VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ

DEPARTMENT OF TELECOMMUNICATIONS

WEBOVÁ APLIKACE PRO TVORBU PROFILU KYBERNETICKÉ BEZPEČNOSTI

WEB APPLICATION FOR CREATING A CYBER SECURITY PROFILE

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE — Michal Stejskal

AUTHOR

VEDOUCÍ PRÁCE — Ing. Petr Dzurenda, Ph.D.

SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Informační bezpečnost**

Ústav telekomunikací

Student: Michal Stejskal ***ID:*** 231282

Ročník: 3 ***Akademický rok:*** 2022/23

NÁZEV TÉMATU:

Webová aplikace pro tvorbu profilu kybernetické bezpečnosti

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se s vývojem moderních GUI webových aplikací, např. Progressive Web Apps, Flutter, AWS, Node.js, React, Vue atp. Studujte mechanismy vyhledávacích algoritmů. Seznamte se s Evropským rámcem dovedností v oblasti kybernetické bezpečnosti. Navrhněte a implementujte webový modul doporučující absolvování vhodných univerzitních studijních programů, tréningů a certifikací pro docílení požadovaného profilu kybernetické bezpečnosti. Webový modul integrujte do již existující aplikace [1]. Výstupem práce bude webová aplikace umožňující vytvářet profil kybernetické bezpečnosti s ohledem na definované vstupní priority, např. celková cena, délka studia atp.

DOPORUČENÁ LITERATURA:

[1] HAJNY, Jan, Marek SIKORA, Athanasios VASILEIOS GRAMMATOPOULOS a Fabio DI FRANCO, 2022. Adding European Cybersecurity Skills Framework into Curricula Designer. In Proceedings of the 17th International Conference on Availability, Reliability and Security (ARES '22).: Association for Computing Machinery. New York, NY, USA, 1–6.

[2] S. SKIENA, Steven, 2008. The Algorithm Design Manual. 2. London: Springer London. ISBN 978-1-84800-070-4.

Termín zadání: 6.2.2023 ***Termín odevzdání:*** 26.5.2023

Vedoucí práce: Ing. Petr Dzurenda, Ph.D.

doc. Ing. Jan Hajný, Ph.D.
předseda rady studijního programu

ABSTRAKT

Bakalářská práce se zabývá návrhem a implementací optimalizačního programu určeného pro nalezení nejvýhodnějších kombinací kurzů. Cílem práce bylo seznámit se s Evropským rámcem dovedností v oblasti kybernetické bezpečnosti (ECSF), optimalizačními algoritmy a projektem Cybersecurity Skills Alliance – A New Vision for Europe (REWIRE). Práce taktéž zahrnuje webový vývoj a práci s databází. Program je vyvinut v programovacích jazycích Python, PHP a JavaScript. Kritickými požadavky na funkci programu byla optimálnost řešení a doba ve které je řešení nalezeno. Program je spojený s webovým rozhraním, kde je možné vyhledávat optimální kombinace na základě vstupních podmínek.

KLÍČOVÁ SLOVA

Optimalizace, REWIRE, optimalizační algoritmus, ENISA, linearní programování, rámec dovedností v oblasti kybernetické bezpečnosti, React, Python, web, PHP

ABSTRACT

This Bachelor thesis describes design and implementation of optimalization program designed to find best combination of courses. The goal in this thesis was to get acquainted with European Cybersecurity Skills Framework (ECSF) in the area of cyber security, optimalization algorithms and with project Cybersecurity Skills Alliance – A New Vision for Europe (REWIRE). This thesis also includes web design and working with the database. Program is developed in the Python, PHP and Javascript programming languages. Between critical requirements of the program belongs optimality of found solution and time in which is the solution found. Program is connected with web interface, in which it is possible to search for optimal combinations based on input constraints.

KEYWORDS

Optimalization, REWIRE, optimization algorithm, ENISA, linear programming, cyber security skills framework, React, Python, web, PHP

STEJSKAL, Michal. *Webová aplikace pro tvorbu profilu kybernetické bezpečnosti*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2023, 47 s. Bakalářská práce. Vedoucí práce: Ing. Petr Dzurenda, Ph.D.

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	Michal Stejskal
VUT ID autora:	231282
Typ práce:	Bakalářská práce
Akademický rok:	2022/23
Téma závěrečné práce:	Webová aplikace pro tvorbu profilu kybernetické bezpečnosti

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářské práce panu Ing. Petru Dzurendovi, Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Obsah

Seznam obrázků

Úvod

Bakalářská práce se zabývá analýzou a vývojem optimalizačního modulu určeného pro webovou aplikaci na tvorbu profilu kybernetické bezpečnosti.

Vzhledem k rostoucí poptávce po profesionálech v oblasti kybernetické bezpečnosti na Evropském, ale i celosvětovém trhu vznikají nová pojmenování a termíny pro role a dovednosti v oblasti kybernetické bezpečnosti. Tato pojmenování jsou však často zcela odlišná. To může přinášet zmatky a nepochopení ohledně toho, co by se mělo v odborných kurzech vyučovat a co vlastně trh práce vyžaduje. Nedostatek pracovníků v oboru kybernetické bezpečnosti může do budoucna znamenat bezpečnostní riziko pro mnohé společnosti nebo vládní organizace.Evropskou agenturou pro bezpečnost sítí a informa (ENISA) se proto rozhodla tento problém vyřešit a vytvořila ECSF, který sjednocuje pojmy používané v této specializaci [1]. S ohledem na to vytvořil projekt REWIRE spojení mezi dovednostmi a znalostmi ECSF a kompetencemi v oblasti kybernetické bezpečnosti REWIRE. Tyto kompetence REWIRE jsou pak propojeny zpět s profily ENISA. Pomocí této metodiky je možné zmapovat stávající vysokoškolské osnovy, profesionální školení a certifikační kurzy ECSF. Projekt REWIRE dále navrhl webovou aplikaci Cybersecurity Profiler (CSP) [2], která může sloužit lidem v této oblasti, protože zahrnuje kurzy po celé Evropě a různé nástroje, které mohou pomoci s tvorbou osnovy studia, profesionální školení a certifikační kurzy a porozumění požadavkům pro plnění pracovních rolí.

První kapitola se věnuje evropskému rámci a projektu REWIRE. Podobě jednotlivých profilů, dovednostních skupin a znalostí. Ve druhé kapitole se krátce zaměřuje pozornost na výběr datových struktur pro práci s dostupnými daty. Dále je zpracován výběr potencionálních kandidátů na optimalizační algoritmus, který má za úkol vytvoření nejvhodnějších kombinací kurzů. Ke konci kapitoly jsou zmíněny balíčky pro lineární programování a také metody, kterými je optimalizace možné dosáhnout. Třetí kapitola shrnuje vývoj webových aplikací s rozdělením do dvou hlavních částí frontend a backend. V rámci obou částí jsou vysvětleny používané technologie a programovací jazyky, které jsou pro tuto práci relevantní. Čtvrtá kapitola se zabývá praktickou částí a to nejprve analýzou zadání ve které je dále zvolen vhodný algoritmus a modul, který bude provádět optimalizační operace. Poté je popsána architektura zpracovaného řešení, která zahrnuje vytvořenou webovou aplikaci a její propojení s optimalizačním modulem. Součástí praktického řešení bylo i srovnání několika řešících balíčku a výběr toho nejvhodnějšího pro implementaci.

1 Evropský rámec dovedností v oblasti kybernetické bezpečnosti

V dnešní době se Evropská unie potýká s nedostatkem expertů v oblasti kybernetické bezpečnosti. Vývoj oborů zaměřených na bezpečnost v prostředí internetu stále probíhá. Vzhledem k tomu, že se jedná o relativně nový obor je potřeba zaměřit se na sjednocení dostupných vzdělávacích kurzů v Evropské unii do jednoho celku. Celkem, který poskytne přehled nad dostupnými kurzy, certifikacemi a schopnostmi, které jsou pro dané zaměření potřeba. K tomuto slouží právě zmíněný ECSF [2], který byl navržen organizací ENISA. Tento nástroj, tedy poskytuje společné porozumění dovednostem, rolím a znalostem v oblasti kybernetické bezpečnosti, které jsou v současné době využívány v členských státech EU. Tímto je usilováno o vytváření základů, které v budoucnu umožní jednotné prostředí a posílí zaměstnatelnost v oborech souvisejících s kybernetickou bezpečností. V rámci je také zahrnuta tvorba ENISA profilů, které odrážejí reálně žádané pozice v oblasti kybernetické bezpečnosti.

Na rostoucí poptávku po odbornících v oboru kybernetické bezpečnosti reagují především univerzity a soukromé vzdělávací programy, což lze vyčíst z obrázku 1.2 ve kterém lze zřetelně sledovat nárůst vznikajících programů pro kybernetickou bezpečnost na vysokých školách. I pro ně tak mohou být nápomocné vzniklé termíny a vztahy mezi vlastnostmi a profily definovanými projekty ENISA a REWIRE.

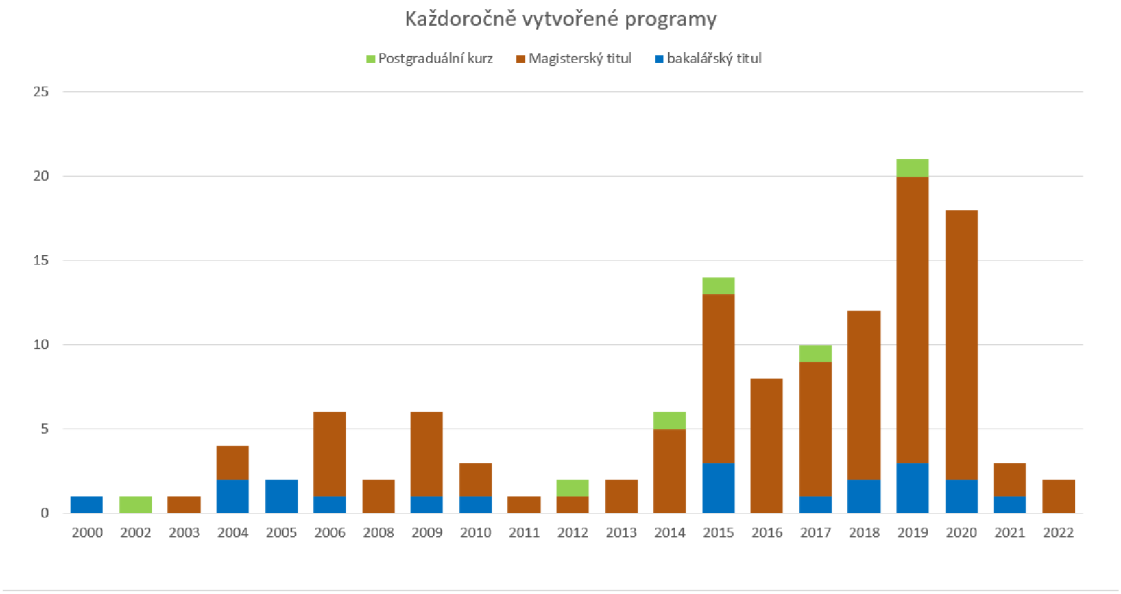
1.1 Projekt REWIRE

V rámci tohoto projektu je cílem sjednocení dostupných kurzů a jejich namapování k existujícím ENISA profilům. Rámec doposud rozlišuje dvanáct specializací, které jsou úzce propojeny s dostupnými vzdělávacími kurzy v Evropské unii. Projekt se zaměřuje na to jak definovat jaké dovednosti jednotlivé profily obsahují a zájemcům o zaměření v některé z těchto oblastí, tak poskytnout předlohu znalostí, které jsou pro profil významné [3]. Hlavním cílem je analýza dostupných osnov, školení, certifikací a jejich zařazení do dovednostních skupin a následně jejich uložení do společné databáze. Na evropském trhu, je dostupných několik databází ze kterých se dá tyto znalosti čerpat. Mezi použité v tomto projektu patří CONCORDIA, SPARTA a naopak jako nevyhovující byly vyhodnoceny ECHO a CyberSec4Europe [2]. Pro implementaci vybraných projektů bylo nutné sjednocení dat dostupných z jednotlivých databází.

1.1.1 REWIRE dovednostní skupiny

Projekt se zabývá i analýzou jednotlivých dovedností do nadřazených skupin, které byly definovány v rámci projektu REWIRE. Celkem bylo vytvořeno třicet jedna dovednostních skupin, které spadají do tří kategorií: Dovednosti v oblasti kybernetické bezpečnosti, Dovednosti v oblasti IT a Sociální dovednosti jak je vyobrazeno v obrázku 1.1. Dovednostní skupiny, které jsou v kategorií zahrnuty se skládají z jednotlivých dovedností a znalostí, které jsou rámcem definovány. Nadřazených třicet jedna skupin, tak zahrnuje 104 dovedností a 85 znalostí [3], které jsou rozloženy do odpovídajících skupin podle jejich relevantnosti. REWIRE se svojí jednoduchostí nabízí lehkou orientaci, díky pouhým dvanácti profilům. Jedná se však i o nevýhodu, kvůli své jednoduchosti není rámec dostatečně přesný, pokud jde o výběr konkrétnější profesní dráhy, navazujícího studia nebo možností rekvalifikace.

