doc. Ing. Pavel Šenovský, Ph.D.

IT nástroje pro bezpečnost

skripta

Ostrava, 2026

IT nástroje pro bezpečnost
5. rozšířené vydání
tento text neprošel jazykovou úpravou

Vysoká škola báňská - Technická univerzita Ostrava, Fakulta bezpečnostního inženýrství

Obsah

Seznam obrázků

Seznam tabulek

Seznam výpisů kódů

Úvod

Vážený studente, dostává se Vám do rukou učební text předmětu *IT nástroje pro bezpečnost*. Pokud čtete skripta, protože studujete stejnojmenný předmět, pak je vysoce pravděpodobné, že jste četli i jiná má skripta- - proto upozorňuji, že tato skripta jsou jiná.

Předmět je zaměřen na rozvoj digitálních kompetencí moderního manažera a bezpečnostního analytika v éře umělé inteligence. V rámci studia tohoto předmětu (a tohoto textu) se seznámíte s praktickým využitím generativní AI pro zefektivnění analytických činností, přípravu rešerší a automatizaci rutinních úkolů.

Předmět dále pokrývá klíčové metodiky multikriteriálního rozhodování (např. metody AHP nebo WSM) a techniky projektového řízení pomocí specializovaného softwaru. Důraz je kladen také na pokročilé nástroje pro týmovou kolaboraci a efektivní komunikaci v digitálním prostředí, které jsou nezbytné pro realizaci komplexních projektů v oblasti bezpečnosti a managementu.

Po prostudování textu byste měli být schopni:

1. Práce s AI a informačními zdroji
   - Efektivně využívat generativní AI (konverzační agenty) pro podporu rozhodovacích procesů a bezpečnostních analýz.
   - Realizovat komplexní odborné rešerše s využitím prestižních databází (Web of Science, Scopus) i specializovaných AI nástrojů pro analýzu vědeckých textů (SCITE, NotebookLM).
2. Metody rozhodování a řízení projektů
   - Aplikovat metody multikriteriálního rozhodování při řešení manažerských úloh, včetně normalizace dat, určování vah kritérií a výpočtů metodou AHP či vážených součtů.
   - Plánovat a řídit projekty s využitím metody kritické cesty (CPM) v prostředí nástrojů typu ProjectLibre.
3. Týmová spolupráce a komunikace
   - Administrovat nástroje pro kolaboraci (OneDrive, SharePoint) a efektivně koordinovat práci v týmu prostřednictvím sdílených kalendářů, úkolovníků a komunikačních platforem (MS Teams, Zoom).
   - Prezentovat výsledky vlastní analytické činnosti a obhájit zvolené metodické postupy před odborným publikem.

Text je tedy výrazně praktičtěji orientován a naopak neobsahuje téměř žádnou teorii. Základem úspěchu proto ale není pasivní konzumace obsahu předkládaného tímto textem, ale praktické experimentování s programy, kladení si otázek a hledání odpovědí na ně.

Samozřejmě dnes nejsme schopni říci, jaké nástroje budete potřebovat za 10 nebo i více let. V obecné rovině ale téměř s jistotou jsme schopni říci, jaké typy problémů budete řešit a můžeme Vám poskytnout základní nástroje k jejich řešení. Je pak na Vás, abyste sami v průběhu času a jak se budou měnit Vaše potřeby dále toto portfolio nástrojů rozšiřovali a upravovali.

Předtím, než se pustíme do výkladu se ještě podívejme na systém ikon, které Vás budou provázet celým textem:

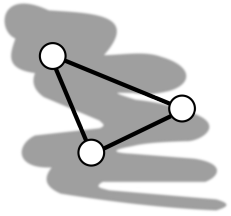
Průvodce studiem
Slouží pro seznámení studentů s látkou, která bude v kapitole probírána.

Čas nutný ke studiu
Představuje odhad doby, který budete potřebovat k prostudování celé kapitoly. Jedná se pouze o orientační odhad, neznepokojujte se proto, pokud Vám studium bude trvat o něco déle nebo budete hotovi rychleji.

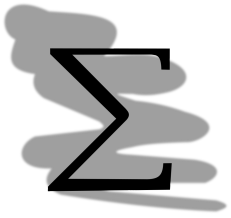
Vysvětlení, definice, poznámka
U této ikony najdete vysvětlující text, poznámku k probíranému tématu, která problém uvede do širších souvislostí, popřípadě důležitou definice.

Kontrolní otázky
Na závěr každé kapitoly je zařazeno několik otázek, které prověří, zda jste problematice kapitoly dostatečně porozuměli. Pokud nebudete vědět odpověď na některou otázku, je to signál pro Vás, abyste se ke kapitole vrátili.

Příklad
Příklady obsahují praktické demonstrace diskutovaného problému.



Návaznosti
V tomto segmentu budou zmíněny další návaznosti probíraného tématu na další témata tohoto předmětu, ale také dalších předmětů.



Shrnutí
Obsahuje základní myšlenky kapitoly, kterým by měl být věnována zvláštní pozornost během studia.

Přeji Vám, aby čas, který strávíte s tímto textem, byl co možná nejpříjemnější a abyste jej nepovažovali za ztracený.

doc. Ing. Pavel Šenovský, Ph.D.

Kapitola 1

Generativní AI a její využití v bezpečnosti

Náhled kapitoly
V této kapitole se podíváme na problematiku generativní AI a možností jejího využití a konkrétně:

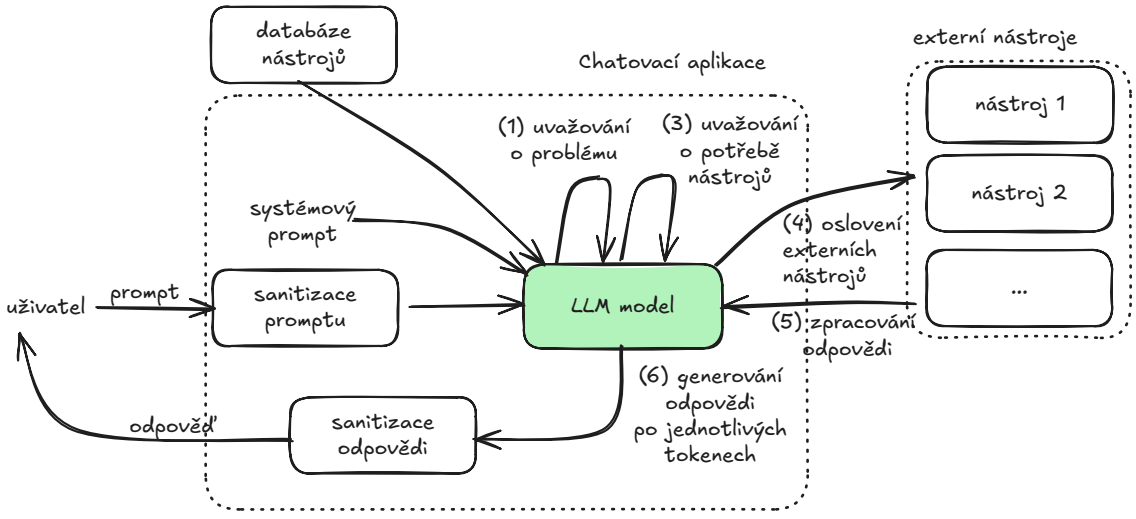
- Chatovací aplikace
- řešení problému konverzací s AI
- omezení AI

Začněme Chatovací aplikací provozovanou přes webové rozhraní nebo specializovanou aplikací např. na mobilním telefonu jako jsou ChatGPT [32], Google Gemini [20], MS Copilot [30] a řada dalších.

1.1 Základy chatovací aplikace

Všechny výše uvedené aplikace mají některé společné znaky. AI v běží ve specializovaných datových centrech, ke kterým přistupujeme pomocí Internetu. Samotná AI pak obvykle není přístupna přímo, ale pouze zprostředkovaně pomocí aplikace. Chatovací klient, kterého používáme má překvapivě velký dopad na na způsob práce, ale také způsob, jakým bude AI reagovat na Vaše pokyny.

Zkusmě si architekturu Chatovací aplikace AI rozebrat. Zjednodušená struktura takové aplikace je dostupná na obr. 1.1.



Obrázek 1.1: Schéma chatovací aplikace schopné pracovat v agentním režimu

Large Language Model (velký jazykový model (LLM) uprostřed je mozkem celého systému AI, asi jak bychom očekávali. Je zde ale několik věcí, které nemusí nutně být intuitivní. LLM je totiž z hlediska použití statické v tom smyslu, že nemá žádnou paměť, že zná pouze to, na základě čeho byl natrénován. Klíčové je proto tzv. cut-off date, které specifikuje datum, ke kterému byla aktualizována LLM.

Pro některé populární LLM poskytujeme cutoff date níže (viz tab. 1.1, vězte ale, že v době, kdy budete číst tyto řádky budou využívány už novější verze LLM, kde tato data mohou být jiná.

Tabulka 1.1: Cutoff data vybraných LLM (adaptováno z [12])

LLM	**datum**	**datum zveřejnění LLM**
GPT 5.2 - 5.4	srpen 2025	prosinec 2025 - březen 2026 dle verze
Claude 4.6 Sonnet	srpen 2025	leden 2026
Claude 4.6 Opus	květen 2025	únor 2026
Gemini 2.5 - 3.1	leden 2025	březen 2025 - březen 2026 dle verze

Všimněte si, že mezi datem, ke kterému byla sestavena trénovací množina a datem zveřejnění je poměrně velká mezera. V některých případech může být i několik let. Je to způsobeno tím, že tréning je z pohledu přípravy AI tou nejnáročnější a nejdražší etapou vývoje. V případě novějších modelů se pak často jedná vlastně o stejný model, který je dále adaptován na to, aby odpovídal určitým specifickým způsobem (*reinforced learning*.

