenou dĺžkou výpočtu. Pre každý čas sme tak získali najlepšie doposiaľ nájdené riešenie. Tie boli následne použité v simulácii, aby sme získali skutočné výšky hladín a hodnoty funkcie výkonu zo vzťahu (54), keďže v samotnom modeli sú výkon a výška hladiny aproximované. Ak solver nebol schopný na danom časovom horizonte nájsť žiadne prípustné riešenie, tak chyba bola nastavená na hodnotu 1.

Z výsledkov na Obr. 6 vidíme, že model, ktorý využíval lineárnu po častiach spojitú aproximáciu v priestore prietoku a objemu, bol schopný veľmi rýchlo nájsť prípustné riešenie, ale následne už dochádzalo iba k malým zlepšeniam v kvalite riešenia. Navyše aj dosiahnutá presnosť modelu $M2$ bola minimálne porovnateľná s ostatnými modelmi.



Obr. 6: Vývoj chyby MIP modelov v čase

Na druhej strane model $M1$ dlhý čas nebol schopný nájsť žiadne prípustné riešenie, ale následne ihneď našiel veľmi kvalitné riešenie. Takýto výsledok sa dal očakávať, keďže rozmer modelu $M1$ je väčší kvôli potrebe zavedenia viacerých pomocných premenných pri trojuholníkovej aproximácii. Výpočet spomaľuje aj potreba neustáleho prevádzania objemu nádrží na výšky hladín.

Pri modeli $M3$ možno pozorovať úsek, kde sa s časom kvalita riešenia zhoršila. Tento jav bol bol pravdepodobne spôsobený tým, že solver postupne hľadá riešenia, ktorých aproximácia výkonu sa blíži k diagramu, ale môže sa stať, že aj keď sa aproximovaný výkon blíži k požadovanej hodnote, tak reálna hodnota funkcie výkonu (54) sa môže vzdialiť od požadovanej hodnoty.

Na základe týchto výsledkov môžeme z analyzovaných modelov za najlepší považovať model $M2$. Napriek tomu, že nenašiel globálne optimum v stanovenom časovom horizonte, možno ho použiť v prípade, keď je potrebné nájsť čo najrýchlejšie prípustné riešenie, ktoré nie je príliš vzdialené od optimálneho riešenia.

3.2 Lineárny iteratívny model v priestore prietoku

V časti 3.1.5 sme videli, že bez ohľadu na zvolenú aproximáciu a formuláciu hľadanie optimálneho riešenia bolo časovo náročné. Tento fakt bol motiváciou k návrhu optimalizačnej metódy, ktorú popíšeme v tejto časti.

V tomto optimalizačnom prístupe aproximujeme funkciu výkonu jednoduchou lineárnou funkciou závislou iba od prietoku $Q_{t,p}$. Presnejšie aproximovaná funkcia výkonu bude v tvare

$$P_{t,p} = a_p Q_{t,p} + \text{intercept}_p, \tag{58}$$

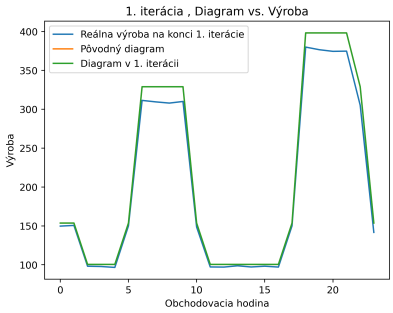
kde koeficienty a_p a intercept_p boli odhadnuté pomocou lineárnej regresie. Touto aproximáciou zabezpečíme, že model bude v tvare MILP formulácie. Navyše sme sa vyhli zavedeniu množstva pomocných premenných, ktoré boli potrebné pri trojuholníkovej aproximácii.

Ako ukázali numerické experimenty, nájsť optimálne riešenie pre takto zjednodušený model je možné za veľmi krátky čas. Nevýhodou aproximácie (58) je veľmi malá presnosť. Z tohto dôvodu sme navrhli nasledujúci iteratívny prístup. Hlavnou myšlienkou je využiť rýchlosť výpočtu pri použití jednoduchej lineárnej aproximácie funkcie výkonu a zlepšovanie presnosti postupným prispôsobovaním hodnoty vstupného diagramu na základe výsledkov z predchádzajúcej iterácie.

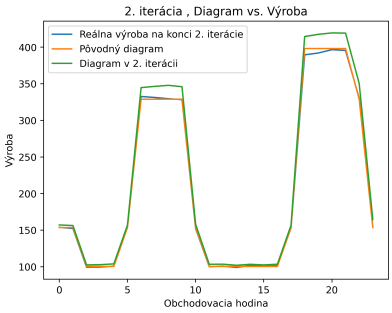
Iteračný algoritmus môžeme popísať nasledovne. Označme si D_t^i ako hodnotu diagramu v čase t a v iterácii i. Podobne označme $P_{t,p}^i$ ako reálny výkon elektrárne p v čase t a v iterácii i. Potom D_t^1 je hodnota pôvodného diagramu v čase t a $P_{t,p}^1$ je výsledný reálny výkon elektrárne p v čase t po optimalizácii. Hodnoty diagramov v nasledujúcich iteráciách budeme potom počítať ako

$$D_t^{i+1} = D_t^i + \alpha^i \left(D_t^1 - \sum_{p=0}^{plant-1} P_{t,p}^{i-1} \right), \quad t = 0, 1, \ldots, T-1;\ i = 2, 3, \ldots \tag{59}$$

Vzťah (59) bol navrhnutý na základe nasledujúcej myšlienky. Ak v nejakej iterácii i je po optimalizácii reálna výroba v čase t vyššia ako hodnota pôvodného diagramu D_t^1, tak v ďalšej iterácii bude hodnota D_t^{i+1} menšia oproti hodnote D_t^i. Takýmto postupom budeme umelo navyšovať hodnoty požadovaného diagramu v časoch, kde aproximovaný výkon podhodnocuje reálny výkon. Podobne v časoch, keď aproximácia výkonu preceňuje



Obr. 7: Porovnanie výroby a diagramu v 1. iterácii



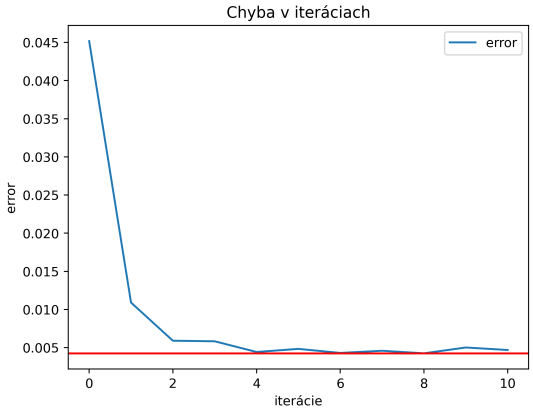
Obr. 8: Porovnanie výroby a diagramu v 2. iterácii

reálny výkon, hodnoty diagramu znížime. Parameter $\alpha \in \langle 0, 1\rangle$ slúži na to, aby sa po čase diagram ustálil a tak sa zabránilo prípadnému zacykleniu algoritmu.