Obr. 1.1: Kategorizované dovednostní REWIRE skupin



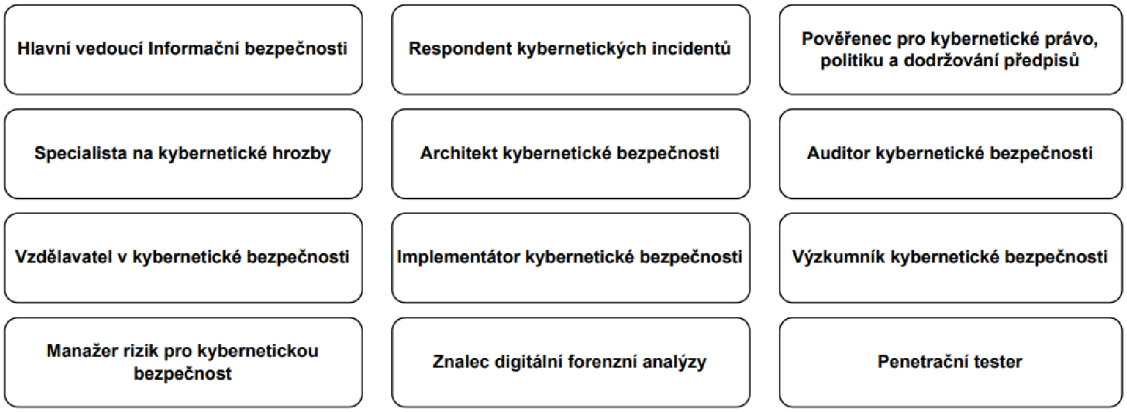
Obr. 1.2: Počet každoročně vytvořených programů

1.1.2 ENISA profily

Vytvoření dvanácti profilů napomáhá diverzifikaci v kybernetické bezpečnosti především zájemcům o zaměření na konkrétní část bezpečnosti z celého spektra. Rovněž nabízí snazší uchopení terminologie z pohledu náborových oddělení nebo zaměstnavatelů, kteří nemají v tomto oboru přehled o potřebných dovednostech pro naplnění potřeb jejich korporací. Každý profil zahrnuje popis poslání, alternativní názvy, klíčové znalosti a dovednosti a dále... Rámec byl navržen tak, aby umožnil srozumitelný náhled do kybernetické bezpečnosti a zároveň poskytl možnost flexibilního přizpůsobení podle potřeb. Každý profil je popsán tabulkou, která obsahuje: alternativní název pro tentýž profil, shrnutí účelu profilu, zdůvodnění cíle za kterým byl profil vytvořen, relevanci profilu, hlavní úkoly pro vykonavatele, klíčové dovednosti a znalosti a e-Kompetencí pokrytých profilem [1]. Profily byly přeloženy do českého jazyka na obrázku 1.3.

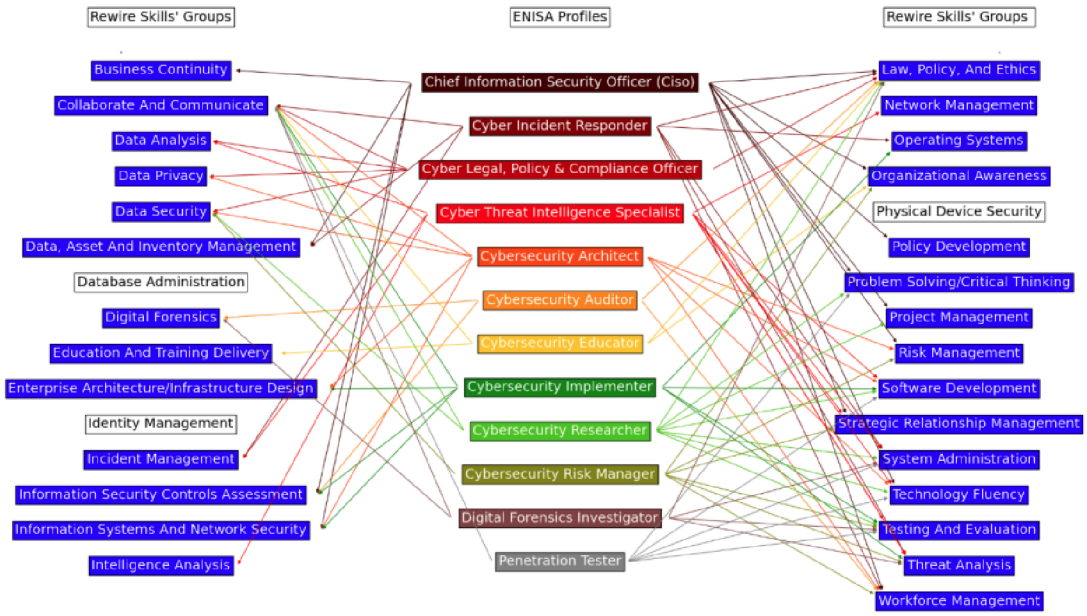
1.1.3 Mapování profilů s dovednostními skupinami

Z velkého množství dovedností a znalostí bylo zapotřebí vytvořit konkrétnější celky, které by profily naplňovaly, jak již bylo zmíněno, tak součet všech znalostí a dovedností od ECSF dosahuje téměř dvou set vlastností a bylo by nepřehledné je přímo přiřazovat k samotným profilům. Za účelem zjednodušení a namapování vlastností k profilům bylo projektem REWIRE vytvořeno mapování dovedností a znalostí na dovednostní skupiny, kterých projekt deklaroval celkem 31 a vytvořil tím užší celek



Obr. 1.3: Profily rolí v oblasti kybernetické bezpečnosti

obecnějších znalostí a dovedností v podobě těchto skupin. Menší počet skupin, tak umožnil větší přehled nad dovednostmi, které poté bylo možné mapovat k příslušným profilům kybernetické bezpečnosti. Jak je patrné z obrázku 1.4, díky menšímu počtu skupin, lze přehledněji mapovat vlastnosti pro profily. Díky vytvořeným vztahům profilů s dovednostními skupinami bylo možné analyzovat i vzájemné vztahy jednotlivých profilů, které jsou popsány v publikaci [3].



Obr. 1.4: Mapování ENISA profilů a REWIRE skupin [4]

2 Algoritmizace

Za algoritmus lze považovat postup, který je nutné vykonat k provedení specifického úkolu [5]. Tato procedura je schopná řešit problém po elementárních instancích. Z těchto instancí vzniká přesný postup řešení problému, který algoritmus definuje. Každý algoritmus vzniká za konkrétním účelem a je schopný pracovat s předem určeným typem vstupních dat. Tyto data zpracovává a na jejich základě poskytne požadovaný výstup neboli řešení.

2.1 Datové struktury

Funkčnost a efektivnost každého programu pracujícího s daty je ovlivněna výběrem datové struktury. Vhodný výběr datové struktury je právě pro algoritmy klíčový. Každý typ rozdílným způsobem data uchovává a přistupuje k nim. Proto je vhodné jejich implementaci uvážit již při tvorbě algoritmu.

Efektivita a rychlost jednotlivých typů datových struktur je porovnávána pomocí **asymptotické složitosti**. Například notace O(n) udává, že se stoupajícím počtem prvků bude lineárně růst i doba práce algoritmu. U časové složitosti O(1) není podstatný počet prvků, protože každý prvek je možné nalézt v konstantním čase. U prostorové složitosti udává potřebné paměťové nároky v závislosti na délce využívaných dat. Vhodné použití struktur v případě velkých objemů dat dokáže přímo ovlivnit dobu zpracování operací viz. porovnání datových struktur v tabulce 2.1. Datové struktury mohou být klasifikovány do dvou základních kategorií podle toho jestli jsou založené na polích nebo ukazatelích [5].

- **Souvisle alokované struktury** zabírají v paměti souvislý prostor a patří mezi ně pole, haldy, matice a hashovací tabulky.
- **Propojené datové struktury** umožňují data ukládat různě v paměti a přistupovat k nim pomocí tzv. ukazatelů, které jednotlivé datové části spojují v celek. Zahrnují listy, stromy a grafy [5].

2.1.1 Pole

Každý prvek zabírá v paměti stejný prostor. Prvkům jsou zároveň přiřazené indexy, které umožňují přistupování k prvkům v konstantním čase. Pole může nabývat jedné nebo více dimenzí (matice) [5]. Seřazené pole má rozdílné vlastnosti od pole s neseřazenými prvky, záleží tedy na pořadí ve kterém jsou prvky vkládány. Výhodou pole je úspora paměti, díky tomu, že nevyužívají ukazatele ukládají pouze data. Díky návaznosti uložených dat se pole vyhýbá nutnosti hledání prvků v paměti a tím zvyšuje rychlost procházení.

2.1.2 Slovníky

Tato datová struktura umožňuje přistupovat ke všem hodnotám na základě klíčů. Jedná se o abstraktní strukturu, protože se skládá s více datových typů. Klíč může být asociován s jednou hodnotou nebo množinou hodnot [5].

Vlastnosti slovníku závisí na implementaci datových struktur ze kterých se skládá. Například pokud se nevyžaduje od slovníku rychlé vkládání nebo mazání hodnot, ale za důležitější je považováno vyhledávání klíčů, tak je vhodné použít seřazené pole.

Tab. 2.1: Složitosti datových struktur

struktura	vložení prvku	odstranění	obsahuje prvek	obsahuje klíč
pole	O(1)	O(n)	O(n)	-
uspořádané pole	O(n)	O(n)	O($log(n)$)	-
slovník	O(1)	O(1)	O(n)	O(1)
uspořádaný slovník	O($log(n)$)	O($log(n)$)	O(n)	O($log(n)$)

2.2 Optimalizační algoritmy

Cílem optimalizace je nalezení optimálního řešení, pro které není možné vyjádřit matematický popis řešení problému. Optimalizace se zabývá nalezením požadovaného maxima nebo minima. Důležitým faktorem, při výběru algoritmu v této práci je procesorový čas, který program využije. Ten blízce souvisí s počtem operací, které musí být provedeny k nalezení globálního optima. V této sekci je popsáno několik možných metod přístupu k optimalizačním problémům.

2.2.1 Brute-force

Optimalizace za pomocí hrubé síly je z pohledu časové i prostorové složitosti tím nejhorším způsobem řešení optimalizace [5]. Při tomto způsobu optimalizace je nutné vypočítat všechna možná řešení a vybrat z nich to nejlepší. S přibývajícími dimenzemi se exponenciálně mění složitost výpočtu a je tedy možné tento přístup použít jen pro velmi nízký počet prvků. Navzdory tomu má řešení hrubou silou i výhody. Mezi ně patří jednoduchá implementace nebo nalezení více možných řešení a především jistota nalezení nejvhodnějšího řešení pokud existuje. Dle rovnice (2.1) lze jasně určit počet vytvořených kombinací. Berme například, že máme dispozici 50 prvků n a hledáme kombinace o třech prvcích r. Po dosazení do rovnice vychází C rovno 19 600 kombinacím. Takový počet kombinací je nutno během výpočtu uchovávat v

paměti a poté z nich vybrat nejlepší řešení.

$$C(n,r) = \frac{n!}{r!(n-r)!} \tag{2.1}$$

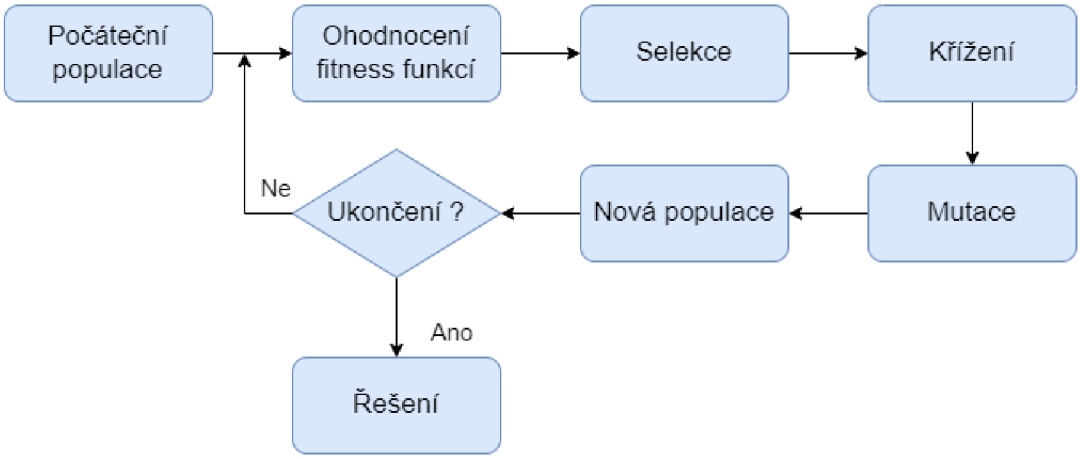
2.2.2 Genetické algoritmy

Čerpají inspiraci z přirozené biologické evoluce [5]. Mezi základní prvky algoritmu patří:

1. **Populace** uskupení jedinců (chromozomů), kteří reprezentují potencionální kandidáty pro řešení. Chromozomy se skládají z genů, které popisují vlastnosti jedince, které jsou pro optimalizaci relevantní. Po určení vlastností chromozomů je jimi naplněna populace. Ta může být vytvořena náhodnými hodnotami nebo na základě nějaké znalosti a také se může jednat o populaci z minulé generace, která již prošla procesem evoluce.
2. **Fitness funkce** určuje kvalitu každého vytvořeného řešení. Na tomto kvantifikovaném základě je algoritmus schopný určovat, která řešení vykazují potenciál a která mají zaniknout. Chromozom se dá vyčíslit dle různých kritérií. Mezi nejčastější patří rozdíl mezi požadovaným a aktuálním řešením, rozdíl řešení od referenční hodnoty, dosažení přesnosti nebo délka doby výpočtu. Kvalita řešení bývají vyjádřena v celých číslech.
3. **Selekce** je přístup jakým zajistit posun ve výběru jedinců. Typ výběru jakým je populace filtrována rozhoduje o rozmanitosti a rychlosti gradování k optimálnímu řešení. Operátory selekce:
   - **Ruletový výběr** – Pravděpodobnost se kterou je jedinec vybrán do další populace se odvíjí od jeho celkové kvality. Ta je reprezentována i pro zbytek populace pomyslnou výsečí na ruletě. Tudíž čím kvalitnější jedinec, tím větší prostor na ruletě zabírá a má vyšší šanci na postup. Tento operátor způsobuje v pozdějších iteracích selekce zvýšenou podobnost genů v populaci, což ztěžuje a někdy i znemožňuje nalezení optimálního řešení, protože populace ztrácí rozmanitost. Tento problém je možné řešit způsobem tzv. "ranking selection". Kdy místo reprezentace kvality přímo na ruletě jsou jedinci řazeni vzestupně se stejnými rozdíly mezi jedinci, čím se eliminuje dominance jednotlivců.
   - **Turnajový výběr** – Připomíná svojí funkcí středověké turnaje, kde mezi sebou jedinci soupeří. Většinou dochází k porovnávání dvou jedinců.
4. **Genetické operátory** mají za cíl upravovat geny chromozomu. Slouží k pozměnění populace, tak aby nedocházelo ke ztrátě diverzity a snížení rychlosti optimalizace řešení.
   - **Elitářství** – Nejlepší jedinec je automaticky zařazen do další populace a je zachována podoba jeho genů.

- **Křížení** – Modifikace vlastností chromozomů tím, že se ovlivňují dva či více chromozomů . Křížení lze provádět v jednom nebo více bodech. Výstupem jsou potomci, kteří mohou fitness funkcí překonat své rodiče a nahradit je tak v další generaci.
- **Mutace** – Používá se náhodně a velmi zřídka. Slouží především ke zvýšení rozmanitosti v populaci. Aplikace má význam především při malé populaci.[6]

Funkci algoritmu lze popsat všeobecně dle diagramu na obrázku 2.1. Nejprve je nutné problém reprezentovat, tak aby bylo jasné jaká bude podoba chromozomů a jejich genů. Poté je na základě znalosti nebo náhody vytvořena počáteční populace. Ta je následně ohodnocena fitness funkcí. V dalším kroku je vybrána nadějná část populace, která se zařadí do další generace. S vybranými jednotlivci se provádí křížení a mnohdy i mutace, pokud to problém umožňuje a nabízí se možné zlepšení výsledků. Poté co je nová populace vytvořena dojde k vyhodnocení fitness funkce jedinců. Pokud je hodnota některého jedince v tolerované odchylce s optimem dojde k ukončení algoritmu a chromozom je brán jako nalezené řešení. V opačném případě, kdy není nalezeno řešení je celý cyklus opakován s nově vytvořenou populací. K ukončení může rovněž dojít překročením limitu určeného pro běh algoritmu a je tak vráceno nejvhodnější řešení z populace, ačkoliv nemusí být optimální.



Obr. 2.1: Diagram genetického algoritmu

2.2.3 Lineární programování

Lineární programování (LP), je matematickou metodou pro optimalizaci účelové funkce za určitých omezujících podmínek. Problém nalezení řešení je možné řešit buďto maximalizací nebo minimalizací. Příkladem může být maximalizace zisku

nebo minimalizace nákladů [7]. Účelová funkce udává co je předmětem maximalizace nebo minimalizace. Omezující podmínky udávají hranice, po kterých se během výpočtu lze pohybovat.

Na rozdíl od genetického algoritmu (GA) skončí LP vždy, až při nalezení nejlepšího řešení, pokud nějaké existuje. GA mohou být ukončeny po definovaném počtu iterací nebo po uplynutí stanoveného času. Nemusí tak vždy dosáhnout globálního optima, což může být způsobeno i špatným navržením jednotlivých prvků algoritmu. Například nevhodným způsobem křížení nebo nevhodným výběrem jedinců. Pro úlohy binárního charakteru na které je aplikovatelná účelová funkce a omezovací podmínky se lineární programování jeví jako nejoptimálnější algoritmus. Výběr však závisí na více faktorech, jako je velikost množiny prvků, jejich podoba a množství omezovacích podmínek. Aplikace algoritmu se tak mnohdy může zdát být optimální avšak, až do chvíle implementace jiné potentní metody.

Simplexová metoda

Algoritmus, který slouží pro řešení lineárního programování, což je matematický problém optimalizace za předpokladu lineárních omezujících podmínek [5]. Cílem je najít optimální hodnotu cílové funkce při splnění těchto omezujících podmínek.

V průběhu algoritmu se hodnota cílové funkce zlepšuje pohybem po hranicích, které jsou pomocí aplikovaných omezení nastoleny, dokud není dosaženo optimálního řešení. Pokud je problém neomezený, lze jej rozpoznat na základě neexistujícího výběru výstupní proměnné. Algoritmus pozná, že se nachází v optimálním bodě, pokud již není možné pohybovat se dál po hranici, aniž by se aktuální hodnota účelové funkce nezhoršila. Existují také modifikace a vylepšení simplexové metody, které umožňují rychlejší konvergenci a řešení složitějších problémů, které není možné řešit pomocí klasické implementace této metody.

Interior point metody

Interior point metody jsou matematické algoritmy používané v optimalizačních problémech pro nalezení optimálního řešení uvnitř konvexního prostoru [8]. Tyto metody se pokoušejí nalézt řešení uvnitř tohoto prostoru místo toho aby se pohybovali po jeho vrcholech jako to dělá například simplexová metoda. Interior point metody pracují s tzv. "penalizačními"funkcemi, které kombinují cílovou funkci s omezeními a přidávají penalizační členy pro omezení, která jsou porušena. Tyto penalizační členy trestají odchylku od omezení a tím motivují algoritmus hledat řešení, které minimalizuje tuto odchylku. V průběhu let bylo vyvinuto několik variant interior point metod, které se liší v detailní implementaci a optimalizaci výpočetního procesu.

Interior point metody mají několik výhod oproti jiným metodám, jako je například simplexová metoda. Patří sem schopnost řešit problémy s velkým počtem proměnných a omezení, efektivnější výpočetní čas a schopnost pracovat s problémy, které nemají jednoznačné optimální řešení.

Mixed-Integer Programming (MIP)

Je matematická optimalizační metoda používaná k řešení problémů, které zahrnují kombinaci spojitých proměnných (reálných čísel) a celočíselných proměnných (celých čísel) [9]. V MIP jsou některé proměnné omezeny na celočíselné hodnoty a není možné je převádět na spojité hodnoty, například při přiřazování úkolů není možné jeden stroj rozdělit a používat současně na více prací. Je-li tedy vyžadován výsledek v celých číslech je hledáno v okolí spojitých proměnných a jako výsledek mohou sloužit nejbližší celočíselné hodnoty. Cílem je tedy nalézt takovou kombinaci celočíselných hodnot, aby byly naplněny podmínky a řešení bylo optimální. MIP problémy mohou být vyřešeny například pomocí různých metod lineárního programování.

2.3 Knihovny pro LP

Knihovny nebo také moduly, které jsou prostřednictvím jazyku python k dispozici poskytují pomyslnou kostru pro řešení problému. Liší se použitými přístupy při zpracovávání vstupních a výstupních dat. Nabízejí také rozdílné funkce a podporují různé řešící balíčky tzv. „solvery“. Dalšími rozdíly je například velikost komunity, která moduly využívá a podpora vývojářů. Data v grafech 2.2 a 2.3 jsou čerpána ze studie na analýzu komerčních a „open source“ řešících balíčků viz [10].

Pulp

Knihovna Pulp [11] umožňuje řešení optimalizačních problémů a to včetně těch binárních. Oproti dalším dostupným knihovnám vyčnívá především rychlým načtením modulu a jednoduchostí implementace. Modul je schopný využívat celou řadu řešících balíčků. Ve výchozím stavu využívá Coin Branch and Cut solver dále jen CBC, která je dále popsána v samostatné podkapitole 2.4.1. Jako další řešící balíček s otevřenou licencí umožňuje využívat GLPK. Podporuje i komerční solvery a to především GUROBI nebo CPLEX.

Scipy

Jedná se o nástavbu nad matematickým modulem Numpy, který umožňuje především práci s maticemi, vektory a vícerozměrnými poli. Pro reprezentování problému

v modulu Scipy je mnohdy zapotřebí upravit vstupní datové struktury a rovněž neumožňuje řešení binárních problémů. Neumožňuje totiž práci se slovníky a je nutné využívat pro vstupní proměnné maximálně dvourozměrné pole. Umožňuje grafické zobrazení problému a je dostupný pod volnou licencí. Scipy podporuje méně známé a výkonné balíčky. Bohužel nepodporuje solvery jako GUROBI nebo CPLEX. Prostřednictvím modulu Scipy je možné využívat nedávno vypuštěný HiGHS solver, který svým komerčním rivalům úspěšně konkuruje [12].

Pyomo

Pyomo je rozsáhlejší, více podporovanou knihovnou a nabízí také více funkcí, jako nelineární optimalizaci. Podporuje objektově orientované modely problému. Umožňuje práci s daty z databází, slovníků a tabulek. Seznam podporovaných solverů je podobný jako u Pulp s tím, že umožňuje jejich asynchronní spouštění, čímž lze dosáhnout souběžného chodu řešení problémů [13].

2.4 Řešící balíčky pro LP

Data prezentovaná na obrázcích 2.2, 2.3 pro srovnání popisovaných balíčků jsou čerpána ze studie [14], která zkoumá aplikaci jak komerčních tak bezplatných solverů. Výsledky jednotlivých balíčků se mohou výrazně lišit podle problému, který mají zpracovávat. Obrázek 2.2 zeleně zobrazuje komerční řešící balíčky, které dosahují vyššího počtu vyřešených typů problémů na rozdíl od modrých nekomerčních balíčků. Důležitým faktorem těchto balíčků je rychlost, jak je vidět z obrázku 2.3 rozdíly mezi komerčními a nekomerčními balíčky je drastický, záleží však na konkrétní aplikaci a pro méně náročné problémy nebo aplikace nevyžadující výsledky v reálném čase mohou být nekomerční balíčky dostačující.

2.4.1 COIN Branch and Cut solver

Je balíček pro řešení lineárních problémů dostupný pod volnou licencí. Slouží jako kontrolní funkce pro výstup simplexové metody [15]. V případě nepřijatelného výstupu metoda cílí na upravení vstupních parametrů tak, aby bylo dosaženo při dalším výpočtu optimálního řešení.
Funkce je následující:

- Krok 1. (Hranice) Využívá se jednoduchá nebo duální simplexová metoda, výběr záleží na počtu omezení oproti počtu proměnných. Pokud jsou po vypočtení simplexové tabulky hodnoty celočíselné a proměnné jsou zadány jako celá čísla je přijato nalezené řešení jako optimální. V opačném případě není nalezeno správné řešení a pokračuje se dalším krokem.

- Krok 2. (Větvení) Jestliže hodnota proměnné není celé číslo, tak následuje vytvoření dolní a horní hranice pro spojitou proměnnou. Pro snazší pochopení berme hodnotu 0,536. Proměnná musí nabývat binárních hodnot 0 nebo 1. Vytvořením těchto hranic jsou proměnné přiřazeny dvě varianty hodnot, které může nabývat. Hodnoty jsou uloženy do stromové struktury a obě hranice jsou postupně aplikovány jako omezení do simplexové tabulky. Je vytvořena relaxace a po vypočtení obou variant je vybrána ta s lepším celočíselným výsledkem.