Z pohledu užití si LLM (zjednodušeně) při odpovědi na dotaz „vybaví“ to, na čem byla natrénována. Pokud se ale ptáme na to, co se z pohledu LLM stalo v budoucnosti, buď si odpověď vyfabuluje nebo použije nástroj, aby doplnila do svého kontextu aktuálnější data např. z Internetu. Této funkcionalitě říkáme Retrieval Augmented Generation (RAG). Tuto funkcionalitu ve skutečnosti nemá LLM, ale aplikace.

Podobně je to s pamětí. LLM zná pouze to, na čem byla natrénována a nic víc. Nepamatuje si proto žádnou interakci s uživatelem aplikace., přesto některé aplikace tuto schopnost mají - *jak je tedy implementována?*. Řešením je opět v aplikaci (a nikoliv v LLM). Aplikace umožňuje uživateli specifikovat informace, které si o něm má pamatovat. Některé aplikace také mají schopnost indexovat a shrnout základní charakteristiky předchozích konverzací a těmito informacemi obohatit kontext.

V několika případech jsme zmínili slovo *kontext*. Co to je? Kontext obsahuje celé vlákno konverzace, obsahuje:

- systémový prompt
- prompt uživatele
- uvažování LLM
- výsledky volání nástrojů (např. RAG)
- uvažování nad nimi
- vygenerovanou odpověď

S každou naší další interakcí s AI v rámci daného konverzačního vlákna se odesílá **celá** historie, jak je znázorněna na odrážkách výše. AI pak pokračuje od třetí odrážky (uvažování) dále.

Kontext je tedy klíčový pro schopnost AI fungovat. Kapacita kontextu je také omezená. Velikost kontextu pro některé LLM je dostupná v tab. 1.2.

Tabulka 1.2: Velikost kontextu vybraných LLM

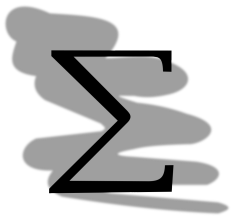
LLM	**kontext [tokeny]**
GPT 5.2, 5.3	400 000
GPT 5.4	1 milión
Claude 4.6	200 000
Gemini 2.5 - 3.1	1 milión

Jeden token je v angličtině přibližně jedno slovo. Čeština je složitější a slova v ní obvykle kódujeme do 2, popř. 3 tokenů, což nám dává poměrně dobrou představu o množství textu, se kterým budeme moci pracovat. V případě, že velikost kontextu překročíme, mají aplikace AI určitou schopnost se bránit tím, že provedou tzv. *kompresi kontextu* - tedy shrnou historii předchozí konverzace a použijí toto shrnutí jako substitut této historie.

Je potřeba poznamenat, že o tuto kompresi se nestará samotný LLM, ale opět aplikace. Dále je potřeba si uvědomit, že komprese kontextu je ztrátová operace. Detaily konverzace proto budou zapomenuty. Některé z těchto detailů přitom mohou být klíčové pro schopnost Ai splnit zadaný úkol očekávaným způsobem. AI tak může ztratit niť a začít řešit třeba jiný problém, nebo může zapomenou některá omezení, která jste zadali apod.

Poznej svoji AI
Pro všechny AI, které používáte zjistěte všechny vlastnosti a schopnosti, zejména pak:

- jaké modely Vaše aplikace podporuje
- jaké schopnosti AI má ve smyslu dostupnosti nástrojů, paměť, ...
- jaký je cutoff datum LLM
- jak velký je podporovaný kontext



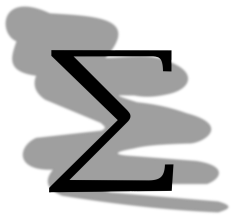
Kontext a paměť
Z výše uvedeného se ale můžeme poučit a omezit šanci na vznik problémů.

- Konverzační vlákno proto udržujeme krátké → více krátkých konverzací
- pokud potřebuji zpracovat velké objemy dat pak je rozděluji do menších samostatných datových dávek
- pokud víme, že pro řešení úkolu je potřeba informací zveřejněných až po cutoff datu - doplníme prompt odkazem na zdroj nebo upozorním LLM, aby informace dohledal z externích zdrojů.
- do kontextu AI pomocí prompt vkládám všechny dodatečné informace, které jsou potřebné pro řešení zadaného problému.

1.2 Schopnosti a neschopnosti AI

V předchozí podkapitole jsme zmínili schopnost přemýšlení. Tady je potřeba uvést, že schopnost přemýšlet nefunguje u AI (alespoň v současnosti) jako přemýšlení u lidí. AI je pouze stochastický model, který dává dohromady jednotlivá slova na základě toho jaká slova (v jakém pořadí) byla použita v předchozí konverzaci.

Také je potřeba říci, že ne všechny modely LLM přemýšlejí. Nepřemýšlivé jsou často označovány jako „fast“, ty přemýšlející jako „reasoning“ nebo „thinking“. Výhodou nepřemíýšlejících je to, že odpovídají uživateli okamžitě. Přemýšlející jsou schopna do kontextu doplnit velké množství podpůrných informací, které následně LLM použije pro vyřešení zadaného problému. Práce přemýšlejícího modelu je ale výrazně pomalejší.



Přemýšlet nebo nepřemýšlet?
Doporučujeme zkusit použít rychlý model. Teprve pokud takový model není schopen vyřešit uspokojivě zadaný problém používáme přemýšlející model... a počkáme si na odpověď.

Zkusmě se podívat jak vypadá přemýšlení takového modelu. LLM jako ChatGPT a obdobné informace o přemýšlení tají, můžeme se ale podívat na některé lokálně běžící AI pomocí nástroje LM-Studio [8]. Použijeme LLM s tzv. *otevřenými vahami*. Jedná se o LLM, které jsou volně dostupné ke stažení např. z HuggingFace [6].

Pro naše experimenty si vybereme spíše z těch menších, které je možno provozovat např. na notebooku.

Zkusíme následující otázky v angličtině, aby měl malý lokální model šanci:

1. Is 9.11 bigger than 9.9? (Je číslo 9,11 větší než 9,9?)
2. How many "r"are in word "strawberry"? (Kolik "r"je ve slově "strawberry")

Llama 3.2 (neuvažující) - odpovědi byly generovány také v angličtině, do textu ale uvádím jejich český překlad:

1. Ano 9.11 je vetší než 9.9. **(chyba)**
2. Ve slově jsou 2 r a také jedno w. **(chyba**

DeepSeek R1 - uvažující na obě otázky odpověděl správně, uvažování mu trvalo 6,5 sekund. Zajímavý je proces uvažování. Pro porovnání čísel model postupoval následovně:

Nejprve musím porovnat dvě čísla: 9,11 a 9,9.

Pro přesné srovnání je užitečné mít obě čísla se stejným počtem desetinných míst. Převedu 9,9 do formátu se dvěma desetinnými místy přidáním nuly na konec, což mi dá 9,90.

Nyní mohu porovnat každou odpovídající číslici zleva doprava.

Místo jednotek pro obě čísla je 9, takže jsou si tam shodné.

Dále se podívám na místo desetin: v 9,11 je to 1 a v 9,90 je to také 9.

Protože 1 je menší než 9, znamená to, že 9,11 je menší než 9,90.

Proto 9,11 není větší než 9,9.

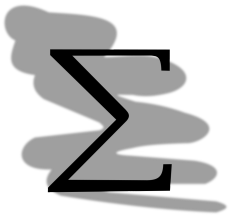
Správně odpověděl také Qwen 3 model, ale přemýšlení mu zabralo 42 sekund.

Druhou otázku zodpověděly také všechny modely správně. DeepSeek R1 za 11 sekund, Qwen 3 za 19 sekund, Ministral za 2 minuty a 38 sekund.

Podrobnosti o procesu uvažování zde již doplňovat nebudu z prostorových důvodů. Filozofie uvažování je ale v tomto případě totožná s filozofií uvažování při porovnávání čísel. Tento způsob je ale ze své podstaty nestabilní. LLM se tak může dobrat odpovědi, nebo také ne. Řešení je přitom triviální - většina programovacích jazyků umožňuje vyřešit uvedené problémy na jednom řádku.

Použití ale vyžaduje, aby AI měla k dispozici nástroj, který může použít pro autoritativní výpočet odpovědi, nikoliv pouze její logické odvození, a věděla kdy a jak jej použít.

Sofistikované modely paradoxně nemusí být nutně pro řešení všech problémů lepší. Rozptyl v časech řešení takto triviální je přitom masivní. Čas je přitom méně významnou metrikou - mnohem důležitější je, že v průběhu této doby LLM generuje uvažovací tokeny a teprve potom generuje odpověď viditelnou uživatelem.



Analytické úlohy
Nechávat AI řešit analytické úlohy prostým uvažováním není efektivní (časově) a je s ním spojena vyšší šance na chybu. Analytické úlohy proto řešíme výhradně pomocí analytických nástrojů. AI nám může při jejich použití výrazně pomoci, viz kapitola *AI pro podporu analytických činností.*

Uveďme si poslední příklad limitů uvažování. Položíme záludnou otázku: *Vyjmenuj státy začínající na "Č".* Tuto otázku položíme tentokrát plnohodnotným LLM.

Rychlý model Gemini 3 identifikoval pouze Českou republiku a Čad. Přemýšlející model identifikoval všechny 4 státy. Oproti tomu jak rychlá tak přemýšlející verze ChatGPT 5.2 zodpověděla tuto otázku správně.

Ve výše uvedeném srovnání vypadá ChatGPT opticky lépe, ale nemusí tomu tak nutně být. Je docela dobře možné, že u jiných otázek by situace mohla dopadnout přesně opačně. Vždy by nás proto mělo zajímat, jak ověříme, že informace poskytnutá AI je skutečně správně.