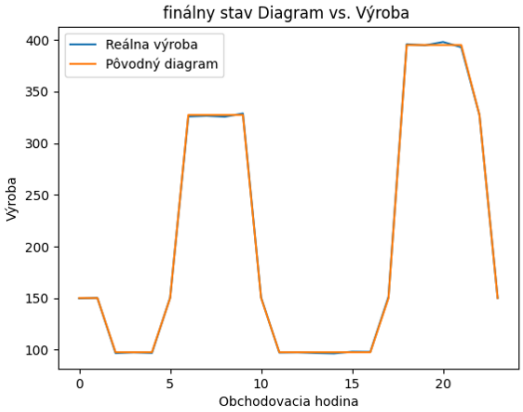
Na Obr. 7 a Obr. 8 sme graficky ilustrovali navrhnutý iteračný prístup. V prvej iterácii (viď Obr. 7) je hodnota diagramu D^1 a pôvodného diagramu identická. Vidíme, že v prvej iterácii bola reálna výroba v niektorých hodinách príliš nízka, preto boli v týchto hodinách hodnoty diagramu v druhej iterácii navýšené. Ako vidíme na Obr. 8 pri takomto postupe sa reálna výroba v druhej iterácii viac priblížila pôvodnému diagramu.

Iterácie sme opakovali až pokým sa nenaplnil časový limit stanovený na hodnote jednej minúty. Model sme opäť implementovali pomocou solverov Gurobi a Cplex. Krátky optimalizačný čas nám zároveň umožnil porovnať výpočtové časy jednotlivých solverov.

Výpočtové časy solverov sme porovnávali vzorke 200 iterácii. Výsledky ukazujú, že solver Cplex je schopný problém riešiť trochu rýchlejšie. Priemerný čas potrebný na optimalizáciu MILP úlohy v jednej iterácii pomocou solveru Gurobi bol 2.89 sekundy, zatiaľ čo priemerný čas výpočtu za pomoci solveru Cplex bol na úrovni 2.57 sekundy. Treba poznamenať, že na horizonte jednej minúty nejde o veľký rozdiel, keďže mimo samotnej optimalizácie je v tomto prístupe ešte potrebné presimulovať reálne výšky hladín a reálne výkony turbín. V priemere dokážeme s využitím solveru Gurobi za jednu minútu uskutočniť 10 iterácií algoritmu a pomocou solveru Cplex 11 iterácií. Tomu zodpovedajú aj finálne výsledky iteratívneho algoritmu, kde výstupy získané pomocou Cplexu sú jemne lepšie ako výsledky získané pomocou Gurobi. Chyba pri optimalizovaní pomocou Gurobi bola na úrovni $error = 0{,}0046$ zatiaľ čo chyba pri optimalizácii pomocou Cplex bola na úrovni $error = 0{,}0043$.



Obr. 9: Vývoj chyby lineárneho iteratívneho modelu



Obr. 10: Porovnanie finálnej výroby a pôvodného diagramu

Na Obr.9 je znázornené, ako sa vyvíjala chyba v iteráciách pri využití solveru Cplex. Červená horizontálna čiara zodpovedá najnižšej dosiahnutej hodnote veličiny *error*. Vidíme, že pri postupe definovanom vzťahom (59) nie je zaručené, že odchýlka od diagramu sa zmenšuje každou iteráciou.

Na Obr. 10 môžeme porovnať presnosť tejto metódy na vybranom obchodovacom dni, ktorú sme dosiahli vo výpočtovom čase rovnej jednej minúte. Ako vidíme, podarilo sa nám dosiahnuť dostatočne veľkú presnosť. Navyše sme neprekročili požiadavku na dĺžku výpočtového času. Oproti štandardným numerickým technikám popísanými v prechádzajúcich častiach sme dosiahli rádovo presnejší výsledok v rádovo nižšom výpočtovom čase.

3.3 Zhodnotenie heuristických prístupov

Heuristické prístupy nevyžadujú na vstupe lineárny model, čo možno považovať za ich hlavnú výhodu. Funkciu výkonu (54), ako aj ostatné nelineárne vzťahy vystupujúce v modeli optimálnej prevádzky Vážskej kaskády, preto nie je nutné aproximovať. Avšak heuristické prístupy majú niektoré iné nevýhody. V prípade aplikácie heuristických prístupov na problém popísaný v Kapitole 1 sa jednou z výpočtovo náročných úloh ukázala potreba na základe riadenia dopočítať objemy a výšky hladín celej kaskády pri vyhodnocovaní každého kandidáta, aby sa dala určiť jeho prípustnosť. Podrobnejšie zhodnotenie aplikovateľnosti heuristických prístupov si predstavíme v nasledujúcich častiach.

3.3.1 Aplikácia genetického algoritmu

Pri aplikácii genetických algoritmov sme využili vstavané metódy z balíka Pygad (viď. [20]). Riadiacimi premennými pri tomto prístupe boli premenné $Q_{t,p}$, $Q_spill_{t,p}$ a $Q_spill_plant_{t,p}$, pre $t = 0, 1, \ldots, T-1$; $p = 0, 1, \ldots, plant-1$ a $r = 0, 1, \ldots, res-1$. Podobne ako pri aplikáciach MIP formulácie budeme uvažovať prietok cez elektráreň $Q_{t,p}$ namiesto prietokov cez jednotlivé generátory.

Riešenie (chromozóm) potom zakódujeme ako vektor pozostávajúci z nasledujúcich génov

$$x = (Q_{0,0}, \cdots, Q_{T-1,plant-1}, Q_spill_{0,0}, \cdots, Q_spill_{T-1,res-1}, \\ Q_spill_plant_{0,0}, \cdots, Q_spill_plant_{T-1,plant-1}), \quad (60)$$

kde

$$Q_{t,p} \in \{0\} \cup \langle 0, Q^{max}_{t,p} \rangle \quad t\text{=}0,\ldots,T\text{-}1;\ p\text{=}0,\ldots,plant\text{-}1 \quad (61)$$

$$Q_spill_{t,r} \in \langle outflows_bio_{t,r}, Q_spill^{max}_{t,r} \rangle \quad t\text{=}0,\ldots,T\text{-}1;\ r\text{=}0,\ldots,res\text{-}1 \quad (62)$$

$$Q_spill_plant_{t,p} \in \langle 0, Q_spill_plant^{max}_{t,p} \rangle \quad t\text{=}0,\ldots,T\text{-}1;\ p\text{=}0,\ldots,plant\text{-}1 \quad (63)$$