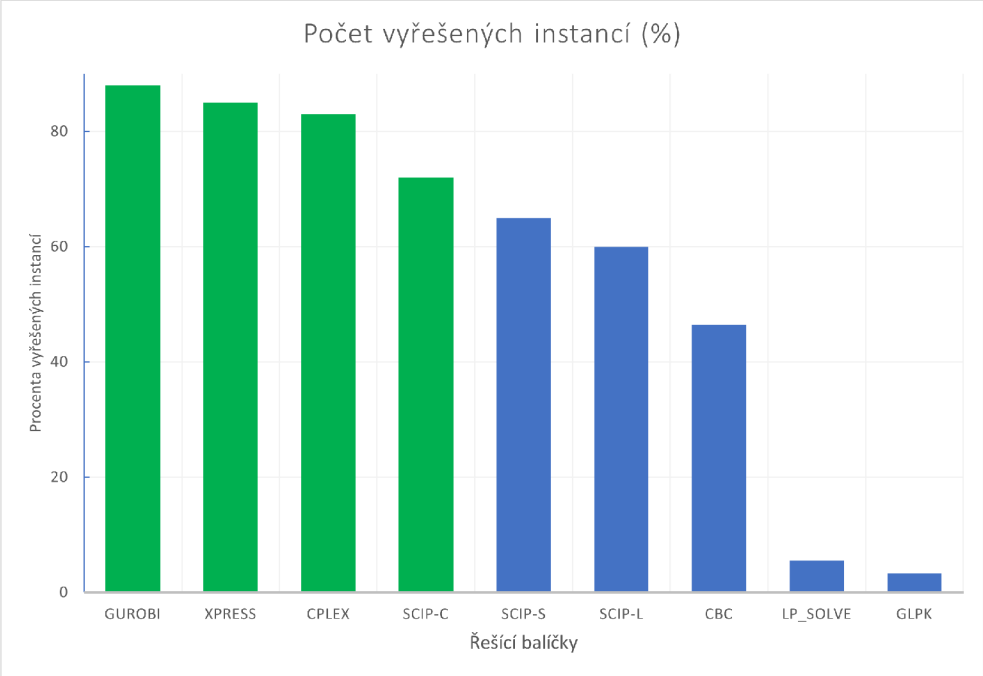
Pokud alespoň jeden z výsledků obsahuje opět spojitou proměnnou je druhý krok opakován na této proměnné. Po získání nejlepšího výsledku je program ukončen. Pokud by došlo k neustálému opakování tohoto kroku tzn. není možné získat binární hodnotu, bude program ukončen po překročení určitého počtu iterací a bude vybráno celočíselné řešení pokud nějaké existuje [15]. Program, při průchodu binárním stromem v případě nenalezení celočíselného řešení, využívá v základním nastavení rekurzivní volání a výpočty možných variant, tak probíhají paralelně.

2.4.2 GLPK

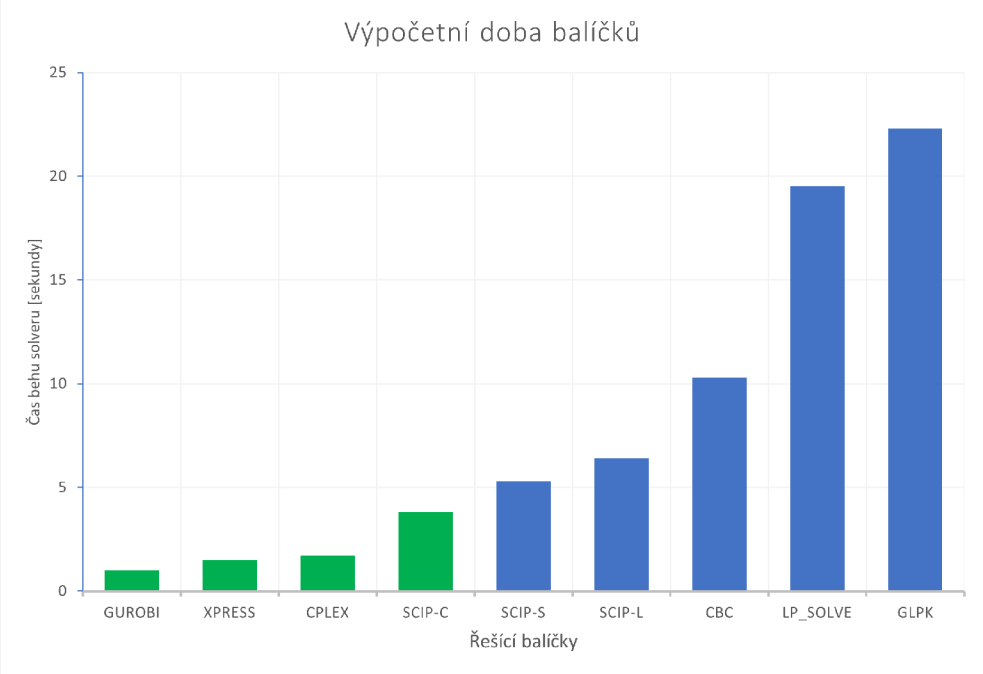
Balíček slouží pro řešení problémů lineárního a mixed-integer programování. Využívá simplexovou a interior point metodu pro výpočet neceločíselných problémů. Na výpočet mixed-integer a celočíselných problémů se využívá kombinace metod branch and bound a cutting plane, které jsou popsány v podkapitole 2.4.1.

2.4.3 HiGHS

Je vytvořen na základu „high performance dual revised simplex implementation (HSOL)“ a „parallel variant (PAMI)“. To znamená, že tento balíček umožňuje paralelní řešení simplexových úloh a variant výsledků. Jako většina solverů obsahuje MIP. To znamená, že se při výpočtu vyskytují proměnné, které nabývají celočíselných i spojitých hodnot [16].



Obr. 2.2: Počet vyřešených instancí v grafu



Obr. 2.3: Rychlost nalezení řešení jednotlivých balíčků v grafu

3 Vývoj Webové aplikace

Tato část se zabývá stručným popisem vývoje webových aplikací. Jaké jsou používány technologie, programovací jazyky a propojení mezi nimi. Kapitola je rozdělena do dvou částí. V první části je popsán vývoj tzv. „frontend“ části aplikace, která se zabývá prezentací informací klientům [17]. Důraz je kladen na vývojové prostředí React.js. Druhá část popisuje vývoj webové aplikace ze strany serveru, tedy akce probíhající na pozadí tzv. „backend“, jako je práce s daty a vyhodnocování uživatelských vstupů.

3.1 Frontend webových aplikací

Frontend je část webového vývoje, která se zaměřuje na tvorbu uživatelského rozhraní webové stránky nebo aplikace [18]. Jedná se o viditelnou část webu, se kterou uživatel přichází do kontaktu při prohlížení stránky v prohlížeči. Skládá se z několika základních technologií, z nichž nejvýznamnější jsou Hyper Text Markup Language, dále jen HTML, CSS (Cascading Style Sheets) a JavaScript.

3.1.1 HTML a CSS

HTML, je značkovací jazyk jehož úkolem je definovat strukturu a obsah stránky pomocí různých značek a atributů, a tím vytvořit ucelený dokument pro prohlížeč.

Značky jsou nejen prostředkem pro zobrazení textu, obrázků a odkazů, ale také umožňují vložení různých prvků na stránku, jako jsou tlačítka, formuláře, videa, interaktivní prvky a mnoho dalšího. Atributy pak slouží k přidání dalších informací ke značkám, například popisu obrázků nebo specifikaci velikosti a barvy textu.

CSS je stylovací jazyk používaný pro vizuální úpravu webových stránek. Jeho hlavním úkolem je oddělit prezentaci webového dokumentu (jako jsou barvy, velikosti písma, pozice prvků atd.) od samotného obsahu. To umožňuje větší flexibilitu a kontrolu nad vzhledem stránky [19]. CSS se skládá z pravidel, která definují, jakým způsobem se mají určité elementy na stránce zobrazovat. Každé pravidlo se skládá z selektoru (který určuje, které elementy se mají ovlivnit) a deklarací (které určují, jak se mají tyto elementy zobrazovat). HTML umožňuje vytvořit strukturu a obsah webových stránek, ale pro vizuální úpravu se používá jazyk CSS.

3.1.2 React framework

React je open-source knihovna pro tvorbu uživatelských rozhraní webových aplikací, která je napsána v jazyce JavaScript. React umožňuje snadno vytvářet interaktivní

a dynamické webové stránky, které jsou rychlé a efektivní. Abychom ale mohli plně pochopit, Jeho hlavním principem je tzv. "komponentní architektura", která umožňuje vytvářet komponenty, což jsou znovupoužitelné části kódu, které lze snadno spojovat a skládat do složitějších celků. Využívá vlastního virtuálního DOM (Document Object Model), což je abstraktní reprezentace struktury webové stránky v paměti počítače. Virtuální DOM Reactu je velmi rychlý a efektivní, což umožňuje minimalizovat počet nutných aktualizací a optimalizovat výkon aplikace. React také umožňuje jednosměrnou vazbu dat, tzv. "one-way data binding", což umožňuje snadnou správu dat v aplikaci. Knihovna se stala velmi populární mezi vývojáři, protože nabízí rychlý a efektivní způsob tvorby moderních webových aplikací, které jsou snadno udržovatelné a rozšiřitelné. Její použití se stává standardem v mnoha moderních webových projektech [20].

JavaScript je programovací jazyk, který umožňuje tvorbu dynamických webových stránek a interaktivních webových aplikací. Původně byl vytvořen pro použití v prohlížečích, ale dnes se používá i na serverové straně. S jazykem JavaScript se setkáváme na každém kroku při prohlížení webu, protože je součástí mnoha webových stránek. Používá se například pro generování odpovědi na stisknutí tlačítka, vyplňování formulářů, dynamickému ukládání proměnných a podobně [17].

3.2 Backend webových aplikací

Backend je vývoj serverové části webové aplikace. Tato část umožňuje interakci s databázemi, zpracování dat a poskytuje datové rozhraní pro komunikaci s frontendem. Backend je tedy část webové aplikace, která je skrytá před očima uživatele, ale zajišťuje všechny klíčové funkce pro správné fungování webové aplikace. Backend se skládá z několika základních prvků, jako jsou databáze, aplikační servery, API (Application Programming Interface), bezpečnostní mechanismy a další nástroje pro zpracování dat a poskytování informací [18]. Bezpečnostní mechanismy jsou klíčovou součástí backendu, protože chrání data a aplikaci před neoprávněným přístupem a útoky. Mezi bezpečnostní mechanismy patří šifrování dat, autentizace a autorizace.

3.2.1 PHP

Hypertext Preprocessor, dále jen PHP je jedním z nejčastěji používaných jazyků pro vývoj backendu webových aplikací. Backend zpracovává a ukládá data a poskytuje je frontendu - tedy webovému rozhraní, které vidí uživatelé. Jednou z hlavních funkcí backendu je také interakce s databázemi a správa uživatelských účtů [21].

V kontextu webu, PHP umožňuje propojení frontendu a backendu. Frontend, tedy webové stránky, jsou vytvářeny pomocí HTML, CSS a JavaScriptu a interak-

tivita stránek, jako například validace formulářů, odesílání dat, a načítání dat bez nutnosti stránku znovu načítat, je řízena pomocí backendu, který pracuje v prostředí serveru. To umožňuje webovým stránkám interagovat s uživateli a ukládat a zobrazovat data, která jsou potřebná pro běh aplikace.

PHP je často používán jako backendový jazyk pro vytváření webových aplikací a dynamických stránek. Jeho syntaxe je snadno čitelná a srozumitelná a má mnoho vlastních funkcí a knihoven pro práci s daty a databázemi. Jeho interpretovaná povaha také umožňuje rychlou a efektivní práci s daty a uživatelskými účty. PHP také podporuje různé frameworky, jako například Laravel, Symfony nebo CodeIgniter, které poskytují různé nástroje a knihovny pro vývoj backendu webových aplikací. Tyto frameworky zvyšují efektivitu a rychlost vývoje backendu, což umožňuje vývojářům rychle vytvářet webové aplikace s minimálním množstvím kódu [17].

Celkově lze říci, že PHP je klíčovým jazykem pro propojení frontendu a backendu webových stránek a aplikací. Jeho snadná syntaxe a mnoho vlastních funkcí a knihoven umožňují rychlou a efektivní práci s daty a databázemi, což umožňuje vývojářům rychle vytvářet webové aplikace s minimálním množstvím kódu.

3.2.2 Databáze

Databáze SQL (Structured Query Language) jsou technologií, která umožňuje ukládání, správu a manipulaci s daty. SQL databáze se skládají ze souboru tabulek, které uchovávají data ve strukturované formě. Tyto tabulky mohou být propojeny pomocí vztahů, které umožňují komplexní manipulaci s daty.