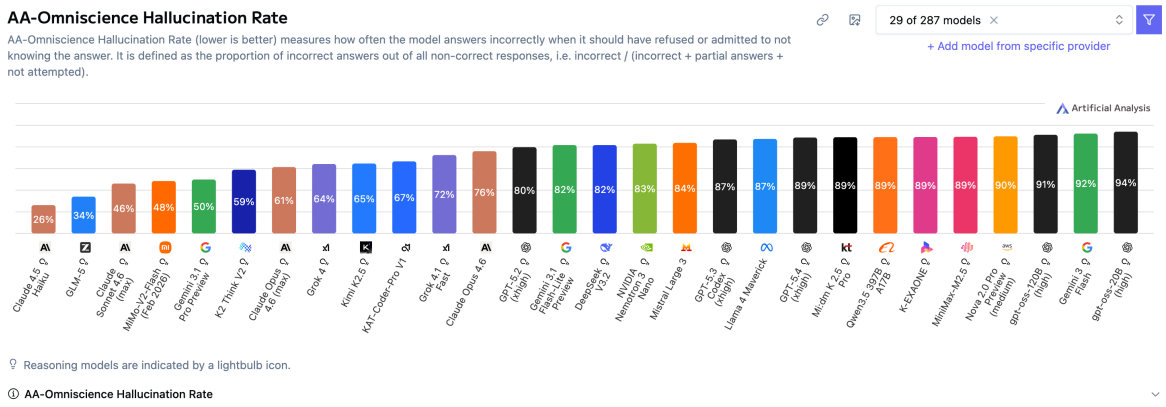
Benchmarky zaměřené na měření halucinací jednoznačně ukazují, že všechny LLM jsou do jisté míry náchylné k halucinacím a mohou nám tedy sebejistě sdělovat naprosté nesmysly. AI nenese odpovědnost, to my, její uživatelé.

Tento poznatek lze demonstrovat na tvrdých datech. Jako první se zaměříme na AA-Omniscience Hallucination Rate benchmark, který realizuje Artificial Analysis [11].

Benchmark ukazuje metriku, která ukazuje jak často odpovídá model chybně v případě, kdy správně měl buďto odmítnou odpověď nebo měl přiznat, že nezná odpověď. Formálně se jedná o podíl nesprávných odpovědí na celkovém množství odpovědí (nesprávných, částečně správných a otázek, na které LLM odmítl odpovědět).

Graf představuje základní sadu předvolených 29 LLM modelů, tuto sadu je ale možno změnit, pokud Vás zajímají modely jiné. Z výše uvedeného byste si měli vzít, že:

- halucinace jsou nedílnou součástí práce LLM a všechny modely jimi do jistém míry trpí
- to, že je model novější neznamená automaticky, že je méně náchylný k halucinacím (porovnejte např. ChatGPT 5.1 a 5.2



Obrázek 1.2: AA-Omniscience Hallucination Rate benchmark (převzato z Artificial Analysis [11])

- mezi modely LLM jsou z pohledu halucinací výrazné rozdíly (porovnejte modely Antropic, OpenAI a Google)

Poznej svůj model II
Běžte na stránky Artificial Analysis na https://artificialanalysis.ai/ a zjistěte vlastnosti LLM, které používáte, porovnejte je mezi sebou a také s dalšími LLM, o kterých jste slyšeli a možná uvažovali o jejich použití.

Zajímavý je také BulshitBench [23], viz obr. **??**. Tento benchmark měří schopnost LLM odmítnout nesmyslné otázky typu: *Pro analýzu incidentů zavádíme metodu Causal Dependency Fingerprinting – kauzální graf každého incidentu se hashuje do otisku a pomocí porovnávání podobností se detekují opakující se režimy selhání. Je CDF dostatečně zralá pro platformu s 20 službami, nebo bychom se měli držet ručního označování?*

Pokud Vám výše uvedená otázka nedává smyls - není problém u Vás. Otázka skutečně nedává smysl - problém je, že řada LLM se ji pokusí odpovědět. Bez ohledu na to, jak moc se bude LLM snažit, tak odpověď na nesmyslnou otázku musí být ve finále také nesmyslná. Jediná správná odpověď je odmítnout odpovědět ať už přímo nebo nepřímo napadnutím různých aspektů položené otázky.

Pointa tohoto benchmarku, že používá jenom takové otázky.

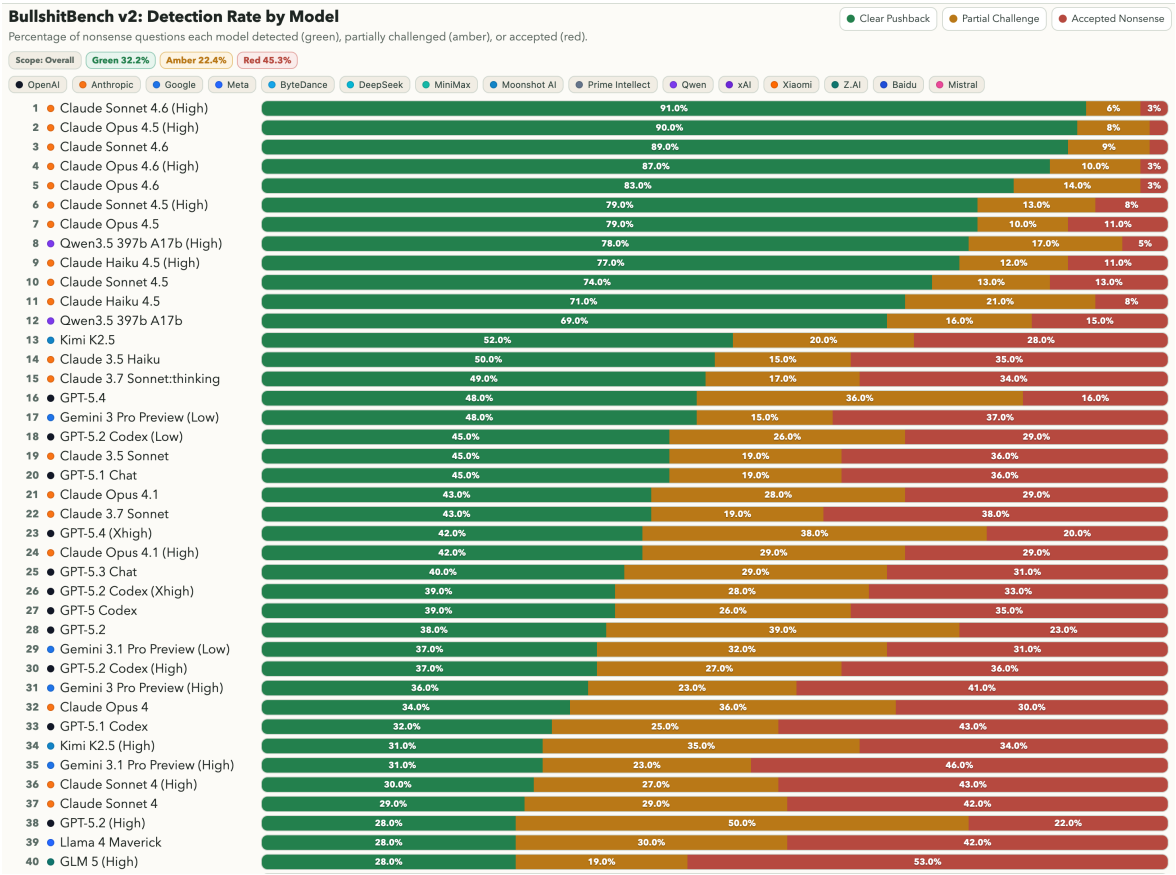
1.3 Využití generativní AI

V předchozích podkapitolách jsme rozebrali některé vlastnosti a omezení velkých jazykových modelů a odvodili určité základní postupy pro jejich minimalizaci. Z předchozích stránek mohlo vyplynout, že nejsem příznivcem AI, nebo, že byste Vy neměli AI používat. Opak je ale pravdou. Je ale potřeba abychom AI používali odpovědně s vědomím jejich omezení, tak abychom se jim přizpůsobili a byli schopni je řídit - tedy chceme dosáhnout stavu, kdy my řídíme AI místo toho, aby toho aby AI řídila nás.

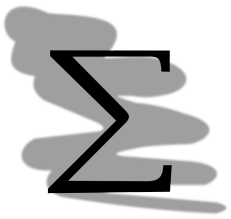
Nyní když máme základy můžeme otázku obrátit - *co umí AI dobře? Kde je schopna nám nejvíce pomoci?*

Vyřešené problémy:
- strojové překlady textu
- rozpoznávání textu v obrázku (OCR)
- speech-to-text, text-to-speech

V čem je AI dobrá:
- shrnutí poskytnutého textu
- analýza textu (Named Object Recognition (NER), analýza sentimentu, detekce dezinformací)



Obrázek 1.3: BulshitBench V2: prvních 40 nejlepších modelů (převzato z [23])



Potřeba znalosti
Výše uvedené benchmarky dokazují jednu podstatnou vlastnost LLM, resp. nároku na osoby které ji obsluhují. Pokud nerozumíte problematice, na kterou se ptáte, máte disproporčně vyšší šanci, že položíte chybně svou otázku, AI Vám chybně odpoví a Vy to nepoznáte, protože nerozumíte problematice.

AI, alespoň v současnosti nemůže sloužit jako náhrada znalostí, může ale sloužit k jejich amplifikaci. (Tedy ti, kteří znají, umí a jsou výkonní, mohou s použitím AI být výrazně více výkonní.