Parametre $Q_spill^{max}_{t,r}$, resp. $Q_spill_plant^{max}_{t,p}$ vyjadrujú maximálny jalový prietok nádrže, resp. jalový odtok elektrárne, ktoré boli zavedené pre potreby modulu Pygad. Pomocou riešenia x vieme jednoznačne dopočítať objemy nádrží $V_{t,r}$ a výšky hladín $H_{t,p}$. Získané hodnoty následne použijeme na vyčíslenie hodnôt reálnej funkcie výkonu $P_{t,p}$. Priestor vymedzený ohraničeniami (61)–(63) stále môže obsahovať aj neprípustné riadenia Vážskej kaskády. Preto je potrebné do účelovej funkcie zahrnúť penalizácie za porušenie prípustnosti. Modifikovanú účelovú funkciu vyjadríme v tvare

$$\sum_{t=0}^{T-1} - \left| \left(\sum_{p=0}^{plant-1} P_{t,p} \right) - D_t \right| - \sum_{t=0}^{T-1} \sum_{p=0}^{plant-1} Q_spill_pen_{t,p} Qspill_plant_{t,p} \\ - \sum_{t=0}^{T-1} \sum_{r=0}^{res-1} spill_pen_{t,r}(Qspill_{t,r} - outflows_bio_{t,r}) \quad (64) \\ + \sum_{t=0}^{T-1} \sum_{r=0}^{res-1} (V_penalt_min_{t,r} \min(0, V_{t,r} - V^{min}_{t,r}) + V_penalt_max_{t,r} \min(0, V^{max}_{t,r} - V_{t,r})),$$

kde $V_penalty_min_{t,r}$,resp. $V_penalty_max_{t,r}$ sú konštanty, ktoré penalizujú prekročenie minimálneho, reps. maximálneho objemu v nádržiach. Uvedomme si, že navrhnutá penalizácia neprípustných riešení bude fungovať len v prípade, ak penalizačné konštanty budú

nastavené na rádovo vyššie hodnoty ako možné odchýlky od diagramu. Táto podmienka zabezpečí, že každé prípustné riešenie bude mať väčšiu účelovú hodnotu (fitness) ako akékoľvek neprípustné riešenie.

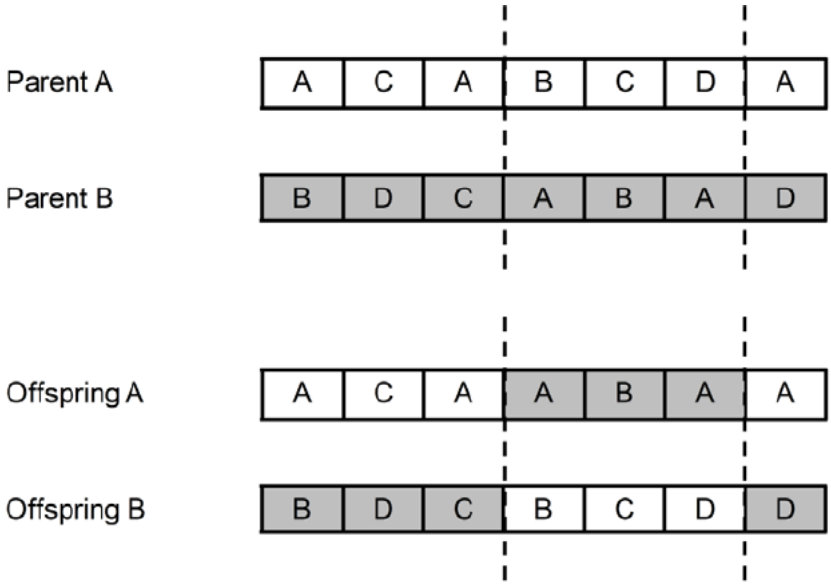
Dôležitou súčasťou genetických algoritmov je voľba počiatočnej populácie. Táto hodnota môže výrazným spôsobom ovplyvniť výkonnosť algoritmu. Modul Pygad štandardne generuje jednotlivé riešenia počiatočnej populácie náhodne. Presnejšie, zložky jednotlivých riešení prislúchajúce premenným $Q_spill_{t,r}$ $Q_spill_plant_{t,p}$ sa vygenerujú z rovnomerného rozdelenia na intervaloch danými vzťahmi (62)–(63). Špeciálny prístup je potrebný pri generovaní zložiek riešenia zodpovedajúcim premennej $Q_{t,p}$, keďže vzťah (61) nevymedzuje konvexnú množinu. Hodnoty $Q_{t,p}$ preto generujeme nasledujúcim postupom. Vygenerujeme číslo Q_{tmp} z rovnomerného rozdelenia na intervale $\langle 0, Q_{t,p}^{max} \rangle$. Hodnota $Q_{t,p}$ potom bude stanovená ako

$$Q_{t,p} = \begin{cases} 0, & \text{ak } Q_{tmp} < Q_{t,p}^{min} \\ Q_{tmp}, & \text{inak.} \end{cases} \tag{65}$$

Numerické experimenty ukázali, že genetický algoritmus dosahuje presnejšie výsledky, ak do počiatočnej populácie zahrnieme aj riešenia prevádzky Vážskej kaskády z predchádzajúcich dní.

Ďalším krokom algoritmu je výber rodičov (selekcia). Existuje veľké množstvo spôsobov, ktorými vybrať z populácie rodičov. Informácie k jednotlivým metódam možno nájsť v [17]. Numerické experimenty ukázali, že najlepšia výkonnosť sa dosahuje pri turnajovej selekcii. Tá prebieha tak, že sa náhodne z populácie vyberie podmnožina k riešení (modul Pygad štandardne využíva $k = 3$). Z tejto podmnožiny sa riešenie s najväčšou hodnotou účelovej funkcie (fitness) vyberie ako rodič. Tento postup opakujeme $n-$krát, kde n je počet požadovaných rodičov.

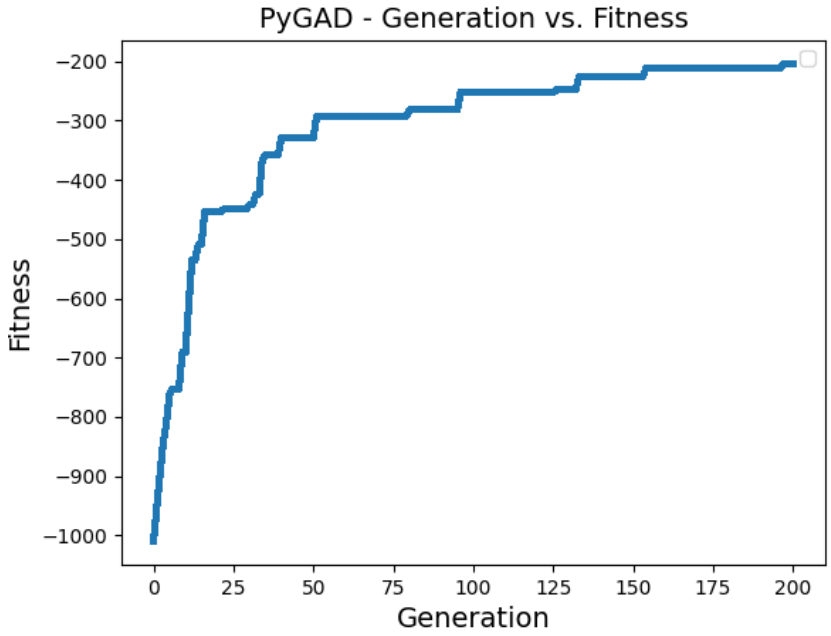
Ďalším operátorom, ktorý sa vyskytuje v genetických algoritmoch, je operátor kríženia. Pri aplikácii numerických experimentov sme skúšali aplikovať jednotlivé operátory kríženia uvedené v [20]. Experimenty ukázali, že najlepšiu výkonnosť dosahoval algoritmus pri dvoj-bodovom krížení. To prebieha tak, že sa vyberú dve náhodné pozície v rodičovských chromozómoch (v oboch rovnaké pozície). Úsek chromozómov medzi týmito dvoma pozíciami si následne rodičia vymenia. Na Obr.11 je názorne ilustrované dvoj-bodové kríženie.