SQL databáze jsou často používány jako backendová technologie pro webové aplikace a stránky. Backendová část aplikace se stará o ukládání a manipulaci s daty, a tedy i s SQL databázemi. Frontendová část aplikace je pak zodpovědná za prezentaci těchto dat uživatelům pomocí webového rozhraní.

Backendové technologie, jako je PHP, jsou často používány pro komunikaci mezi frontendem a SQL databázemi. Umožňují zpracovávání uživatelských požadavků a poskytování dat z SQL databází do frontendu, a také zpracovávání uživatelských interakcí a ukládání dat zpět do databází [21]. Vývojáři backendu často využívají SQL databáze jako úložiště pro ukládání uživatelských údajů, jako jsou přihlašovací údaje, profily a transakční údaje. Tyto údaje jsou ukládány v tabulkách s definovanými vztahy a jsou přístupné pomocí SQL dotazů, které umožňují rychlé a efektivní vyhledávání a manipulaci s daty.

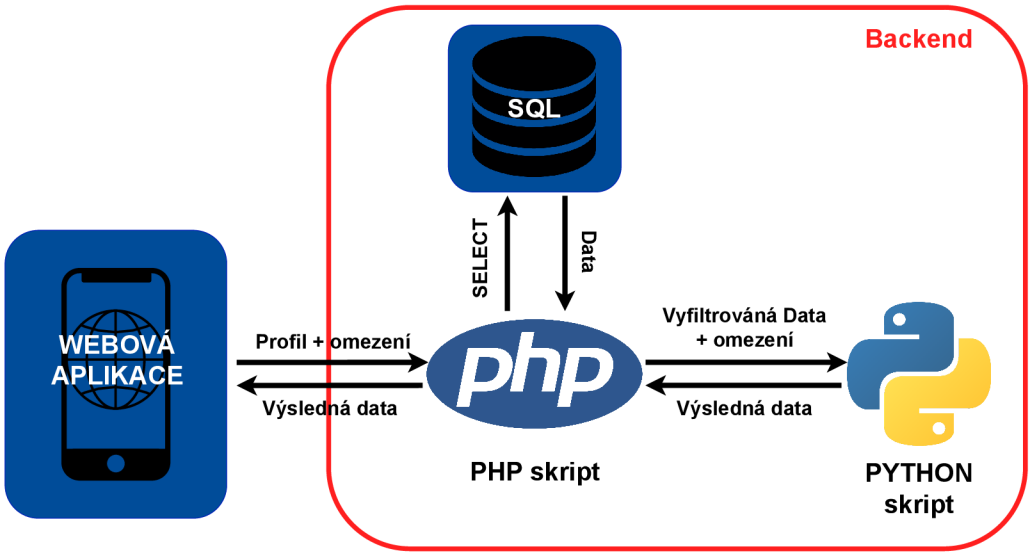
Celkově lze tedy říci, že SQL databáze jsou důležitou součástí backendové části webových aplikací a stránek. Jsou používány pro ukládání, správu a manipulaci s daty a jsou často využívány jako úložiště pro uživatelské údaje. Backendové technologie, jako je PHP, umožňují propojení frontendu a backendu a umožňují inter-

aktivitu webových stránek s uživateli a ukládání a zobrazování dat potřebných pro běh aplikace. Databáze SQL se staly kritickou součástí vývoje moderních aplikací. Backendové aplikace jsou nyní schopny poskytovat sofistikované databázové řešení, která umožňují uživatelům snadno přistupovat k datům a manipulovat s nimi. Frontendové aplikace pak mohou využívat tato data a poskytovat uživatelům snadno ovladatelné uživatelské rozhraní pro manipulaci s daty.

4 Vyhledávací modul pro aplikaci Cyber Security Profiler (CSP)

4.1 Analýza zadání

Je nutné analyzovat jakým způsobem, která data budou čerpána a filtrována. Jak je vidět z obrázku 4.1 na popředí se nachází webová aplikace, která nabízí uživateli rozhraní přes něž dochází ke komunikaci s tzv. „backendem“. Uživatel si zvolí v aplikaci profil a určí omezující podmínky. Tyto informace zachytává skript napsaný v jazyce PHP. Následně jsou na základě vstupu uživatele vybrána data z databáze a předána prostřednictvím souboru JSON volanému python skriptu. Python po zpracování vstupních dat nalezne optimální řešení a výsledek předá zpět do PHP, které posílá do webové aplikace data k prezentaci pro uživatele.



Obr. 4.1: Propojení backendu s webovou aplikací

Data jsou v tomto případě tréningy, které jsou pro uživatele vybírány, tak aby nejlépe vyhovovaly jejich kritériím. Formát dostupných dat není plně jednotný a bylo nutné provést úpravy pro některé klíčové hodnoty jako je cena nebo doba trvání tréningu. Pro získávání dalších tréningů bude v budoucnosti k dispozici webový formulář, kterým budou uchazeči žádat o zařazení tréningu do databáze. Tento krok umožní přeskočení kontrolní části zadávaného vstupu při schvalování tréningu a také by zajistil kompatibilitu s programem, který data zpracovává. Tato pevná omezení nemusejí žadatelům dávat dostatečný prostor ve kterém by se mohli více vyjádřit k

bodům, které nejsou zařazeny v šabloně nebo nejsou jednoznačně definovatelné. K tomuto účelu by mohly sloužit komentáře, které by poskytly náhled do podrobností tréningů. Na výpisu 4.1 je nevyplněný formulář uložený ve formátu JSON, který umožňuje konstantní přístup ke každému prvku, který je ve slovníku uložen pod klíčovou hodnotou tj. hodnota ohraničená v uvozovkách. Za dvojtečkou se poté nacházejí hodnoty přiřazené ke klíči. Hodnoty bývají v podobě textového řetězce nebo také celočíselné hodnoty u id,price a duration. V hranatých závorkách se nachází více hodnot v tzv. poli.

Výpis 4.1: Ukázka struktury ukládání tréningů do slovníku ve formátu JSON

```
                                                                    1
{ "id":                                                             2
  "organizer":                                                      3
  "short_description":                                              4
  "link":                                                           5
  "language":                                                       6
  "type_format":                                                    7
  "country":                                                        8
  "timing":                                                         9
  "course_dates":                                                   10
  "duration":                                                       11
  "content_type":                                                   12
  "price":                                                          13
  "prerequisites":                                                  14
  "includes_exams_for_certification":                               15
  "skills_group": []                                                16
  "specific_skill": []                                              17
  "specific_knowledge": []                                          18
  "other_skills": []                                                19
  "other_knowledge": []                                             20
}                                                                   21
```

Po prvotní filtraci je zapotřebí tato data kombinovat tak, aby kurzy docílily shody s vybraným profilem. Kurzy jsou tak kombinovány na základě ceny, doby trvání a také podle toho kolik kurzů v součtu naplňuje zvolený profil/y. Po vytvoření kombinací jsou podmínky pozměněny a opět jsou hledány kombinace. Po získání dostatečného počtu možností jsou varianty seřazeny a odeslány k prezentování uživateli. Celá tato sekce probíhá v prostředí jazyka python.

4.1.1 Volba algoritmu

V praktické části je implementováno pro řešení optimalizace lineární programování. Konkrétně je využita simplexová metoda.

LP a konkrétně simplex je vybrán jako nejvhodnější způsob, díky nízkému počtu iterací potřebných k nalezení optimálního výsledku. Tyto iterace obsahují jednoduché kroky ve kterých dochází pouze k výpočtům v jednom řádku a sloupci simplexové tabulky. každé iteraci je vybrán nejlepší proměnná a ta je jediná ve sloupci a řádku uchována. Všechny ostatní hodnoty jsou vynulovány, což změní parametry i ostatním proměnným. Tento způsob perfektně vyvažuje ještě nevyužité proměnné a směřuje řešení ke globálnímu optimu. Algoritmus se tak nemůže zaseknout v lokálním optimu.

Existuje mnoho algoritmů pro LP a některé z nich nabízejí i lepší výsledky z principu funkce algoritmu, avšak záleží na typu řešeného problému a v praxi je simplexová metoda osvědčená a implementovaná v řadě řešících algoritmů, mnohdy právě v kombinaci s jinými algoritmy, které efektivněji řeší specifické problémy.

Možným kandidátem bylo i zformulování problému pomocí GA, ale od tohoto přístupu bylo upuštěno, protože by bylo nutné vytvořit kompletní algoritmus, který by s vysokou pravděpodobností nebyl schopný konkurovat existujícím balíčkům pro LP. Jako další důvod by se dala uvést nejistota výkonosti balíčku, protože GA jsou vhodnější spíše k hledání řešení s nejasným výsledkem a globální optimum není zaručeno. S tím se váže i rozdílný průběh hledání řešení díky náhodnosti, která může směřovat řešení po každém spuštění jiným směrem. Záleží tedy na tom jakým způsobem by byl celý algoritmus navržený, což značně komplikuje predikci efektivity v porovnání s definováním problému pro již existující algoritmus. Výběr byl tedy zúžen na algoritmy LP a to především na ty nejpoužívanější v praxi a to jsou interior point metody a simplexová metoda. Není jasně dané jaká z metod je efektivnější a záleží především na který problém jsou metody aplikovány. Při výběru řešícího balíčku v další sekci je proto kladen důraz na celkovou efektivitu jak metody optimalizační, tak metody udržující hodnoty v celočíselné binární podobě nebo-li nabízející relaxaci problému.

4.1.2 Výběr řešícího balíčku pro LP

Pro vývojové prostředí v jazyce python byla k dosažení lineární optimalizace zvolena knihovna Pulp. Důvodem zvolení knihovny Pulp jsou dobře formátované metody a přehledné vytváření omezení a definování účelové funkce. Pulp také využívá široká komunita uživatelů a část z nich přispívá ke zlepšení funkcí a optimalizaci chodu modulu. Obrovskou výhodou knihovny je fakt, že se neustále rozšiřuje o přibývající řešící balíčky a v současné době jich podporuje víc než ostatní dostupné knihovny.

Díky této skutečnosti je právě Pulp ideální volbou, protože vytvořené pojetí problému může zůstat v případě naskytnutí lepšího balíčku neměnné a postačí pouhá změna na jiný řešící balíček, kterých je knihovnou Pulp poskytováno nejvíce.

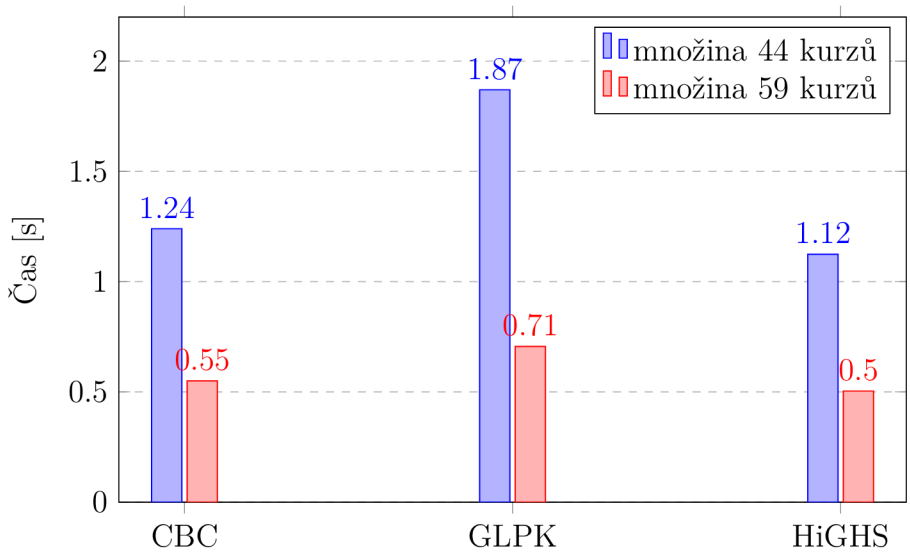
Řešící balíčky, které jsou považovány za potenciální kandidáty pro použití v CSP aplikaci byly podrobeny testování na dvojici datových množin. Výsledky jsou viditelné z obrázku 4.2. Pulp dříve používal GLPK jako svůj výchozí balíček, ale ten byl kvůli lepším výsledkům nahrazen CBC, jak je také patrné z testů provedených na tomto problému viz 4.2.

Jak je patrné z obrázku 4.2, test byl prováděn na množinách o 44 a 59 trénincích a balíčky na těchto množinách byly schopné vyprodukovat výsledky v časovém rozmezí od půl do dvou sekund. Nalezení řešení bylo u všech balíčků rychlejší v případě většího počtu dostupných kurzů a to z důvodu častějšího nalezení řešení a vyhnutí se většímu počtu iterací cyklu 4.7. Požadovaný počet kombinací byl nastaven na tři a při nižším počtu kurzů bylo provedeno okolo devíti iterací. Na rozdíl od větší množiny kde proběhly pouze tři iterace.