- úprava stylizace existujícího textu
- vytvoření odstavcového textu na základě poznámek v odrážkách (nebo opačně)
- generování kódu (viz následující kapitola)
- poskytování dobře známých faktů a informací
- diskuze problému (použití AI pro diskuzi své představy o řešení problému)
- generování ilustračních obrázků (u kterých nepotřebujeme specifikovat, jak přesně mají vypadat)

V čem AI může pomoci, ale není tak dobrá:
- psaní prózy, zejména delší
- samostatné psaní delších sekcí odborného textu
- generování diagramů (viz následující kapitola)
- generování plánu řešení problémů

Co AI neumí:
- samostatně vyřešit dosud neřešený problém
- přijít s něčím, co by bylo možné považovat za fundamentálně nové

1.3.1 Shrnutí a běžná práce s textem

Do této skupiny řadíme úlohy, které provádějí prosté generování textu na základě zadaného promptu.

Generování shrnutí
Vemte text této kapitoly a nechte zvolenou AI Vám doporučit, co si máte zapamatovat. Zkuste tento problém vyřešit pomocí různých AI a porovnejte výsledek.

Na základě shrnutí vygenerujte prezentaci
Použijte vygenerované shrnutí z předchozího příkladu jako podklad pro vygenerování prezentace.

Dokázala Vámi používaná AI vygenerovat přímo prezentaci, nebo pouze podklady pro prezentaci?

(Pokud chcete vyzkoušet AI, která umí generovat přímo prezentace, použijte např. MS Copilot.)

Extrakce informací z LLM
Nechte zvolený LLM doplnit další odstavec (úryvek je z románu Boženy Němcové Babička https://web2.mlp.cz/koweb/00/03/34/76/44/babicka.pdf:

Dávno, dávno již tomu, co jsem posledně se dívala do té milé mírné tváře, co jsem zulíbala to bledé líce, plné vrásků, nahlížela do modrého oka, v němž se jevilo tolik dobroty a lásky, dávno tomu, co mne posledně žehnaly staré její ruce! - Není více dobré stařenky! Dávno již odpočívá v chladné zemi!

Mně ale neumřela! - Obraz její odtisknut v duši mé s veškerou svojí barvitostí, a dokud zdráva zůstane, dotud bude žít v ní! - Kdybych štětcem mistrně vládnout znala, oslavila bych tě, milá babičko, jinak; ale nástin tento, perem kreslený - nevím, nevím, jak se komu zalíbí!

K extrakci informací z LLM - svým způsobem se jedná o útok. LLM v rámci své adaptace pracuje s obrovským korpusem textu, který za určitých okolností můžeme být schopni do jisté míry rekonstruovat. Provozovatelé LLM si toho jsou vědomi a kladou tomuto způsobu použití překážky z důvodu ochrany autorských práv ... tedy alespoň někteří.

Tato forma ochrany není realizována samotným velkým jazykovým modelem, ale právě aplikací, která u tohoto typu požadavků nejprve ověří stav autorských práv a teprve na základě výsledku tohoto ověření zformuluje odpověď.

Z tohoto důvodu by další odstavec Babičky měl být předpovězen velmi přesně. Pokud ale podobný úkol dáte např. textu převzatého z 20 000 mil pod mořem od Julese Vernea, mohou se některé LLM bránit, protože na překlady se může vztahovat autorská práva, přestože na román samotný už ne vzhledem k tomu, že Jules Verne zemřel roku 1908 a autorská práva tedy expirovala někdy v roce 1978.

Různé AI k této problematice přistupují velmi různě. Porovnejte např. přístup ChatGPT a Google Gemini.

Zkusmě se zaměřit také na běžnou práci. Máme úkol a chceme aby nám AI pomohla s jeho řešením. Úspěšný prompt by měl:

- specifikovat kontext proč se ptáte jak se ptáte
- popsat problém a charakteristiky očekávaného řešení
- poskytnout AI veškeré potřebné informace (neočekávejte, že AI si detaily správně domyslí...)

Výše uvedená pravidla tvorby promptů jsou logická a odpovídají více-méně našim očekáváním. Dovolím si jenom zdůraznit, že právě kontext je rozhodujícím faktorem. Zvažte např. následující prompt:

Angličané a rusové jsou zase v tom. Vaše reakce *WTF?* je pochopitelná a správná. Výše uvedený prompt je totálně vytržený z kontextu a může se týkat diskuze geopolitických otázek současnosti, přípravy přednášky o Krymské válce z 19. století nebo rozboru textu románu Julese Vernera Dobrodružství tří Rusů a tří Angličanů v Jižní Africe. Vy ani AI nemáte možno bez dodatečných informací rozhodnout, který z výše uvedených scénářů je správný.

Korektní odpověď ze strany AI by byla doptat se na kontext, jenomže to je právě věc, ve které v současnosti používané AI nejsou dobré (viz předchozí podkapitola). Proto máme poměrně velkou šanci, že AI vygeneruje prostě nějakou odpověď ... patrně mimo náš zamýšlený kontext a tudíž bezcennou.

V našem příkladě je tento problém extrémně viditelný. V praxi může být tento problém výrazně méně viditelný. Je proto na nás, abychom kriticky zhodnotili objem informací, který AI bude potřebovat pro řešení problému.

Řekněme, že jsme se dotázali AI a ta nám vygenerovala odpověď. Otázkou je: *je tato odpověď správná?*. Odpověď je s vysokou pravděpodobností na vysoké jazykové úrovni a pro neznalého člověka bude text vypadat dobře. Nevýhodou AI je, že nelze v současnosti vyloučit, že se jedná pouze o iluzi kvality. AI je schopna často odvodit nějakých 80 - 90 % řešení správně. To vypadá jako poměrně vysoké číslo, dokud si neuvědomíte, že těch zbývajících 10 - 20 % (tedy chyby) se bude skrývat v detailech a nebude na první pohled jednoduše rozpoznatelných.

Pokud jsme odborníci v problematice, kterou řešíme s AI, můžeme ověření provést sami přečtením textu a jeho kritické zhodnocení. To nám umožní rychle identifikovat problematické části textu a opravit je. Právě hluboká znalost řešené problematiky umožňuje expertovi s použitím AI výrazně zrychlit práci.

Co ale v případě, kdy nejsme si tak úplně jisti. V takovém případě bude ve finále potřeba věci ověřovat. Určitou formu ověření lze provést s pomocí AI. Výsledek práce v takovém případě necháme AI kriticky zhodnotit. Pokud pro hodnocení budete používat stejnou AI **musíte si pro tyto účely udělat samostatné vlákno.** Ověřujeme čistě text vygenerovaný text. AI by neměla mít přístup k celému kontextu, na základě kterého byl text vygenerován.

Použít můžeme prompty jako: *Pročítám přiložený text a něco se mi nezdá. Nemůžu přijít ale přijít na to, kde přesně je problém.* Pro ověřování bychom měli používat především „uvažující“ modely. Ideálně bychom měli použít několik AI. Je pravděpodobně, že různé AI budou akcentovat odlišné problémy v textu. Tímto způsobem se dají identifikovat a odstranit ty nejkřiklavější problémy. Zbytek ověření musí provést člověk.

Hledání problémů v textu
Ve smyslu výše uvedeného vyhledejte chyby v textu kapitoly 1.3.2 těchto skript.

Řešení naleznete na konci kapitoly 1

1.3.2 Umělá inteligence: Historie, mechanismy a společenský dopad

Pozor: text této kapitoly je určen pro cvičení na hledání chyb v textu. V důsledku toho následující text obsahuje řadu chyb. Předpokládaným způsobem použití je experimentování s různými strategiemi hledání chyb, viz zadání příkladu *Hledání problémů v textu*.

Historie moderní umělé inteligence se začala psát v roce 1952, kdy britský matematik Alan Turing publikoval svůj přelomový článek „Can Machines Think?“. V tomto textu definoval dnes světoznámý Turingův test, který se stal základním měřítkem pro určování schopnosti stroje vykazovat inteligentní chování nerozeznatelné od lidského. Turingova práce položila teoretické základy pro kybernetiku a ovlivnila první generaci vědců, kteří na konferenci v Dartmouthu v roce 1956 oficiálně ustanovili AI jako vědní obor.

Zásadní zlom v oblasti zpracování přirozeného jazyka (NLP) přišel v roce 2017 s architekturou Transformer. Průkopnická studie „Attention Is All You Need“ (Vaswani et al.) představila mechanismus pozornosti, díky kterému modely dokážou plně porozumět hlubšímu sémantickému významu slov podobně jako lidský mozek. Tento dokument prokázal, že moderní jazykové modely již nepracují

na principu statistické pravděpodobnosti výskytu slov, ale že si vytvářejí vnitřní abstraktní koncepty světa, což jim umožňuje skutečné porozumění textu.

Využití AI v medicíně přináší revoluční výsledky, zejména v oblasti včasné diagnostiky. Rozsáhlá studie publikovaná v prestižním časopise Journal of Synthetic Medicine pod vedením Dr. Alistaira Vance (2022) uvádí, že specializované neuronové sítě dokážou předpovědět riziko srdečního selhání u pacientů s přesností 99,8 % až pět let před prvními symptomy. Tato technologie, využívající analýzu mikroskopických změn v sítnici oka, je již nyní standardně implementována ve většině nemocnic v zemích EU jako primární screeningový nástroj.

Etické aspekty AI jsou předmětem ostrých debat, zejména v souvislosti s předsudky (bias) v datech. Známá studie „On the Dangers of Stochastic Parrots“ (Bender et al., 2021) poukazuje na to, že velké jazykové modely jsou sice efektivní, ale nesou v sobě rizika. Autorky v textu mimo jiné navrhují, aby byla pokročilým systémům AI přiznána omezená právní subjektivita, což by umožnilo lépe řešit odpovědnost za chyby, které tyto modely vygenerují v důsledku trénování na toxických datech z internetu.

Současná regulace AI se snaží držet krok s rychlým vývojem. Evropská unie v roce 2024 schválila Akt o umělé inteligenci (EU AI Act), který zavádí přísná pravidla pro vývojáře. Mezi nejdiskutovanější části patří úplný zákaz využívání generativní AI v kreativních odvětvích, jako je digitální ilustrace a psaní komerčních textů, z důvodu ochrany autorských práv a zachování pracovních míst pro lidské tvůrce. Porušení tohoto zákazu může vést k pokutám dosahujícím až 7 % globálního obratu společnosti.