Obr. 11: Dvoj-bodové kríženie, zdroj: [18]

Posledným operátorom, ktorý dokáže výrazne ovplyvniť výkonnosť algoritmu, je operátor mutácie. V bežnom genetickom algoritme mutácia funguje tak, že sa vyberie jedna pevná mutačná pravdepodobnosť p_{mut}. Každý gén riešenia potom zmutuje s pravdepodobnosťou p_{mut}. Samotná mutácia prebieha tak, že gén bude nahradený novou náhodnou hodnotou, ktorá sa generuje rovnako ako pri generovaní zložiek riešení v počiatočnej populácii. Nevýhodou tohto postupu je, že všetky riešenia majú rovnakú pravdepodobnosť mutácie p_{mut} bez ohľadu na hodnotu účelovej funkcie v danom riešení. To znamená, že s rovnakou pravdepodobnosťou zmutujú gény dobrých ako aj zlých riešení. Z tohto dôvodu sa pravdepodobnosť mutácie nastavuje na nízke hodnoty. To však spomaľuje nahrádzanie zlých génov v populácii. V práci [19] poskytujú riešenie tohoto problému aplikáciou adaptívnej pravdepodobnosť mutácie, ktorá funguje nasledovne

- 1. Vypočítame priemernú hodnotu účelovej funkcie populácie f_{avg}.
- 2. Pre každé riešenie v populácii vyčíslime hodnotu účelovej funkcie f.
- 3. Ak $f < f_{avg}$, tak riešenie považujeme nekvalitné a pravdepodobnosť mutácie pre toto riešenie zvolíme vysokú.
- 4. Ak $f > f_{avg}$, tak riešenie považujeme kvalitné a pravdepodobnosť mutácie pre toto riešenie zvolíme nízku.



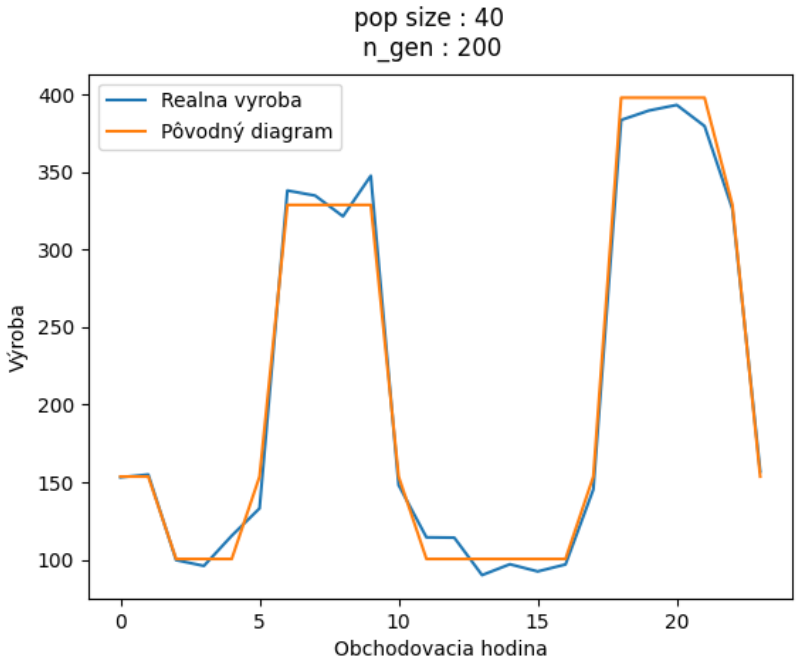
Obr. 12: Vývin účelovej funkcie počas generácií

V prípade riešenia optimalizačného problému Vážskej kaskády numerické experimenty ukázali, že použitím adaptívnej pravdepodobnosti mutácie dochádza k zvýšeniu výkonnosti genetického algoritmu. Pri aplikácii genetického algoritmu sme získali najlepšie výsledky v prípade, ak sme zvolili počiatočnú populáciu veľkosti 40 jedincov, z čoho 7 jedincov boli historické riešenia posledných 7 dní (experimentálne najlepšie výsledky) získané za pomoci lineárneho iteratívneho modelu. Pre takéto nastavenie počiatočnej populácie sme na Obr. 12 graficky znázornili vývoj hodnoty účelovej funkcie v jednotlivých generáciách. Z obrázka vidíme, že spočiatku rýchlo rástla účelová hodnota. Tento progres sa postupujúcimi generáciami spomaľoval.

Na Obr. 13 sme graficky porovnali získané riešenie s požadovaným výrobným diagramom. Chyba tejto metódy bola $error = 0{,}039$. Ako si môžeme všimnúť, ide o podobnú odchýlku, aká bola dosiahnutá pri aplikácii algoritmov v časti 2.1. Navyše čas optimalizácie bol až 4820s. Táto hodnota vysoko presahuje časovú požiadavku Slovenských elektrární.

3.3.2 Aplikácia simulovaného žíhania

V tejto časti prezentujeme výsledky numerických experimentov s využitím metódy simulovaného žíhania (SA) popísanej v časti 2.2.2. Ako sme spomínali v Kapitole 2, kľúčo-



Obr. 13: Porovnanie reálnej výroby vzhľadom k diagramu

vým mechanizmom simulovaného žíhania je generovanie blízkych riešení. Praktické experimenty ukázali, že kvôli veľkému rozmeru úlohy nebol algoritmus SA schopný dostatočne efektívne prehľadávať priestor riešení. V prípade náhodného štartovacieho bodu SA algoritmus nebol schopný dospieť ku kvalitnému výsledku v rozumnom čase. Z tohto dôvodu sme navrhli nasledujúci postup pri riešení. Navrhovaný postup vychádza z myšlienky spojenia genetického a SA algoritmu. Genetický algoritmus použijeme na nájdenie „dobrého" kandidáta na riešenie. Následne tohto „dobrého" kandidáta použijeme ako štartovací bod pre algoritmus SA. Alternatívne môžeme ako štartovací bod použiť aj riešenie získané pomocou iteračného lineárneho modelu z časti 3.2.