Z testu nejlépe vzešel balíček HiGHS, který je stále v raném vývoji. K tomu nabízí možnosti řešení komplexních MIP problémů nebo kvadratických programovacích modelů. Balíček rovněž podporuje vedle sériového řešení problémů, také paralelní běh. Výsledky tohoto balíčku jsou však poměrně zkreslené a to z důvodu neúspěšnosti některých pokusů, při kterých došlo k zamrznutí programu v důsledku selhání balíčku. Tento výsledek však neznamená nefunkčnost balíčku jako takového, ale spíše potvrzuje, že v aktuálním stavu není možné jej aplikovat na tento konkrétní problém. Selhání balíčku během testu bylo vyměřeno na 36.84 % v případě menší množiny a 16.6 % u větší množiny. Pokud bychom tento fakt ignorovali, tak HiGHS dosáhl nejlepších výsledků, avšak pro aplikaci je nezbytné, aby bylo vždy nalezeno řešení. Jak HiGHS tak CBC daleko překonali GLPK, což je společně se spolehlivostí důvodem proč byl zvolen CBC jako balíček pro nasazení v aplikaci.

V průběhu testů se průměrné využití pamětu pohybovalo okolo 73,5 MB a využití GPU (Graphics Processing Unit) bylo mezi 3,5 % – 6,7 % na Intel Core i5-6400 procesoru.

Byť je CBC výchozím balíčkem v knihovně Pulp, tak důvodů pro jeho implementaci je spousta. Díky otevřené licenci je po porovnání s dalšími dostupnými balíčky do knihovny Pulp vhodným kandidátem, jak je možné vydedukovat z obrázků 2.3 a 2.2. Balíček dobře zvládá práci s binárními proměnnými, které jsou předmětem optimalizace pro tuto práci.



Obr. 4.2: Porovnání výkonu různých ILP řešících balíčků

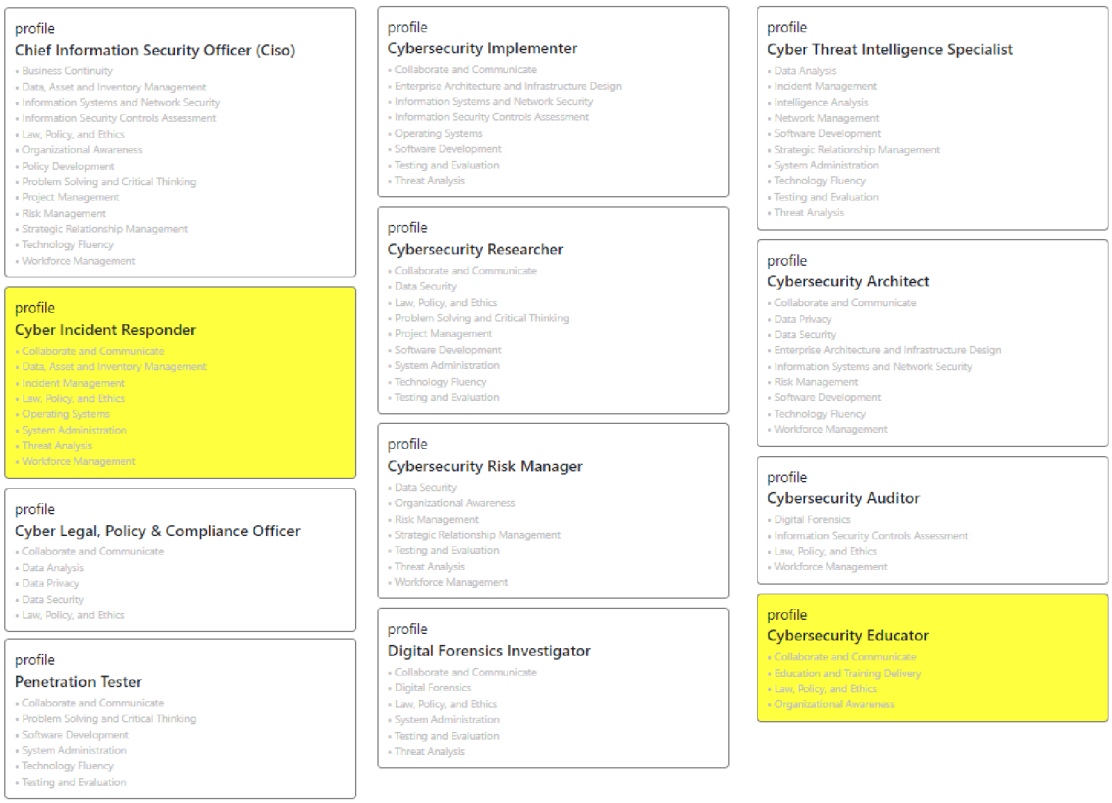
4.2 Architektura řešení

Proces vytvoření programu je popsán od webového rozhraní, na kterém uživatel nejprve zvolí parametry a odešle svůj požadavek. Dále je popsána část spojující webové rozhraní s optimalizačním skriptem a serverovou databází. Samotný algoritmus je obsažen v poslední části zaměřující se na samotnou optimalizaci za pomocí zmíněných technik a knihoven.

4.2.1 Webová stránka

Aby bylo možné modul nasadit a umožnit tak uživatelům práci s ním, bylo nutné vytvořit uživatelské rozhraní, které se v tomto případě přidávalo k již existující aplikaci CSProfiler, která využívá vývojového prostředí Javascript a knihovny React. Vyhledávací modul se tak řadí mezi doplňující funkce pro uživatele, kterým může napomáhat při plánování studia, vytváření osnov a vybírání kurzů k absolvování. Na obrázku 4.3 je uživateli prezentováno dvanáct profilů ze kterých lze zvolit žádoucí profil/y. K profilům je přidán stručný výčet dovednostních skupin, které jsou s profilem spjaté.

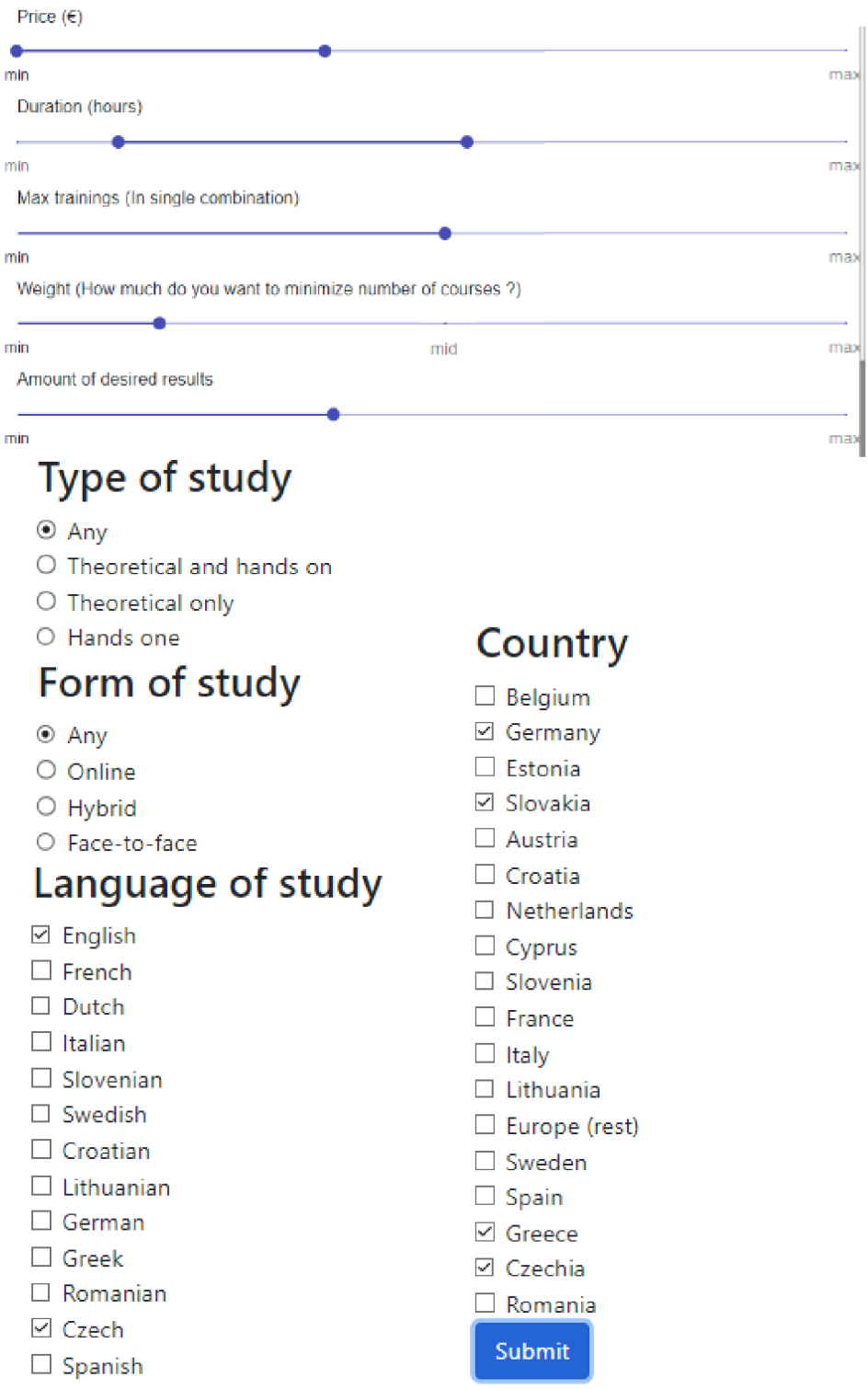
V další řadě je uživateli představena možnost modifikace parametrů, které jsou souhrně vyobrazeny na obrázku 4.4. Uživatel volí rozmezí celkové ceny pro absolvování výsledných kurzů, rovněž i časové rozmezí, které ať už online přednáškám nebo cvičením hodlá vynaložit. Maximální počet tréningů reprezentuje počet tréningů na výslednou kombinaci, aby tak uživatel spolu s nastavením váhy (weight) mohl



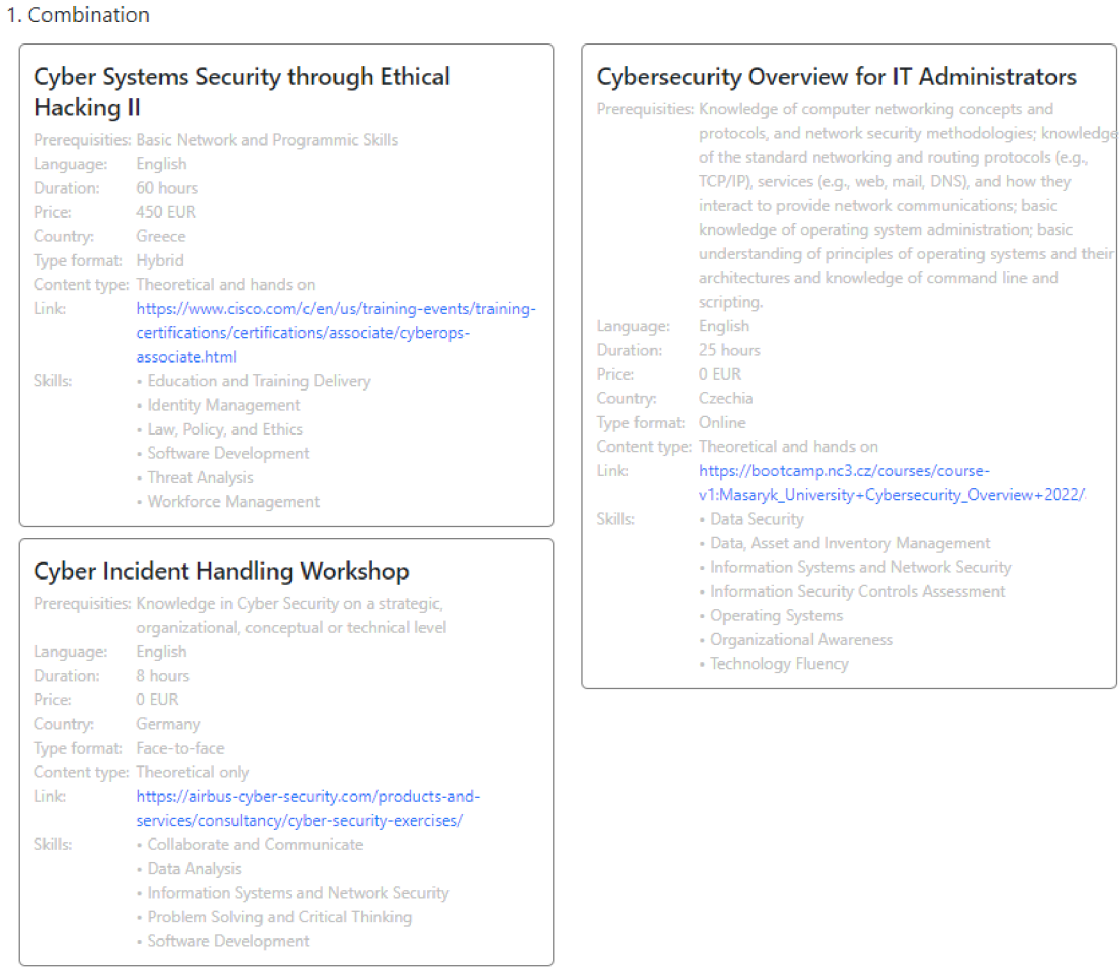
Obr. 4.3: Profily ve webové aplikaci

klást důraz na počet kurzů, které by chtěl podstoupit. V neposlední řadě zvolí požadovaný počet výsledných kombinací, které mu mají být prezentovány. Maximálně tento počet lze nastavit na hodnotu deset, avšak je obtížné při nízkém počtu kurzů a například při vysokých požadavcích uživatele mnohdy naplnit i nižší požadovaný počet. V případě, že není nalezena žádná kombinace vyhovující parametrům, je tak uživatel obeznámen s faktem, že řešení neexistuje. Jako další filtrovací možnosti se řadí i jazyk ve kterém bude kurz vyučován a také země ve které se kurz v případě, že není pouze online bude vykonávat. Tyto možnosti zahrnují pouze země v Evropě, jelikož se jedná o aplikaci, která je vyvíjena v rámci projektu Evropské unie [2].