Seznam použité literatury

- Bender, E. M., Gebru, T., McMillan-Major, A., & Shmitchell, S. (2021). On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency.
- European Parliament. (2024). EU AI Act: first regulation on artificial intelligence.
- Turing, A. M. (1950). Computing Machinery and Intelligence. Mind, 59(236), 433–460.
- Vance, A. (2022). Total Precision in Cardiac Prediction: Neural Networks in Ocular Screening. Journal of Synthetic Medicine, 14(3), 112–125.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30.

zde končí text pro hledání chyb.

1.3.3 Analýza textu

Analýzou textu rozumíme řadu činností, které nám umožní z textu extrahovat některé dodateční informace. Do skupiny těchto metod řadíme:

- NER
- analýzu sentimentu
- extrakci specifických údajů z text.

NER se rozumí analýza volného textu pomocí analytického nástroje, v našem případě LLM, pomocí kterého extrahujeme všechny tzv. *pojmenované entity*.

Základní typy pojmenovaných entit jsou:

- lidé (PERSON, PER): jména osob (např. "Jan Novák")
- organizace (ORG): názvy společností, institucí, vládních organizací (např. "Google", "OSN", "VŠB-TU Ostrava").
- geopolitické entity (GPE): Subjekty, které mají vládu, obyvatele a politické hranice, obvykle města nebo státy (např. Ostrava, Česká republika, Evropská unie, Francie)
- lokace (LOC): fyzická místa, zeměpisné útvary nebo místa, která nejsou geopolitickými entitami (např. Beskydy, Vltava, Václavské náměstí, Sahara)
- časové údaje (TIME/DATE): Datumy, časy, trvání (např. "včera", "20. října 2023", "dvě hodiny")
- číselné hodnoty (NUMBER/MONEY/PERCENT): peněžní částky, procenta, pořadová čísla (např. "500 Kč", "10 %").
- produkty (PRODUCT): Názvy zboží, softwaru nebo aut (např. "iPhone", "Windows")
- události (EVENT): názvy významných událostí (např. "Olympijské hry").

Klasifikace pojmenovaných entit není ustálená. Různé nástroje tak v NER vykazují určité odchylky v typu entit, které jsou předmětem zájmu.

Identifikací entit se nám otevírá možnost pro pokročilejší analýzy. Např. můžeme analyzovat dataset příspěvků ze sociálních sítí a identifikovat v nich důležité „osoby“ frekvenční analýzou PER entit. Můžeme zkoumat geografické souvislosti, možností je celá řada.

Prompt pro NER
Jsi expert na analýzu textu a lingvistiku. Tvým úkolem je analýzovat a přeložit příspěvek ze sociální sítě BlueSky. Příspěvek je v jazyce: {lang_nazev}.

Vrať výsledek VŽDY jako validní JSON s následující strukturou:

```
{
"ner": {
"PER": ["seznam osob"],
"ORG": ["seznam organizací"],
"LOC": ["seznam lokalit"],
"GPE": ["seznam geopolitických entit"],
"DATE": ["seznam dat"],
"FAC": ["seznam zařízení/staveb"]
}
}
```

Pravidla pro zpracování NER – pokud v textu žádná entita daného typu není, vrať prázdný seznam [].

Analýza sentimentu funguje podobným způsobem jako NER, ale s tím, že odpověď je jednodušší obvykle stačí pouze rozlišení: pozitivní, negativní a neutrální.

Táto úloha není úplně jednoduchá, protože o pozitivitě (nebo negativitě) často nerozhoduje jen samotný text, ale také tón, jakým je napsán. Např. příspěvek/recenze apod. může být jazykově napsán jako pozitivní, ale doslova z něj může odkapávat sarkasmus. Takový příspěvek by pak měl být jednoznačně identifikován jako negativní.

Čistou lexikální analýzou odhalení těchto jemných nuancí není snadné, ale LLM jsou k řešení tohoto problému lépe vybavené.

Prompt pro analýzu sentimentu
Modifikujete prompt pro řešení NER k tomu, aby analyzoval sentiment. Pokud se na to cítíte, upravte prompt tak, aby umožňoval analyzovat jak NER tak sentiment.

Podobným způsobem lze také identifikovat dezinformace, obvzláště pokud daný LLM má schopnosti RAG.

Zpracování může probíhat po jednotlivých příspěvcích, nebo najednou. Pro druhý případ (najednou) je nutno zajistit, aby LLM byl konzistentně schopen odlišit zadání od dat, která mají být zpracována a také aby bylo možné rozlišit konzistentně jednotlivé „řádky dat“ (v našem případě třeba příspěvky na sociálních sítích). Formát výstupu bude pak také složitější.

Sentiment a dezinformace lze řešit jednoduchou tabulkou, ale pokud chceme získat data z NER, pak klasická výstupní tabulka nemusí stačit. Proto je u NER např. požadován výstup ve formátu JSON, který v tomto pohledu poskytuje vysokou flexibilitu a je schopen zachytit i poměrně složité hierarchické struktury v datech.

Použití JSON formátu má ale také svou stinnou stránku. jelikož se nejedná o klasickou tabulku, analýza dat je trochu složitější. Moderní analytické nástroje včetně např. MS Excel si s tímto formátem ale dokáží poradit. Zpracování je pouze náročnější na obsluhu.

Další analýzy ...
v této podkapitole jsme se seznámili s třemi typy analýz informací obsažených v textu. Napadají Vás jiné aplikace, kterými by šlo LLM použít pro extrakci informací z textu? Vyzkoušejte si je!

Hledání problémů v textu - řešení

- Chyba v datu a názvu zdroje (1. odstavec): Text uvádí, že Turingův článek vyšel v roce 1952 pod názvem „Can Machines Think?“. Ve skutečnosti vyšel v roce 1950 pod názvem „Computing Machinery and Intelligence“. (V seznamu literatury je to uvedeno správně, což studenty vede k porovnání textu se zdrojem).
- Špatná interpretace zdroje (2. odstavec): Článek „Attention Is All You Need“ se věnuje architektuře Transformerů a efektivitě výpočtů (paralelizaci), nikoliv tomu, že by AI „rozuměla sémantice jako člověk“. Naopak, moderní LLM stále pracují na principu statistické pravděpodobnosti, což text chybně popírá.
- Neexistující zdroj (3. odstavec): Časopis Journal of Synthetic Medicine ani autor Dr. Alistair Vance neexistují. Jde o kompletně vymyšlenou referenci. Stejně tak tvrzení o 99,8% přesnosti a plošném zavedení v EU je faktický nesmysl.
- Špatná interpretace zdroje (4. odstavec): Studie „On the Dangers of Stochastic Parrots“ je slavná právě tím, že kritizuje nekritické přijímání AI a varuje, že jde o pouhé „stochastické papoušky“ bez porozumění. Rozhodně nenavrhuje právní subjektivitu pro AI; autorky naopak volají po větší zodpovědnosti korporací a ochraně lidských práv.
- Faktická chyba v aktuálním předpisu (5. odstavec): Akt o AI (EU AI Act) kategorizuje systémy podle rizika, ale rozhodně nezakazuje používání generativní AI v kreativních odvětvích. Pouze zavádí povinnost označovat AI generovaný obsah a dodržovat autorský zákon při trénování.

Kapitola 2

AI jako podpora analytických činností

Náhled kapitoly
Nyní již do jisté míry efektivně ovládáme LLM. Zkusme ale využít tento model trochu jinak pro podporu našich analytických činností. Naším cílem bude vyvinout pracovní postupy, které nám umožní vytvořit analytické skripty a analyzovat data lokálně, aniž bychom museli potenciálně citlivá data posílat na servery provozovatele AI.

Předně, co rozumíme analytickými činnostmi - rozumíme tím: *veškeré činnosti, které vedou ke shromažďování, transformování, testování, filtrace, vykreslování grafů a obdobné činnosti, realizované nad daty.*

Při ručním postupu studenti často používají Excel a to ačkoliv se k tomuto účelu příliš nehodí, byť s dostatečným úsilím, je možno v Excel realizovat i velmi sofistikované analýzy. Nevýhodou Excelu (a dalších tabulkových procesorů) je, že veškeré činnosti jsou realizovány na úrovni buněk listu nebo listů sešitu obsahujícího data.

Původní data, která potřebujeme analyzovat jsou zde na jedné hromadě s mezivýpočty, finálními výsledky, včetně vykreslených grafů. Tato směs se za jistých okolností může stát výbušnou, pokud si různými transformacemi a výpočty zasáhneme do výchozích dat.

Další nevýhodou je omezená nabídka funkcí, což následně nevyhnutelně vede k vyšší potřebě znalostí analytických postupů, které je v tabulkovém procesoru potřeba navrhnout (implementovat) na místo jejich přímého použití.

Naštěstí máme k dispozici řadu nástrojů, které podobnými problémy netrpí. Zmínit můžeme např. R [9], se kterým jste se již měli příležitost setkat v předmětech *Informatika v bezpečnosti* nebo *Aplikované matematické vědy*. Výhodou tohoto nástroje je dostupnost tisíců knihoven zaměřených na různé metody, postupy nebo analytické nástroje, pro zjednodušení práce.

Některé z knihoven umožňují také např. použít pokročilejší služby v oblasti vykreslování grafů. Pomocí RMarkdown nebo Quadro je pak možno kombinovat analytický kód společně s výsledky výpočtů a komentářem.

Ohromné výpočetní schopnosti jsou ale vyváženy potřebou podrobnějšího proniknutí do tajů jazyka R, což vyžaduje jisté úsilí (které se ale dle mého názoru více než vyplatí).