Experimentálne sa ukázalo, že v simulovanom žíhaní je efektívnejšie upravovať už iba hodnoty prietokov $Q_{t,p}$, pre $t = 0, 1 \ldots T-1$, $p = 0, 1, \ldots plants - 1$. Hodnoty $Q_spill_{t,r}$ a $Q_spill_plant_{t,p}$ ostatnú počas behu algoritmu nezmenené, tj. na rovnakej hodnote ako boli v štartovacom bode algoritmu.

Ako sme spomenuli, kľúčovým bodom algoritmu SA je vyberanie blízkych riešení (susedov) k danému riešeniu. V každej iterácii náhodne zmeníme hodnotu prietoku pre jednu elektráreň v jednej časovej etape. Na začiatku iterácie stanovíme množinu časových indexov T_d, v ktorých sa absolútny rozdiel produkcie odlišuje od diagramu o viac ako stanovenú

konštantu max_dif, teda

$$T_c = \left\{ t = 0, 1 \dots T-1, \left| \left(\sum_{p=0}^{plants-1} P_{t,p} \right) - D_t \right| \geq max_dif \right\}.$$

Z množiny T_c náhodne vyberieme jeden čas t_c. Ak je reálna výroba v čase t_c nižšia ako hodnota diagramu, tak vyberieme náhodne elektráreň p_c takú, aby platilo $Q_{t_c,p_c} < Q^{max}_{t_c,p_c}$. Následne vygenerujeme z rovnomerného rozdelenia na intervale $\langle Q_{t_c,p_c}\,,\, Q^{max}_{t_c,p_c} \rangle$ novú hodnotu prietoku Q_{t_c,p_c}.

Podobne aj v prípade, ak je výroba v čase t_c vyššia ako hodnota diagramu v danom čase, tak vyberieme náhodne elektráreň p_c takú, pre ktorú platí $Q_{t_c,p_c} > 0$. Následne vygenerujeme hodnotu Q_c z rovnomerného rozdelenia na intervale $\langle 0\,,\, Q_{t_c,p_c} \rangle$. Novú hodnotu prietoku Q_{t_c,p_c} stanovíme ako

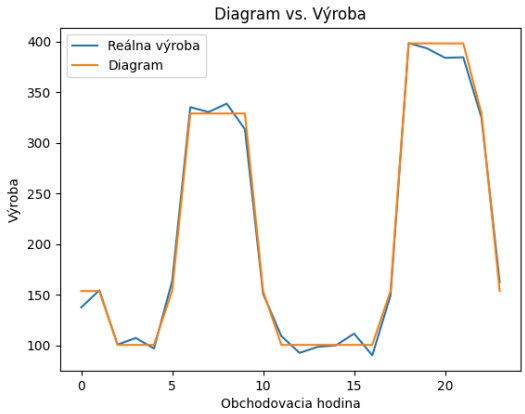
$$Q_{t_c,p_c} = \begin{cases} 0, & \text{ak } Q_c < Q^{min}_{t_c,p_c} \\ Q_c, & \text{inak.} \end{cases} \tag{66}$$

Účelová funkcia ostane rovnaká ako v prípade genetických algoritmov (viď ((64)). Akceptácia horších riešení je rovnaká, ako sme popísali v časti 2.2.2. V práci sme použili exponenciálny plán ochladzovania.

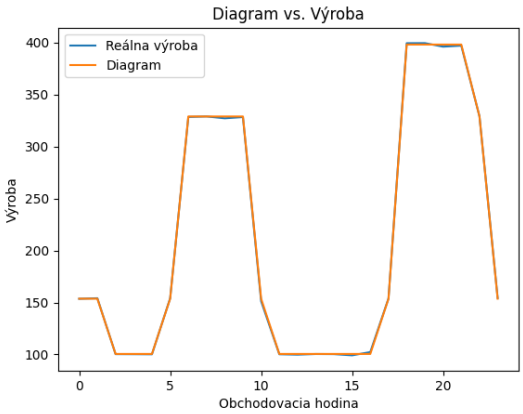
Pri kombinácii SA algoritmu s genetickým algoritmom vyčleníme SA algoritmu 20s na optimalizáciu. Podotýkame, že na získanie počiatočného riešenia pomocou genetických algoritmov bolo potrebných 4820s, čo je rádovo vyššia hodnota ako čas vyčlenený na SA. Celkový čas optimalizácie prostredníctvom genetických algoritmov v kombinácii s SA algoritmom je teda 4840s.

Na Obr. 14 je znázornené riešenie získané SA algoritmom, pričom počiatočné riešenie bolo získané pomocou genetických algoritmov popísaných v časti 3.3.1. Pre pripomenutie aplikáciou genetických algoritmov sme obdržali riešenie, ktorého odchýlka od požadovaného diagramu bola na úrovni $error = 0{,}039$. Po aplikácii SA algoritmu bola odchýlka nového riešenia na úrovni $error = 0{,}035$.

Podobne aj pri kombinácii iteratívneho lineárneho modelu s SA algoritmom je potrebné rozdeliť čas medzi tieto dve optimalizačné metódy, keďže chceme zachovať požiadavku Slovenských elektrární, aby čas výpočtu nepresahoval jednu minútu. Numerické experimenty ukázali, že optimálne výsledky sa dosahujú, ak na lineárny iteratívny algoritmus vyčleníme 40s a zvyšných 20s bude pridelených SA algoritmu.



Obr. 14: Porovnanie výroby a diagramu s počiatočným riešením získaným pomocou genetického algoritmu



Obr. 15: Výsledok SA algoritmu a diagramu s počiatočným riešením získaným pomocou lineárneho iteratívneho modelu

Na Obr. 15 je znázornené riešenie získané pomocou simulovaného žíhania, pričom počiatočné riešenie bolo získané pomocou lineárneho iteratívneho modelu popísaného v časti 3.2. Chyba počiatočného riešenia bola na úrovni $error = 0{,}0043$. S využitím SA algoritmu výsledná chyba klesla na hodnotu $error = 0{,}0037$. Z výsledkov vidíme, že algoritmus SA bol schopný vyladiť drobné odchýlky od diagramu.