Z pohledu uživatelského rozhraní je posledním krokem uživatele odeslání požadavku a následné obdržení odpovědi v podobě vykreslení výsledků jak je vidět na obrázku 4.5, který znázorňuje první vytvořenou kombinaci, která je zároveň z pohledu algoritmu tou nejoptimálnější. Řešení se v tomto případě skládá ze tří kurzů, které tak naplňují žádaný profil. Druhá kombinace na obrázku 4.6 opět sestává ze tří kurzů, které lze považovat za sub optimální řešení. Ačkoliv bylo v parametrech specifikováno, že uživatel požaduje kombinace tři, byly uživateli prezentovány pouze dvě. Algoritmus tak nebyl za daným parametrů schopný nalézt více než dvě ře-



Obr. 4.4: Parametry optimalizace



Obr. 4.5: První optimální kombinace

šení. Zároveň je při porovnání obrázků 4.5 a 4.6 patrné, že kurz „Cyber Systems Security through Ethical Hacking II“ se nachází v obou řešeních a algoritmus, tak nebyl schopný nalézt separátní kombinaci s kurzy, které nebyly obsaženy v prvním řešení. Prezentovaná data dočasně zobrazují pouze stručný popis kurzů, jelikož se jedná o první verzi webového modulu a jeho funkce je především demonstrační. V budoucnu bude výpis podrobností rozšířen například o data ve kterých budou tréningy probíhat, prerekvizity nutné k zápisu do kurzu, potencionálně získatelné certifikace, podrobný rozpis dovedností, znalostí, doplňující informace o organizátorech a další... Řazení výsledků probíhá na základě optimálnějšího výsledku účelové funkce a jsou tedy řazeny od nejlepších po méně optimální.

2. Combination

Cyber Systems Security through Ethical Hacking I

Prerequisities: Basic Network and Programmic Skills
Language: English
Duration: 12 hours
Price: 120 EUR
Country: Greece
Type format: Hybrid
Content type: Theoretical and hands on
Link: https://ekarinos.weebly.com/uploads/4/4/5/1/44512607/1_orig.png
Skills:
- Data Analysis
- Data, Asset and Inventory Management
- Organizational Awareness
- Software Development
- Threat Analysis

CyberRange: Advanced Persistent Threats and Targeted Attacks

Prerequisities: Knowledge about networks and operating systems as well as attack vectors
Language: English
Duration: 8 hours
Price: 0 EUR
Country: Germany
Type format: Face-to-face
Content type: Hands one
Link: https://airbus-cyber-security.com/products-and-services/consultancy/cyber-security-exercises/
Skills:
- Collaborate and Communicate
- Data Analysis
- Information Systems and Network Security
- Software Development
- Workforce Management

Cyber Systems Security through Ethical Hacking II

Prerequisities: Basic Network and Programmic Skills
Language: English
Duration: 60 hours
Price: 450 EUR
Country: Greece
Type format: Hybrid
Content type: Theoretical and hands on
Link: https://www.cisco.com/c/en/us/training-events/training-certifications/certifications/associate/cyberops-associate.html
Skills:
- Education and Training Delivery
- Identity Management
- Law, Policy, and Ethics
- Software Development
- Threat Analysis
- Workforce Management

Obr. 4.6: Druhá optimální kombinace

4.2.2 PHP modul

Poté co je z Reactu požadavek odeslán skriptu PHP, dojde k navázání spojení se serverovou databází. Následně se parametry i s profily předané z frontendu využijí pro počáteční filtrování požadovaných dat databáze. Filtrování probíhá přímo dotazem na databázi. Data, která jsou databází vrácena jsou poté zachycena v PHP a uložena do souboru ve formátu JSON. Python skript je prostřednictvím PHP spuštěn a spolu s tím je předávána i cesta k uloženému JSON souboru. Po skončení optimalizačního skriptu jsou výsledky zachyceny v PHP a odeslány do frontendu k zobrazení uživateli. Dočasné soubory obsahující již využitá data jsou mazána při další inicializaci PHP skriptu a nahrazena nově filtrovanými daty.

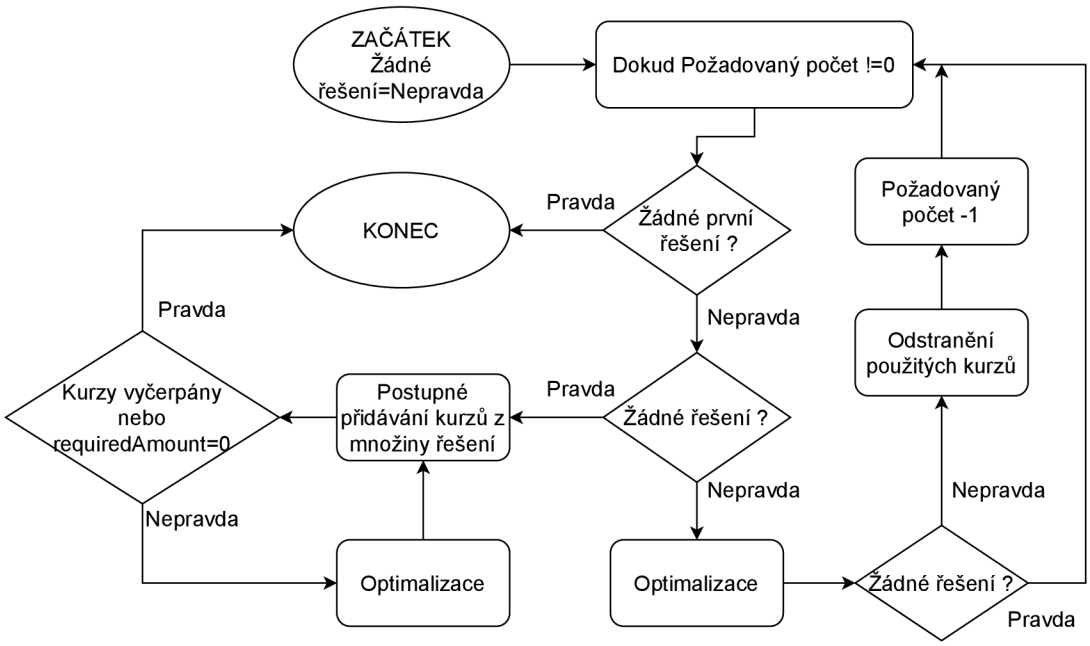
4.2.3 Python optimalizační modul

V této době je možné aby se zvolený profil skládal až z 31 kurzů. Všechny kurzy jsou uložené v serverové databázi. Modul soubor s kurzy zpracuje do společného slovníku a následně dle druhé filtrace dojde k vyřazení kurzů, které nemají žádnou shodu s dovednostními skupinami vybraného profilu. Dále jsou kurzy předmětem účelové funkce, která udává co je cílem minimalizace. Nejprve dojde k definování problému. Je rovněž nezbytné určit proměnné, které bude algoritmus mít možnost modifikovat. Jako proměnné zde slouží samotné kurzy nebo spíš jejich identifikační čísla. Hodnota těchto proměnných může nabývat dvou stavů, nula nebo jedna. Algoritmus tak hodnoty buďto zařadí mezi řešení, což je reprezentováno hodnotou jedna a nebo dojde k jejímu vyřazení nastavením na hodnotu nula. Rozhodování zda bude kurz vybrán závisí na více faktorech, hlavním je jeho celková cena, která se skládá ze samotné ceny kurzu a jeho délky. K tomu se také přidává váha nastavitelná uživatelem, která je přímo spjatá s počtem kurzů zahrnutých v řešení. Finální podmínka je spíše aplikována na celek ta spočívá v tom, že množina vybraných kurzů musí obsahovat všechny dovednostní skupiny, které obsahuje profil.

Vyprodukované řešení algoritmu je vždy pouze jedno a to právě globální optimum. Cílem však je poskytnout uživateli více možných kombinací, ze kterých si lze volit tu nejvíce vyhovující na základě osobních preferencí. Modul tak byl doplněn o rozhodovací cyklus znázorněný na obrázku 4.7. Tento cyklus umožňuje získání libovolného počtu kombinací, pokud to množina kurzů umožňuje.

Cyklus začíná s uživatelsky nastaveným požadovaným počtem výsledků. Během iterování je tento počet s každým výsledkem snižován dokud nedosahuje nuly nebo opakovaně nelze nalézt řešení. Při první iteraci dochází k řešení, po kterém je buďto odečtena hodnota požadovaných výsledků a tréningy které jsou zahrnuty v řešení jsou odebrány z globální množiny všech kurzů a nebo je nastavena hodnota „Žádné řešení“ na pravdivou a cyklus je při další iteraci ukončen bez výsledku. Pokud dojde k

nalezení řešení, tak se cyklus opakuje dokud není nalezeno řešení. V případě, že nebyl nalezen požadovaný počet řešení dochází k přesměrování do levé části cyklu, ve které jsou kurzy obsažené v řešení individuálně přidávány zpět do globální množiny. Tímto způsobem je dosaženo zvýšení šance na nalezení řešení, které je sub optimální, avšak nabízí požadovanou rozmanitost. Pokud není nalezeno řešení ani tímto způsobem, dojde k ukončení smyčky a uživateli jsou prezentována nalezená řešení.



Obr. 4.7: Rozhodovací cyklus optimalizace

Cílem je tedy minimalizace ceny p_{r} v součtu s dobou trvání d_{u} k tomu jsou oba parametry násobeny binární proměnnou b, která udává zda je kurz zahrnut mezi řešení. V případě, že by byl výsledek nerozhodný je součástí minimalizace suma vybraných kurzů b, která je dodatečně doplněna o váhu w. Tato váha umožňuje samotným uživatelům přidávat důraz na hledaný počet kurzů, který je tímto možné více či méně směřovat. Tím pádem bude vybrána skupina s menším počtem kurzů. Účelová funkce je tedy vyjádřena v rovnici (4.1).

$$S_{min} = \sum_{i=0}^{M} (p_{r_i} + d_{u_i})\, b_i + w \sum_{i=0}^{M} b_i \tag{4.1}$$

kde S_{min} je minimalizace cílové funkce.

Omezující podmínky udávají hranice, po kterých se během výpočtu lze pohybovat. Model problému musí splňovat podmínky uvedené níže a také musí platit, že všechny hodnoty jsou větší nule. Jedinými proměnnými v tomto modelu jsou binární hodnoty b . Proměnné rozhodují o zařazení kurzů do množiny. Pokud kurz nesplňuje

požadavky nebo není optimální je hodnota b rovna nule a při násobení je tím kurz vyřazen. V opačném případě je rovna jedné a kurz je zahrnut v řešení. Může nastat, že nebude možné nalézt optimální řešení, díky tomu, že vstupní hodnoty nesplňují všechny omezující podmínky. V takovém případě je zřejmé, že s vyfiltrovanými daty nelze omezení dodržet.