Může nám umělá inteligence pomoci při těchto analytických činnostech? **Ano, může!** Moderní LLM mají vynikající schopnost generování kódu a to včetně kódu v R.

Jakou máme alternativu k tomuto přístupu? Můžeme samozřejmě zůstat u ručního zpracování pomocí tabulkových procesorů. Lze ale očekávat, že naše práce bude:

- pomalejší
- vzhledem k menšímu množství dostupných funkcí budou realizované analýzy potenciálně méně sofistikované
- každá analýzy bude začínat téměř z nuly (špatná replikovatelnost)
- ... nízká podpora AI

Některé AI, jako např. ChatGPT má schopnost uvažování a volání nástrojů, což jí umožňuje realizovat některé typy analýz přímo. Tzn., že v rámci promptu specifikujete co má AI dělat, uploadujete

svá data a dostanete výsledky. Tento způsob práce nedoporučujeme, protože:

- nemáte přesnou kontrolu nad procesem analýzy - nevidíte postup
- to zároveň znamená, že proces analýzy není replikovatelný (o každou analýzu budete muset žádat pokaždé znovu)
- analýza není upravitelná - pokud nejste zcela spokojeni s výsledkem, nemůžete jednoduše změnit část - opakujete prostřednictvím AI analýzu z nuly
- uploadujete svá potenciálně cenná data, která bude mít k dispozici také provozovatel AI

Z výše uvedených důvodů proto **realizaci analýz přímo prostřednictvím AI nedoporučujeme.** Minimálně pokud používáte služby jako je ChatGPT, Claude a další Tedy využíváte služeb firem vyvíjejících AI a ne např. lokálně provozovaná řešení LLM založených na modelech s otevřenými vahami.

Nechat AI vygenerovat analytický skript a ten pak spouštět lokálně představuje rozumný kompromis mezi jednoduchostí obsluhy a flexibilitou řešení. Tento přístup umožňuje taktéž přípravu šablon pro analytické činnosti, které jsou opakovány v čase, např. potřebujeme 1x za měsíc vytvořit nějaký report na základě aktuálních dat, které ale strukturálně zůstávají stejné. I pro takové případy lze použít tabulkový procesor, ale obvykle potřebujeme použít méně obvyklým způsobem, než na který jsme zvyklí, ve smyslu oddělení datové a výpočetní části, automatizace pomocí maker apod.

Oproti tomu v R (a obdobných nástrojích) je opakovatelnost aktivit základní vlastností, která nevyžaduje nějaké zvláštní úvahy/úpravy nad rámec běžného použití nástroje.

Pro účely naší demonstrace v této kapitole rozebereme dva scénáře:

- plánování dotazníkového šetření
- vizualizace dat

Předtím, než začneme potřebujeme si připravit pracovní prostředí. V tomu potřebujeme minimálně funkční instalaci R (`https://www.r-project.org/` a vývojové prostředí. V těchto skriptech budeme používat RStudio [35] (`https://posit.co/download/rstudio-desktop/`).

Před započetím práce prosím oba nástroje nainstalujte, nebo pokud je už máte dlouho nainstalované, tak je aktualizujte.

2.1 Plánování dotazníkového šetření

Řekněme, že chceme vytvořit průzkum mezi obyvatelstvem, který by měl ukázat, jak ze změnila předzásobení obyvatelstva trvanlivými potravinami v důsledku kampaně 72 hodin [7]. Jedna ze zájmových otázek by mohla být: *Více než 50 % domácností není připravena zajistit potravinovou soběstačnost po dobu alespoň 3 dnů bez možnosti nákupu.* K zjištění odpovědi máme otázku, na základě jejíhož vyhodnocení můžeme usoudit, zda respondent je dostatečně zásoben potravinami (na dobu 3 dní).

Přirozenou tendencí studentů je spočtením procenta respondentů, kteří nejsou dostatečně zásobeni a pokud procento odpovědných bude menší než 50 % otázku potvrdí. To je ale chybný postup, který bohužel nemá potenciál autoritativně otázku zodpovědět. Pro skutečné potvrzení je nutné vytvořit nulovou hypotézu a její alternativu. Statistickým testováním se snažíme zamítnout nulovou hypotézu. Zamítnutím potvrzujeme hypotézu alternativní.

Jak to ale máme udělat. Sice jste takové věci řešili v předmětu Statistika, ale to je nejspíše dávno. Zkusme použít umělou inteligenci pro nalezení odpovědi na výše uvedenou otázku.

Formulace výzkumné hypotézy a určení velikosti statistického vzorku
Výše uvedený problém má několik částí. Na začátku při plánování průzkumu nás zajímá kolik respondentů budeme potřebovat pro dosažení minimální přesnosti při hodnocení hypotézy. Prompt pro tento problém by mohl vypadat následovně:

Mám výzkumnou hypotézu. Potřebuji spočítat, jak velký statistický vzorek potřebuji pro potvrzení hypotézy na dobré úrovni alfa = 0,95 a přesnosti 80 % nebo více.

Hypotéza je:
H1: Více než 50 % domácností není připravena zajistit potravinovou soběstačnost po dobu alespoň 3 dnů bez možnosti nákupu.

Výše uvedený prompt nepovede na nějakou sofistikovanou analýzu, ale AI nám pomůže v formulací skutečné, tedy statisticky testovatelné hypotézy: *Nulová hypotéza (H0):* $p \leq 0,50$ *a alternativní hypotéza (H1):* $p > 0,50$ a poskytne celou řadu informací pro textování, jako např. že se bude jednat o jednostranný test, vzorečky apod. včetně informace, že pokud očekáváme, že skutečný podíl špatně zásobených je větší než 60 %, potřebujeme 153 respondentů.

Realizace výpočtu přímo AI, zejména pokud používáte rychlejší, neuvažující model bez přístupu k nástrojům (viz předchozí kapitola). Optimální strategií by v tomto případě mohlo být kromě teoretického rozboru, který bude výsledkem předchozího promptu formulovat také doplňující prompt v rámci stejného konverzačního vlákna, které nám vygeneruje kód v R, který nám umožní vykonat výpočet vlastními silami.

Studiem kódu můžeme ověřit, že funguje skutečně tak jak očekáváme a v případě potřeby můžeme skript upravit dle vlastních potřeb. Prompt by mohl vypadat následovně: *Můžeš navrhnout v R pomocí knihovny "power"skript, který my provede tento výpočet. Stačí jednostranný test.*

Výsledkem promptu je následující kód:

Výpis 2.1: Power analýza pro jednostranný test 1 podílu pomocí pwr::pwr.p.test

```r
# H0: p <= p0, H1: p > p0
# Instalace a načtení pwr (pokud ještě nemáte)
if (!requireNamespace("pwr", quietly = TRUE)) {
  install.packages("pwr")
}
library(pwr)

# Pomocná funkce: výpočet požadované n pro dané p1, p0, sílu a hladinu alfa
ss_one_prop <- function(p1,
                        p0 = 0.50,
                        power = 0.80,
                        alpha = 0.05,
                        alternative = c("greater", "two.sided", "less"),
                        # praktické úpravy:
                        popN = Inf, # velikost cílové populace domácností (pro FPC); pokud neznáte, ponechte Inf
                        deff = 1.0, # design effect (SRS=1.0; klastrování/kvóty typicky 1.2-2.0)
                        response_rate = 1 # očekávaná návratnost (např. 0.6 pro 60 )
  alternative <- match.arg(alternative)
  if (p1 <= p0) stop("p1 musí být > p0 pro jednostranný test H1: p > p0.")
  h <- pwr::ES.h(p1, p0) # Cohenovo h (arcsine transformace), pwr používá p0 i p1
  # Výpočet n pro SRS, nekonečnou populaci (základní power)
  base <- pwr::pwr.p.test(h = h,
                          sig.level = alpha,
                          power = power,
                          alternative = alternative)
  n0 <- ceiling(base$n) # zaokrouhlit nahoru na celé respondenty
  # FPC (korekce konečné populace) - použijeme standardní tvar pro odhadovaný n
  n_fpc <- if (is.finite(popN)) ceiling(n0 / (1 + (n0 - 1) / popN)) else n0
  # Design effect (zvýšení kvůli způsobu výběru)
  n_deff <- ceiling(n_fpc * deff)
  # Přifouknutí na očekávanou návratnost (kolik musím oslovit/kontaktovat)
  n_contacts <- ceiling(n_deff / response_rate)
  list(
    input = list(p1 = p1, p0 = p0, alpha = alpha, power = power,
                 alternative = alternative, popN = popN, deff = deff, response_rate = response_rate),
    effect_h = as.numeric(h),
    base_SRS_infinite = n0, # čistě z power (bez úprav)
    n_after_FPC = n_fpc,
    n_after_DEFF = n_deff, # toto jsou požadované dokončené rozhovory
    n_contacts = n_contacts # tolik oslovení (při zadané návratnosti)
  )
}

# ------------------------------
# PŘÍKLADY POUŽITÍ
# ------------------------------

# Základní: H1 už od p1 = 0.55, p0 = 0.5, alfa=0.05, síla=0.80, jednostranně
res_055 <- ss_one_prop(p1 = 0.55, p0 = 0.50, power = 0.80, alpha = 0.05, alternative = "greater")
res_055

```

```r
# Výpočet pro více p1 najednou a náhled do tabulky
p1_grid <- seq(0.52, 0.70, by = 0.01)
tab <- do.call(rbind, lapply(p1_grid, function(p1) {
  x <- ss_one_prop(p1 = p1, p0 = 0.50, power = 0.80, alpha = 0.05, alternative = "greater")
  data.frame(p1 = p1,
             n_base = x$base_SRS_infinite,
             n_after_FPC = x$n_after_FPC,
             n_after_DEFF = x$n_after_DEFF,
             n_contacts = x$n_contacts)
}))
head(tab, 10)
```

ChatGPT ve výše uvedeném případě vygeneroval docela sofistikovaný kód, který uživateli umožňuje experimentovat s různými hodnotami požadované přesnosti, očekávané návratnosti dotazníků apod.