3.4 Vzájomné porovnanie

V predchádzajúcich častiach sme predstavili výsledky numerických experimetov, v ktorých sme aplikovali relevantné výpočtové techniky na problém optimálnej prevádzky Vážskej kaskády. Na tomto mieste vzájomne porovnáme získané výsledky. Porovnávacím kritétiom bude presnosť a čas výpočtu. Výsledky sú zhrnuté v Tabuľke 5.

model	aproximácia výkonu	premenné výkonu	error	čas $[s]$
$M1$	po častiach lineárna	$P(Q,H)$	0.0324	600
$M2$	po častiach lineárna	$P(Q,V)$	0.0258	600
$M3$	kvadratická	$P(Q,H)$	0.035	600
$M4$	kvadratická	$P(Q,V)$	0.067	600
iteratívny model	lineárna	$P(Q)$	0.0042	60
iteratívny model + SA	lineárna	$P(Q)$	0.0037	60
genetický algoritmus	žiadna	$P(Q,H)$	0.039	4820
genetický algoritmus + SA	žiadna	$P(Q,H)$	0.035	4840

Tabuľka 5: Porovnanie presnosti metód na jednom dni

Spomedzi modelov $M1$ až $M4$ najväčšiu presnosť dosiahol model $M2$. Tento výsledok zodpovedá aj výsledkom v práci [1]. Treba ale podotknúť, že pri minimalizácii odchýlky od diagramu modely $M1$ až $M4$ potrebovali podstatne dlhší výpočtový čas ako v prípade maximalizácie zisku v práci [1]. Pre ilustráciu uveďme, že podľa práce [1] pri maximalizácii zisku bol najpomalším spomedzi týchto modelov model $M4$, ktorý potreboval 169.14 sekúnd na nájdenie optimálneho riešenia, zatiaľ čo v našom prípade ani jeden z týchto modelov nebol schopný nájsť optimálne riešenie v časovom rozmedzí 10 minút.

Pri pohľade na výsledky získané pomocou genetických algoritmov vidíme, že presnosť výstupu je porovnateľná s výstupmi získanými pomocou modelov $M1$ až $M4$, ale čas potrebný na nájdenie riešenia je rádovo vyšší.

Výsledky numerických experimentov jednoznačne preukázali, že lineárny iteratívny model bol rádovo presnejší oproti ostatným modelom a rovnako aj rádovo najrýchlejší.

4 Finálne zhrnutie a porovnanie

V Kapitole 3 sme si predstavili výsledky numerických experimentov pri aplikovaní jednotlivých metód riešenia na jednom dni výroby. Testovať presnosť a rýchlosť všetkých metód na väčšom počte dní by bolo časovo a výpočtovo náročné.

V tejto Kapitole sa preto zameriame na podrobnejšie testovanie len najsľubnejších výpočtových techník: lineárny iteratívny model, genetické algoritmy a kombinácia lineárneho iteratívneho modelu s SA algoritmom. Všetky spomenuté prístupy porovnáme na vzorke 25 náhodne vybraných obchodovacích dní.

Vstupné diagramy sme generovali z výsledkov získaných pomocou optimalizačného modelu popísaného v práci [1]. Názorne si ukážeme, ako pre daný deň vygenerujeme vstupný diagram. Z výsledkov optimálnej prevádzky maximalizujúcej zisk získame objemy elektrickej energie v danom dni. Následne objemy elektriny perturbujeme, tj. k objemom elektrickej energie v daných hodinách pripočítame náhodné vygenerované číslo z $N(0, 30)$ [1]. Keďže chceme, aby boli hodnoty produkcie predpísané diagramom dosiahnuteľné (tj. aby neboli príliš vysoké), perturbované hodnoty v jednotlivých hodinách prenásobíme koeficientom $\beta \in (0,\ 1)$. V práci sme zvolili parameter $\beta = 0.9$. Označme si tieto prenásobené hodinové objemy elektrickej energie v rámci dňa ako P_t, pre $t = 0, 1, \ldots, 23$. Ďalej na získané hodnoty v rámci dňa aplikujeme algoritmus K-means, ktorý dáta rozdelí do I skupín (zhlukov). Algoritmus K-means začlení objemy elektrickej energie do skupín tak, že v každej skupine budú podobné objemy elektrickej energie[2].

Označme C_j ako množinu tých hodín, ktoré prislúchajú do $j-$tého zhluku, pre $j = 1, \ldots, I$. Ďalej označme $\bar{P}_j$ ako priemerný zobchodovaný objem elektrickej energie $j-$teho zhluku, tj.

$$\bar{P}_j = \frac{1}{n_j} \sum_{t \in C_j} P_t,$$

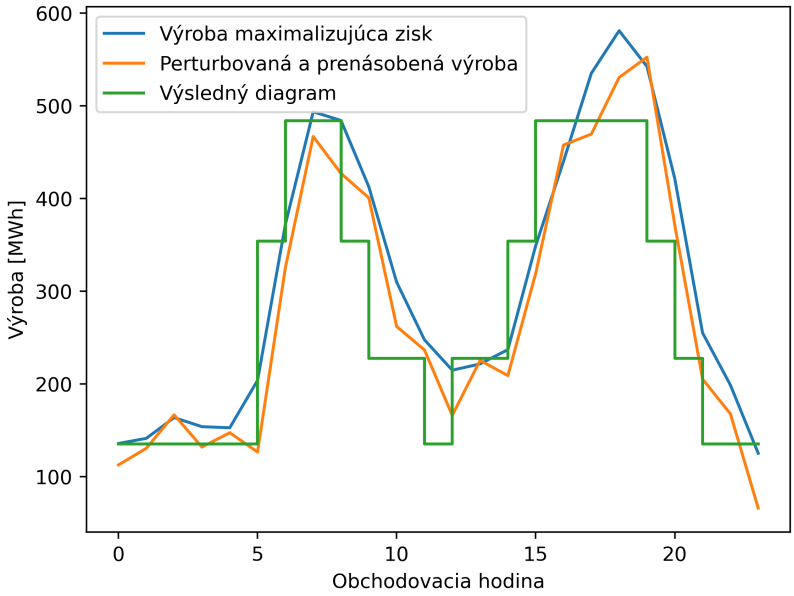
kde n_j je počet indexov v zhluku C_j.

Hodinové hodnoty vstupných diagramov D_t sme potom nastavili podľa vzťahu

$$\forall j \in \{1, \ldots, I\}; \forall i \in C_j : D_i = \bar{P}_j$$

[1] Hodnota variancie bola zvolená subjektívne vzhľadom k hodnotám objemov elektrickej energie.

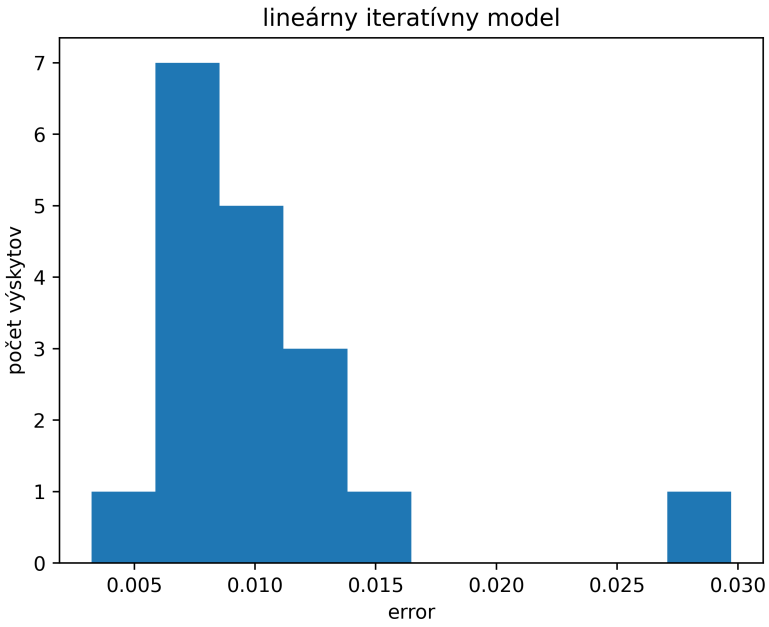
[2] Viac podrobností o fungovaní algoritmu K-means možno nájsť v [21]



Obr. 16: Generovanie diagramu výroby

Na Obr. 16 je názorne zobrazený proces generovania vstupného diagramu pre parameter $I = 4$.