Hodnoty $P_{\min}$ a $P_{\max}$ vymezují cenové rozmezí ve kterém se součet kurzů musí pohybovat. V rovnici (4.3) jsou určeny hranice trvání zvolených kurzů v součtu.

$$\sum_{i=0}^{M}(p_{r_i}b_i) \geq P_{min} \qquad \sum_{i=0}^{M}(p_{r_i}b_i) \leq P_{max} \tag{4.2}$$

$$\sum_{i=0}^{M}(d_{u_i}b_i) \geq D_{min} \qquad \sum_{i=0}^{M}(d_{u_i}b_i) \leq D_{max} \tag{4.3}$$

Omezení uvedené v rovnici (4.4) vlevo určuje maximální počet kurzů obsažených v řešení. Na pravé straně s udává vědomostní skupiny obsažené ve vybraných kurzech. Skupiny ve vybraném profilu $Prof$ musejí být obsaženy ve skupině kurzů a s přitom může obsahovat i redundantní kurzy.

$$\sum_{i=0}^{M} b_i \leq C_{max} \qquad \sum_{i=0}^{M}(s\, b_i) \geq Prof \tag{4.4}$$

Simplexová metoda jako taková poskytuje pouze to nejoptimálnější řešení. Zde je však cílem získat více optimálních řešení, která mohou poskytnout uživateli možnost výběru. Tento cíl nelze dosáhnout jiným způsobem než opravením omezujících podmínek po obdržení výsledku, tak aby vylučovaly možnost stejného výsledku. S tímto přístupem se však pojí i degenerace dalších řešení jejich přehnanou penalizací. Je tedy důležité jakým způsobem budou další výběry penalizovány. Implementovaný způsob využívá odstranění vybraných kurzů z dalšího kola optimalizace. Pokud po omezení nebude nalezen další výsledek budou vybrané kurzy dávkovány do dalších iterací odděleně, aby se zvýšila pravděpodobnost nalezení řešení.

Závěr

Práce se zabývala popisem Evropského rámce dovedností v oblasti kybernetické bezpečnosti a projektu REWIRE. Poté byly analyzovány datové struktury, optimalizační algoritmy a některé knihovny a balíčky pro lineární programování. Nakonec bylo analyzováno zadání praktické části a byly aplikovány vhodné prostředky k dosažení požadovaného řešení.

Cílem bylo vytvoření vyhledávacího modulu, který na základě vstupních parametrů nalezne nejvhodnější kombinace tréningových kurzů v oblasti kybernetické bezpečnosti. Samotný modul, který měl optimalizaci vykonat byl vytvořen v programovacím jazyce Python za pomocí knihovny Pulp, která umožnila formulaci problému a také poskytla kompatibilitu s řešícím balíčkem. Python skript pracuje s databázovými daty ve formátu JSON, která jsou poskytována skriptem v programovacím jazyce PHP. Optimalizační modul je nasazený ve webové aplikaci, která umožňuje jednoduchý a přehledný přístup pro uživatele. Modul bude v budoucnu zakomponován do produkčního prostředí webové aplikace po doladění designu uživatelského rozhraní. Po funkční stránce je webový modul plně operativní a plní svoji funkci v reálném čase. Webové rozhraní bylo vytvořeno za pomocí Javascriptové knihovny React. Webová aplikace umožňuje uživateli výběr ze dvanácti profilů v oblasti kybernetické bezpečnosti. Uživatel má možnost v rámci aplikace navolit podrobnosti týkající se kurzů, které je zapotřebí absolvovat pro dosažení zvoleného profilu. Poté je uživatelem odeslán požadavek, v tu chvíli jsou data na základě parametrů filtrována ze serverové databáze a předána optimalizačnímu skriptu. Výsledná optimalizace je uživateli prezentována ve webové aplikaci. Uživateli se zobrazí zvolený počet výsledných kombinací pokud na základě parametrů bylo možné nalézt řešení.

Pro dosažení aktuálního stavu bylo nutné nejprve problém a dostupná data analyzovat. Následně ustanovit jaké parametry budou uživatelé moci upravovat a vybrat, jakým způsobem bude optimalizace prováděna. Bylo zvoleno lineární programování na základě předchozí analýzy a jako nejlepší zástupce se ukázal být balíček CBC. Balíček nepřevyšuje nijak svou komerční konkurenci, ale má nadprůměrné výsledky mezi ostatními volně dostupnými balíčky. Funkce programu splňuje svůj účel a efektivita modulu může být v budoucnu vylepšována nahrazováním řešících balíčků efektivnějšími variantami. Zdrojový kód modulu byl napsán pro možný budoucí vývoj a je doplněný o komentáře.

Jako největší překážkou se ukázalo být porozumění funkce lineárního programování a to konkrétně vytvoření účelové funkce a omezovacích podmínek. To jak efektivně bude algoritmus fungovat závisí právě na tom jakým způsobem bude problém definován a jaký balíček je na řešení aplikován.

V rámci výběru řešícího balíčku bylo provedeno experimentální testování na třech nekomerčních balíčcích, které rozhodlo o výběru balíčku k implementaci. Pro podrobnější prozkoumání, zda byla implementace CBC a Simplexového algoritmu správná, by bylo vhodné vyzkoušet implementaci dalších možných variant, ať už nasazení genetických algoritmů nebo podobných matematických algoritmů jako je Simplex.

Literatura

[1] European cybersecurity skills framework [online]. [cit. 2022-12-09]. URL: <https://www.enisa.europa.eu/topics/education/european-cybersecurity-skills-framework>.

[2] Petr Dzurenda a Sara Ricci. Mapping the framework to existing courses and schemes [online]. [cit. 2022-12-01]. URL: <https://rewireproject.eu/wp-content/uploads/2022/11/REWIRE_R3.4.1_Deliverable-v7-Final.pdf>.

[3] Chatzopoulou Argyro a Karras Apostolos a Petr Dzurenda. Cybersecurity career pathway analysis [online]. [cit. 2022-12-03]. URL: <https://rewireproject.eu/wp-content/uploads/2022/10/R3.5.1.-Cybersecurity-career-pathway-analysis_Final_v1.0.pdf>.

[4] The mapping of enisa profiles and rewire groups [online]. mapping the framework to existing courses and schemes,. [cit. 2023-05-10]. URL: <https://rewireproject.eu/wp-content/uploads/2022/11/REWIRE_R3.4.1_Deliverable-v7-Final.pdf>.

[5] S. Skiena Steve. In *The Algorithm Design Manual*, volume 2, pages 266–556. London: Springer London, 2008.

[6] Radim Burget. Optimalizace a genetické algoritmy [online]. [cit. 2022-11-20]. URL: <https://vutbr-my.sharepoint.com/:p:/g/personal/burgetrm_vutbr_cz/EUzxnPOycaVOpdn5p_t2pvABpc_dEZdCRES4RzSeVCAKlQ?rtime=-a6Dx7LJ2kg>.

[7] Kazuo Murota. Linear programming. In *Computer Vision*, pages 760–766. Springer, 2021.

[8] Florian A. Potra a Stephen J. Wright. Interior-point methods. *Journal of Computational and Applied Mathematics*, 124(1):281–302, 2000. Numerical Analysis 2000. Vol. IV: Optimization and Nonlinear Equations. URL: <https://www.sciencedirect.com/science/article/pii/S0377042700004337>, doi: https://doi.org/10.1016/S0377-0427(00)00433-7.

[9] Yves Pochet a Laurence A. Wolsey. *Mixed Integer Programming Algorithms*, pages 3–37. Springer New York, New York, NY, 2006. URL: <https://link.springer.com/book/10.1007/0-387-33477-7>.

[10] Bernhard Meindl a Matthias Templ. Analysis of commercial and free and open source solvers for linear optimization problems 1 [online]. 08 2013. [cit. 2022-10-2]. URL: <https://www.researchgate.net/publication/265117825_Analysis_of_commercial_and_free_and_open_source_solvers_for_linear_optimization_problems>.

[11] S.A. Mitchell a F. Peschiera J.S. Roy. Dokumentace modulu pulp [online]. [cit. 2022-11-13]. URL: <https://coin-or.github.io/pulp>.

[12] Eric Jones a Pearu Peterson a Travis Oliphant. Dokumentace modulu scipy [online]. [cit. 2022-11-25]. URL: <https://docs.scipy.org/doc/scipy/dev/index.html#scipy-development>.

[13] William Hart a Jean-Paul Watson. Dokumentace modulu pyomo [online]. [cit. 2022-11-25]. URL: <https://pyomo.readthedocs.io/en/stable>.

[14] Maximilian Parzen, Julian Hall, Jesse Jenkins, and Tom Brown. Optimization solvers: the missing link for a fully open- source energy system modelling ecosystem [online]. 05 2022. [cit. 2022-11-18]. URL: <https://zenodo.org/record/6534004>.

[15] John Forrest. Introduction to cbc [online]. [cit. 2022-11-16]. URL: <https://coin-or.github.io/Cbc/intro>.

[16] Julian Hall a Ivet Galabova a Michael Feldmeier. Dokumentace k balíčku highs [online]. [cit. 2022-11-24]. URL: <https://highs.dev>.

[17] Jakub Asszonyi. Webová aplikace pro monitoring stanic v počítačové síti [online]. [cit. 2023-05-10]. URL: <https://www.vut.cz/www_base/zav_prace_soubor_verejne.php?file_id=242487>.

[18] S Vernerová. Front end vs. back end - jaký je mezi nimi rozdíl? apitree [online]. [cit. 2023-05-01]. URL: <https://www.apitree.cz/blog/front-end-vs-back-end-jaky-je-mezi-nimi-rozdil>.

[19] Webová aplikace pro monitoring stanic v počítačové síti [online]. [cit. 2023-05-10]. URL: <https://developer.mozilla.org/en-US/docs/Learn/HTML>.

[20] React: Quick start [online]. meta platforms. [cit. 2023-04-28]. URL: <https://react.dev/learn>.

[21] Mehdi Achour, Friedhelm Betz, and Antony Dovgal , aj. Php manual [online]. the php group. [cit. 2023-05-08]. URL: <https://www.php.net/manual/en/>.

[22] Tim Peters. Timsort documentation [online]. [cit. 2022-11-22]. URL: <https://bugs.python.org/file4451/timsort.txt>.

Seznam symbolů a zkratek

ECSF	Evropským rámcem dovedností v oblasti kybernetické bezpečnost – European Cybersecurity Skills Framework
REWIRE	Cybersecurity Skills Alliance – A New Vision for Europe
PHP	Hypertext Preprocessor
ENISA	Evropská agentura pro bezpečnost sítí a informací – The European Union Agency for Cybersecurity
CSP	Cyber Security Profiler
LP	Lineární programování – Linear programming
GA	Genetické algoritmy – Genetic algorithms
MIP	Mixed-Integer Programming
CBC	Coin Branch and Cut solver
GLPK	GNU Linear Programming Kit
HiGHS	High-performance parallel linear optimization software
CSS	Kaskádové styly – Cascading Style Sheets
HTML	Hyper Text Markup Language
DOM	Objektový model dokumentu – Document Object Model
API	Aplikační programové rozhraní – Application Programming Interface)
SQL	Strukturovaný dotazovací jazyk – Structured Query Language
JSON	JavaScript Object Notation
GPU	Grafický procesor – Graphics Processing Unit

A Obsah elektronické přílohy

Elektronická příloha obsahuje zdrojové soubory webové aplikace včetně optimalizačního modulu a skriptů PHP. Program je funkční a zdrojový kód v příloze poskytuje přístup modulu vytvořenému v rámci této práce.

Pro zprovoznění webové stránky je nutné disponovat webovým a databázovým serverem, například pomocí XAMPP. Dále je nutné nainstalovat balíček pro správu prostředí npm ve verzi 14.14.5 a starší. Po instalaci otevřete příkazový řádek ve složce `csprofiler/client` a spusťte příkaz „npm install“, aby se nainstalovaly všechny potřebné moduly. Následně příkazem „npm start“ spustíte vývojářský server. Ve složce `csprofiler/db` se nachází SQL databáze, kterou je nutné importovat do databázového serveru. Soubory z `csprofiler/server` je nutné nahrát na webový server, ale předtím je nutné ve skriptu `csprofiler/server/optimization.php` změnit cesty k souborům na řádku 44 a 45. Mezi další prerekvizity patří Python verze 3.9.7. V rámci pythonu je zapotřebí nainstalovat knihovnu Pulp „pip install pulp“. Použití je určené pro Windows zařízení, při používání na zařízeních Linux je nutné ve skriptu `optimization-module.py` změnit na řádcích 21 a 25 „sys“ na „os“ a nahradit knihovnu i v importu.


/
csprofiler/
client/
public/
icon.png
index.html
logo192.png
manifest.json
robot.txt
src/
assets/
style.scss
components/
admin/
Admin.js
Certifications.js
RewireFramework.js
SkillSelectorModal.js
StudyPrograms.js
Trainings.js
UserManager.js
optimization/
enisaProfilesMap.js
Optimization.js
rewireSkills.js
userModals/
EditUserModal.js
LoginModal.js
RegisterModal.js
App.js
utils.js
VisitorStats.js
config.js
index.js
serviceWorker.js
package.json
package-lock.json
db/
csprofiler.sql
server/
config.php
db-local.php
getAllData.php
optimization.php
optimization-module.py