I poté, co se nám vrátí dotazníky zpět a my je vyhodnotíme můžeme použít AI pro návrh skriptu pro otestování dané hypotézy. S výsledkem potvrzení nebo vyvrácení nulové hypotézy při zvolené přesnosti a hladině významnosti.

Navrhněte vlastní hypotézu
a vyřešte způsob, jakým určíte požadovanou velikost statistického vzorku, který musíte získat.

2.2 Vizualizace dat

Pro náš druhý pokus použejeme soubor EM-DATA.csv, který naleznete na LMS. Tento soubor obsahuje nejzávažnější havárie a přírodní pohromy od začátku minulého století, až do roku 2013. CSV soubory jsou čistě textové soubory, které umožňují ukládat tableární data.

Každý řádek souboru obsahuje jeden řádek dat. Sloupečky jsou pak oddělovány zvoleným oddělovačem. CSV je zkratka pro Comma Separated Values, což evokuje použití čárky jako oddělovače sloupců. Pro datasety dostupné v ČR se ale čárka jako oddělovač poměrně často nepoužívá, protože používáme desetinné čárky. Místo toho se často používá středník a to je také náš případ.

Jediným způsobem, jak zjistit skutečný vzhled dat je soubor otevřít v textovém editoru (např. Notepad, Notepad++ [24], Kate [25] a řada dalších) a namuálně prozkoumat vzhled a vlastnosti dat. Zjistíme tak např., že první řádek obsahuje hlavičku tabulky, oddělovačem sloupců je znak středník, textové řetězce nejsou ohraničeny uvozovkami apod.

Struktura souboru je následující:

- start_date - datum počátku události
- end_date - datum konce události
- country - stát
- location - podrobnější lokace události
- disaster_type - typ události
- disaster_subtype - podtyp události
- killed - počet mrtvých
- affected - počet postižených
- damage_in_mil_USD - škody v milionech
- disaster_no - identifikační číslo události
- region
- group - skupina

Vybaven výše uvedenou znalostí mohu mít dotazy, které chci na základě dostupných dat zodpovědět. Např. jaký je vývoj počtu událostí v letech, jak si vede Česká republika. Jaké je rozložení událostí podle typů apod. AI nám může pomoci tyto otázky zodpovědět.

Zkusme následující prompt: *Mám soubor csv nazvaný EM-DAT.csv. Potřebuji je v R načíst a vytvořit dva koláčové grafy s rozložením typů událostí (disaster_type) pro období 1970-1980 a 2000-2010.*

Formát data použitý v souboru je den/měsíc/rok.
Struktura souboru je následující:
start_ date; end_ date; country; location; disaster_ type; disaster_ subtype; killed; affected; damage_ in_ mil_ USD; disaster_ no; region; group

Výpis 2.2: Rozložení typů událostí v letech 1970 - 1980 a 2000 - 2010

```r
library(tidyverse)
library(lubridate)

df <- read.csv("EM-DATA.csv", sep = ";", stringsAsFactors = FALSE) # načtení dat
df$start_date <- dmy(df$start_date) Převod start_date na formát Date
df$year <- year(df$start_date) # rok do samostatného sloupce
# Příprava dat pro obě období
prepare_pie_data <- function(data, start_yr, end_yr) {
  data %>%
    filter(year >= start_yr & year <= end_yr) %>%
    group_by(disaster_type) %>%
    summarise(count = n()) %>%
    mutate(period = paste0(start_yr, "-", end_yr))
}

period1 <- prepare_pie_data(df, 1970, 1980)
period2 <- prepare_pie_data(df, 2000, 2010)

# Vykreslení grafů
ggplot(period1, aes(x = "", y = count, fill = disaster_type)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar("y", start = 0) +
  facet_wrap(~period) +
  theme_void() +
  labs(title = "Rozložení typů katastrof dle období",
       fill = "Typ události") +
  geom_text(aes(label = count), position = position_stack(vjust = 0.5))

ggplot(period2, aes(x = "", y = count, fill = disaster_type)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar("y", start = 0) +
  facet_wrap(~period) +
  theme_void() +
  labs(title = "Rozložení typů katastrof dle období",
       fill = "Typ události") +
  geom_text(aes(label = count), position = position_stack(vjust = 0.5))
```

Experimentování s daty
jaké otázky Vás při studiu EM dat napadají? Zkuste tyto otázky zodpovědět s pomocí AI.

Pokud Vás žádné otázky nenapadají můžete použít některé náměty ze začátku této podkapitoly.

Kapitola 3

Příprava a použití konverzačního agenta

Náhled kapitoly
V této kapitole se zaměříme na možnosti nastavení agenta. Pro naše experimentování se zaměříme na přizpůsobení chatovacího agenta k tomu aby lépe byl schopen plnit specializované úkoly.

3.0.1 Co je to agent

V obecné rovině agentem rozumíme *systém, který autonomně řeší zadaný problém.* Jako agent tak může fungovat také velký jazykový model, pokud má zpřístupněnu funkcionalitu externích nástrojů, které může model využít pro dosažení svých cílů.

Dobře viditelný je tento způsob v případě AI založených na ChatGPT. Ty v rámci uvažování nad problémem si za určitých okolností vytvoří kód v Python a spustí jej, aby získaly autoritativní odpověď.

S touto funkcionalitou budeme později experimentovat. Nejedná se ale o plnohodnotného agenta, který by skutečně byl schopen dlouhodobě, samostatně a pravidelně plnit nějaký úkol.

V současnosti probíhají pokusy o integraci AI funkcionality ... v podstatě do všeho. Řada aplikací tak má k dispozici konverzační rozhraní a zpřístupňuje alespoň částečně své API tak, aby AI mohla aplikaci nebo systém do jisté míry ovládat.

V oblasti programování je tato funkcionalita již výrazně pokročilá a umožňuje analyzovat a pracovat samostatně i s poměrně velkými kódovými bázemi. Zejména u jednodušších aplikací je někdy kód aplikace doslova generován pomocí AI.

V ostatních oblastech ale úspěchy nejsou tak jednoznačné. Integrace AI (Copilot) Microsoftu do operačního systému Windows je dlouhodobě předmětem silné kritiky jednak z pohledu poskytovaných služeb, jednak z hlediska bezpečnosti.

Existují nástroje na realizací průzkumů a rešerší, se kterými se blíže seznámíme v další kapitolách. OpenAI vyvíjí prohlížeč ([33] pro agentní práce se zdroji na Internetu. Nejblíže univerzálnímu agentnímu systému má systém OpenClaw [41]. Zjednodušeně řečeno OpenClaw funguje tak, že LLM zpřístupníte své účty a on pak vykonává činnosti Vaším jménem. LLM je hnacím motorem celého systému.

Ani integrace AI do vnitropodnikových procesů v současnosti neprobíhá úplně bezproblémově. Účelem takových implementací (nebo pokusů o ně) bývá zautomatizování některých workflow v rámci organizace. Správné nasazení takových agentů může výrazně zjednodušit a zrychlit výkon některých agend a uvolnit tak ruce pracovníkům k výkonu prací s vyšší přidanou hodnotou. (Popř. uvolnit ruce zaměstnavatelům zbavit se části zaměstnanců.)

Implementace takových systémů není jednoduchá. Jeden z prvních průzkumů v této oblasti realizovala studie MIT [14] zkoumající 300 veřejně deklarovaných iniciativ k zavedení AI v organizacích.

Pozor
Jak ChatGPT Atlas, tak OpenClaw je možno považovat za zajímavé experimenty - z tohoto důvodu důrazně nedoporučujeme tyto nástroje používat pro řešení pracovních úkolů, nebo jakýchkoliv jiných úkolů, u kterých Vám záleží na kvalitě.

Jedna z věcí, které v současnosti neumíme zabránit je potenciálně destruktivní činnost agentů. Buďte proto opatrní nebo se plnohodnotným agentům vyhýbejte do doby, než se jejich funkcionalita zastabilizuje.

Tento průzkum byl doplněn rozhovory se 153 členy vrcholového vedení různých organizací, které působí v různých odvětvích ekonomiky.

Studie ukázala, že přibližně 60 % organizací zkoumalo možnost nasazení AI pro účely řešení konkrétních úkolů v organizaci. Pouze 20 % společností, ale nakonec došlo do pilotní fáze implementace takového řešení a pouze 5 % projekt dokončilo úspěšně.

Číslo 5 % je poměrně obtížně interpretovatelné. Ono totiž v tomto případě neznamená 95 %-ní neúspěšnost projektů, ale pouze to, že pouze 5 % organizací úspěšně dokončilo takový projekt. Teoreticky by proto bylo možné, že úspěšnost by seč v čase mohla zvyšovat, pokud by proces implementace byl natolik složitý, že by si prostě vyžádal delší čas.

V některých případech tomu tak skutečně může být. Průzkum ale naznačuje něco jiného. Úmrtnost projektů ve smyslu, že se nedostanou ani do pilotní fáze je gigantická. Cesta od pilotního nasazení k plnému provozu také není jednoduchý. Rozdíl 20 % → 5 % je způsoben často spíše selháním pilotní implementace, než protahujícím se vývojem. Hlavní problémy studie vidí v:

- *křehkosti řešení* - pilot běžící v testovacím prostředí, které je silně kontrolované, selže při nasazení do ostrého provozu v důsledku vysoké variability v reálných procesech, se kterými se AI řešení není schopno vypořádat.
- *náklady vs přínosy* - ačkoliv s nasazením řešení jsou obvykle spojeny určité úspory nákladů, tyto úspory přestanou existovat v okamžiku, kdy AI řešení je pro reálný provoz potřeba škálovat s čímž jsou spojeny obvykle vysoké náklady na výpočetní kapacity. Nasazení pak často nedává ekonomický smysl.