Tento proces opakujeme pre každý zvolený obchodovací deň (tj. 25 dní). Pre každý z 25 získaných diagramov vyriešime optimalizačný problém riadenia Vážskej kaskády popísaný v Kapitole 1.



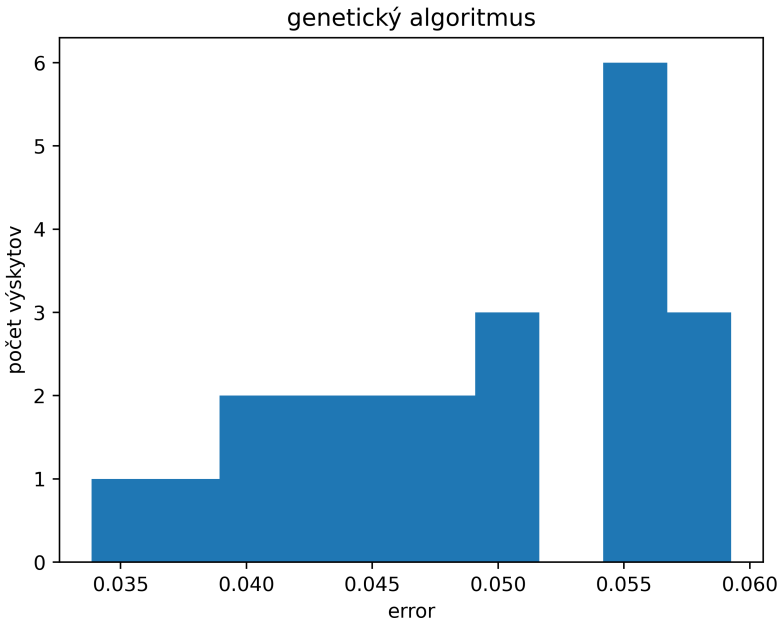
Obr. 17: Histogram odchýlok lineárneho iteratívneho modelu

Na Obr. 17 možno vidieť histogram odchýlok od diagramov, kde riešenia boli získané pomocou iteratívneho algoritmu. Priemerná odchýlka daných údajov je $\overline{error} = 0.0103$, čo je pomerne vyššia odchýlka, akú sme mali pri aplikácii modelu na jeden deň. Na Obr. 17 sa nachádza aj pozorovanie, ktoré by sme mohli označiť ako outliera.

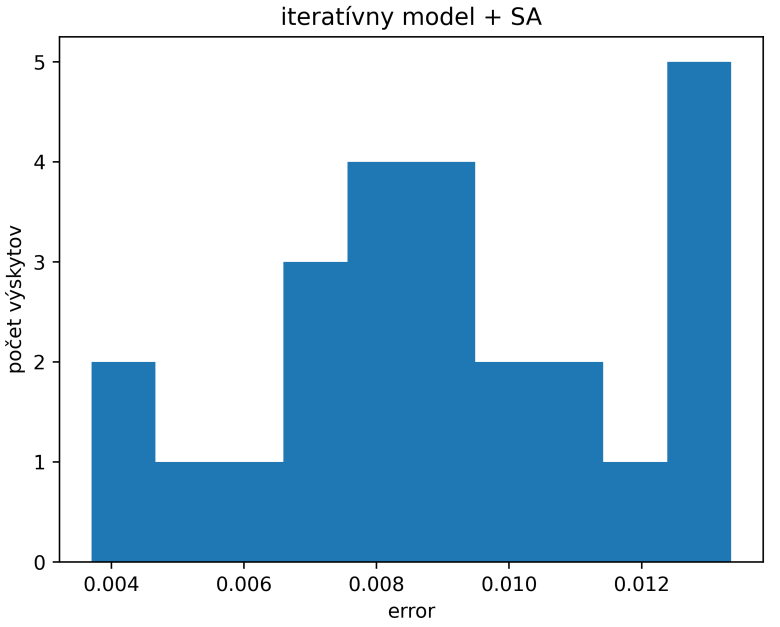
Na Obr. 18 vidíme histogram odchýlok od diagramu riešení získaných s využitím genetických algoritmov. Z histogramu vidíme, že odchýlka od diagramu je vykazuje omnoho väčšiu variabilitu ako pri lineárnom iteratívnom modeli. Priemer odchýlok od diagramu pre genetické algoritmy je $\overline{error} = 0{,}0493$.

Posledným skúmaným prístupom je kombinácia lineárneho iteratívneho modelu s SA algoritmom. Histogram získaných odchýlok možno vidieť na Obr. 19. Priemerná odchýlka bola na úrovni $\overline{error} = 0{,}0095$.

Z dostupných výsledkov je možné zhodnotiť, že kombinácia lineárneho iteratívneho modelu s SA algoritmom je najlepšou voľbou čo sa týka presnosti riešenia, ale aj rýchlosti optimalizácie zo všetkých predstavených metód.



Obr. 18: Histogram odchýlok genetického algoritmu



Obr. 19: Histogram odchýlok kombinácie lineárneho iteratívneho modelu a SA algoritmu

Záver

Riadenie prevádzky prepojeného systému 22 vodných elektrární Vážskej kaskády je komplexný optimalizačný problém. Veľký rozmer úlohy spolu s množstvom technických a prevádzkových obmedzení si žiada hľadanie efektívnych optimalizačných techník, s pomocou ktorých je možné nájsť riešenie v požadovanom čase.

V tejto diplomovej práci sme sa venovali optimalizácii prevádzky vodných elektrární na Vážskej kaskáde na časovom horizonte jedného dňa. Pod pojmom *optimálne riadenie prevádzky* sme rozumeli také riadenie výroby elektrickej energie, pri ktorom sa minimalizovala odchýlka sumárneho výkonu elektrární od predpísanému plánu výroby na danú hodinu. Dôraz pri optimalizácii sa kládol na časovú náročnosť optimalizácie, keďže jedným z cieľov bolo splniť požiadavku z praxe od Slovenských elektrární na získanie riešenia v čo najkratšom čase.

V práci sme predstavili viacero numerických prístupov na riešenie nastolenej úlohy. Všetky použité metódy vychádzali z myšlienky transformácie pôvodnej úlohy dynamického programovania na úlohu nelineárneho programovania väčšieho rozmeru.

Jadro prvej triedy úloh predstavovalo hľadanie vhodnej aproximácie nelinárnych vzťahov v modeli lineárnou, resp. kvadratickou funkciou. Získané úlohy v tvare MILP, resp. MIQCP bolo možné riešiť pomocou dostupných solverov, ktoré boli uspôsobené na riešenie úloh veľkého rozmeru. Narozdiel od práce [1], kde pri použití trojuholníkovej aproximácie funkcie výkonu uvedený prístup viedol k nájdeniu optimálneho riešenia s požadovanou presnosťou v danom časovom horizonte, v prípade minimalizácie odchýlky od diagramu sa tento postup ukázal ako príliš výpočtovo náročný. K nájdeniu dostatočne presného riešenia v danom časovom horizonte nepomohla ani zmena typu aproximácie funkcie výkonu.

Druhou testovanou triedou optimalizačných techník boli heuristické prístupy. Spomedzi nich sme sa zamerali na využitie genetických algoritmov a simulovaného žíhania. Tieto techniky nevyžadujú aproximáciu nelineárnych vzťahov. Nevýhodou sa však ukázal veľký počet riadiacich premenných. Nakoľko sú tieto techniky založené na prehľadávaní množiny prípustných riešení, veľký rozmer úlohy významne spomaľoval výpočet. Ani jedna z implementovaných alternatív nenašla optimálne riešenie v požadovanom čase. Genetické algoritmy, však ponúkajú možnosť paralelného výpočtu, tj. genetický algoritmus môže bežať na viacerých výpočtových jednotkách naraz s čím sa výrazne skráti čas algoritmu.

Paralelizácia však presahuje rozsah tejto práce.

Tieto neúspechy nás prinútili hľadať ďalšie alternatívy. Predchádzajúce numerické experimenty nás priviedli k návrhu vlastného prístupu: hybridu medzi presnou MILP formuláciou a heuristikami, ktorý sme nazvali lineárny iteratívny model. Narozdiel od všetkých predchádzajúcich formulácií navrhnutý postup dokázal nájsť dostatočne presné riešenie v časovom horizonte akceptovateľnej dĺžky pre potreby praxe.

Riadenie prevádzky Vážskej kaskády by bolo možné vylepšiť, ak by sa optimalizoval výkon pre jednotlivé generátory a nie pre celú elektráreň. Takáto modifikácia by si vyžiadala zaviesť nové premenné, čím by sa ešte zvýšil rozmer úlohy. Dôsledkom by bol dlhší výpočtový čas. Ďalším rozšírením by mohla byť optimalizácia na dlhšom horizonte. Takéto rozšírenie by obnášalo aj stochastický aspekt modelu vo forme predpovedí o prítokoch do nádrží.

Literatúra

[1] Jankola, J.: Optimalizácia vodných elektrární Vážskej kaskády v krátkodobom horizonte, Diplomová práca, Univerzita Komenského, Fakulta Matematiky, Fyziky a Informatiky, 2023

[2] Fajkus, A.: Optimalizácia prevádzky vodnej prečerpávacej elektrárne, Diplomová práca, Univerzita Komenského, Fakulta Matematiky, Fyziky a Informatiky, 2020

[3] Vester, M.: Optimalizácia prevádzky vodnej elektrárne v krátkodobom horizonte Diplomová práca, Univerzita Komenského, Fakulta Matematiky, Fyziky a Informatiky, 2020

[4] Eurostat.eu, dostupné na internete (22.04.2024): https://ec.europa.eu/eurostat

[5] Finardi, E. C., Scuzziato, M. R.: Hydro unit commitment and loading problem for day-ahead operation planning problem, International Journal of Electrical Power & Energy Systems, Vol. 44, No.1, 2013, dostupné na internete (30.4.2024): doi.org/10.1016/j.ijepes.2012.07.023

[6] Halická M., Brunovský, P., Jurča, P.: Optimálne riadenie, EPOS, Bratislava, 2009

[7] D'Ambrosio, C.: Application-oriented Mixed Integer Non-Linear Programming, Dizertačná práca, Universita Degli Studi Di Bologna, 2009

[8] D'Ambrosio, C., Lodi, A., Martello, S.: Piecewise linear approximation of functions of two variables in MILP models. Operations Research Letters, Vol. 38 No. 1, 2010, dostupné na internete (8.04.2024): https://doi.org/10.1016/j.orl.2009.09.005

[9] Gurobi QCP and SOCP optimizer overview, dostupné na internete (30.12.2023), https://www.gurobi.com/events/gurobi-qcp-and-socp-optimizer-overview/

[10] Dušička, P., Šulek, P.: Popis algoritmov hydromodelovania navrhnutých pre SW model prípravy prevádzky VE, Katedra hydrotechniky, Stavebná fakulta STU Bratislava, 2006

[11] Gurobi Optimization, LLC: *Gurobi Optimizer Reference Manual*, dostupné na internete (30.12.2023): www.gurobi.com

[12] ILOG CPLEX Optimization Studio: *Gurobi IBM ILOG CPLEX Optimization Studio documentation*, dostupné na internete (26.02.2024): https://www.ibm.com/docs/en

[13] Christophel, P.M.: Separation algorithms for cutting planes based on mixed integer row relaxations implementation and evaluation in the context of mixed integer programming solver software, Dizertačná práca, Universita Paderborn, 2009

[14] Anand, R., Aggarwal, D., Kumar, V.: A comparative analysis of optimization solvers, Journal of Statistics and Management Systems, Vol.20, No.4, 2017, dostupné na internete (30.04.2024): https://www.researchgate.net/publication/314750497_A_Comparative_Analysis_of_Optimization_Solvers

[15] Linderoth, J. T., Savelsbergh, M. W. P.: A Computational Study of Search Strategies for Mixed Integer Programming, *INFORMS Journal on Computing*, Vol 1, No. 2, 1999

[16] Kronqvist, J., Bernal, D. E., Lundell, A., Grossmann, I. E.: A review and comparison of solvers for convex MINLP, Optimization and Engineering, Vol.20, No. 2, 2018, dostupné na internete (27.03.2024): https://doi.org/10.1007/s11081-018-9411-8

[17] Brabazon, A., et al.: Natural Computing Algorithms, Springer, 2015.

[18] Koch, S., Wäscher, G.: A grouping genetic algorithm for the Order Batching Problem in distribution warehouses, Journal of Business Economics, Vol. 86, No. 1, 2016, dostupné na internete (8.5.2024): https://doi.org/10.1007/s11573-015-0789-x

[19] Libelli S. M, Alba P.: Adaptive mutation in genetic algorithms ,Soft computing, 4. vydanie, 2000

[20] Gad. A. F., PyGAD: An Intuitive Genetic Algorithm Python Library, dostupné na internete (29.12.2023): https://pygad.readthedocs.io/en/latest/

[21] James G., Witten D., Hastie T., Tibshirani R.: An Introduction to Statistical Learning with applications in R, Springer, 2023