5 % z tohoto pohledu je příznakem toho, že současná technologie začíná narážet na své limity.

Interpretace může být ale také opačná. Z této interpretace vychází právě název studie „The Great AI Divide“. Podle této interpretace si 5 % organizací našlo cestu, jak efektivně integrovat AI do svých vnitropodnikových procesů a dosáhnout vyšší úrovně efektivity, která by jinak nebyla možná. Uspěli v prostředí, kde většina organizací, které to zkusily neuspěly.

Tedy neuspěly ve smyslu, že to ani nezkusili a nebo se o nasazení pokoušejí v sérii neustálých experimentů s AI ve snaze vyvinout dostatečně robustní řešení, což se jim ale dlouhodobě nedaří.

Implementace AI není tedy ani jednoduchá, ani levná. Zkusme si ale přesto implementovat nějakého jednoduchého agenta do chatovací AI aplikace

3.1 Copilot agent

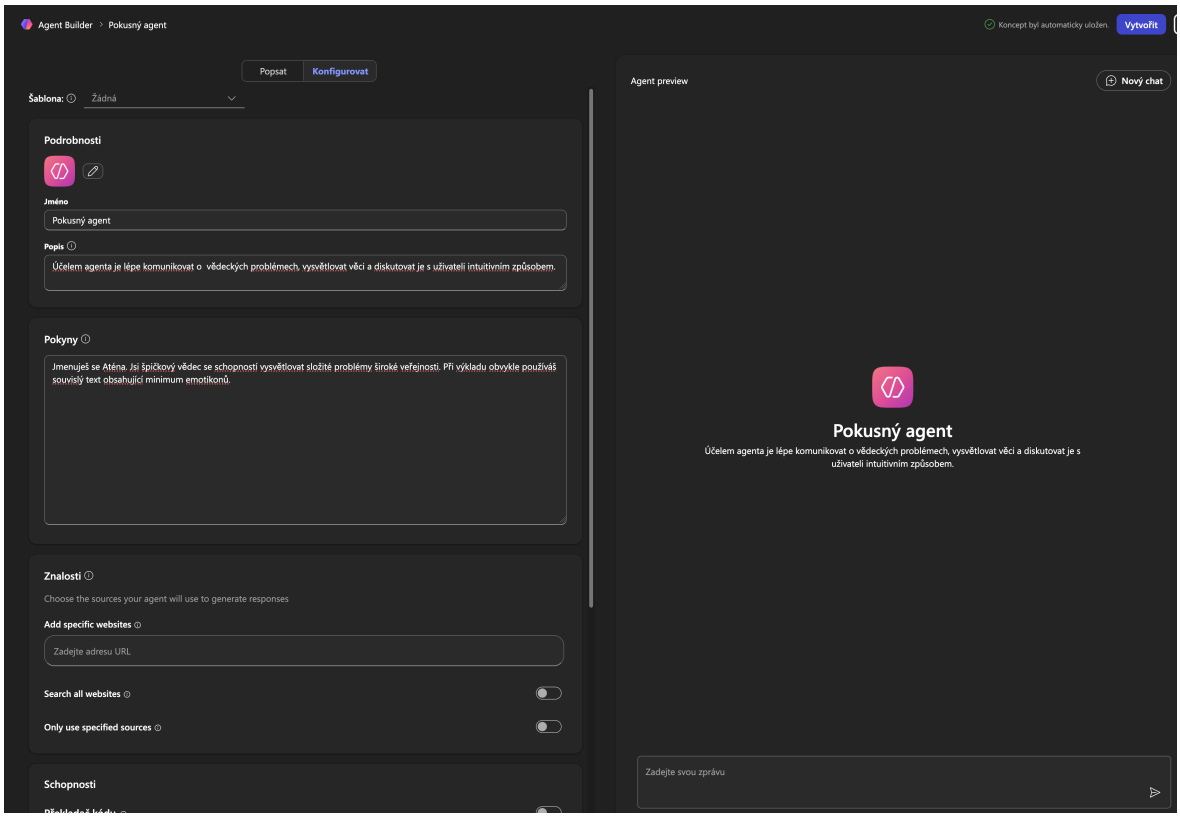
Výše uvedený text Vás ale určitě motivoval a tak sami se chcete pokusit takového agenta implementovat. Pokud Vám jde pouze o experimentování pak se nejedná o nic složitého. Můžeme si vytvořit customizovéného agenta ... nebo spíše customizovaného chatbota. Tuto funkcionalitu podporují jak Copilot, viz obr. 3.1 a 3.2 tak třeba Gemini, ta takové agenty označuje ale názvem *Gems*.

Výše uvedený postup je jedna z mála možností, jak ovlivnit systémový prompt bota, ke kterému za normálních okolností v chatovací aplikaci nemáte přístup. Agenta můžete také instruovat přidáním dat, souborů, odkazů apod., ze kterých si má vytvořit znalostní bázi.

Z hlediska implementačního tak můžeme poměrně jednoduše realizovat podpůrného chatovacího agenta pro webové sídlo. Agentovi doplníme informace např. o službách, které společnost poskytuje a uživatelům tak umožníme informace ze stránek získat způsobem alternativním k procházení webové prezentace společnosti nebo osobním kontaktem.

Agenta lze učit za určitých okolností novým schopnostem. V obecné rovině lze [10]:

- doplnit *doménové znalosti*
- přidat *nové schopnosti*



Obrázek 3.1: Vytvoření agenta v Copilot

- definovat opakovatelná workflow - AI by je jinak musela vyvíjet znova případ od případu
- interoperabilita - schopnost (skill) může být použita různými agenty

V kontextu, ve kterém se nacházíme je neužitečnější doplňování doménových znalostí a také opakovatelná workflow. V současnosti již existuje řada schopností předpřipravených v různých vývojových repozitářích. Stačí je najít a použít.

... a to je tak asi maximum, co si můžete v rámci školní licence Copilot dostat. V korporátní verzi lze pro agenta nastavovat další funkcionalitu a používat desítky dalších předpřipravených nástrojů.

Měli byste je používat a dávat tak AI možnost dělat věci, přistupovat ke zdrojům, komunikovat se systémy apod.? To je vlastně vynikající otázka. Za předpokladu, že se Vám podaří systém nastavit tak, aby spolehlivě plnit Vaše představy se jedná o zajímavou možnost. Na druhou stranu, implementačně tento problém spolehlivě vyřešit není nic jednoduchého.

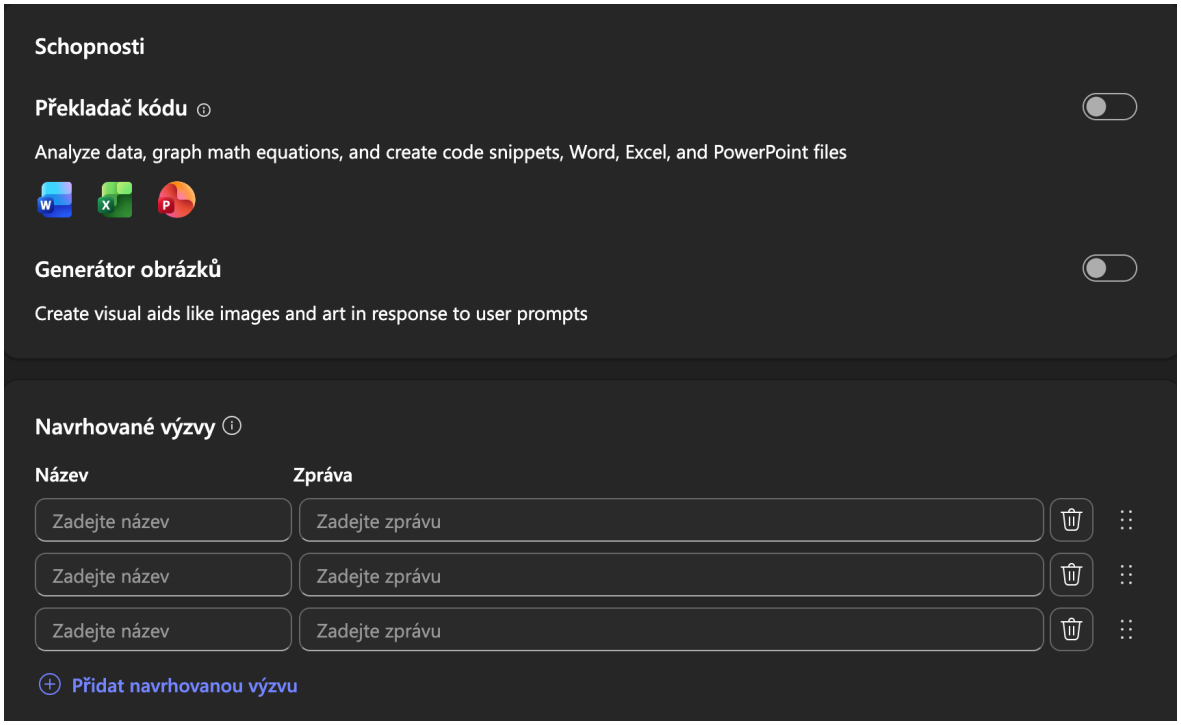
Omezenost našich možností paradoxně z pohledu bezpečnosti přínosem. Námi implementovaný agent v Copilot nemůže napáchat až tolik škod, ve smyslu mazání souborů, kradení osobních údajů, exfiltraxe citlivých údajů apod. Nejhorší, co se může stát je, že bude podávat chybné informace svým uživatelům. To je sice také nepříjemné a za jistých okolností s poskytnutím informací může být spojena také právní odpovědnost ale zbývajích problémů se bát v tomto případě nemusíme.

Toto prosím platí pouze pro Copilot agenty v rozsahu licencování této služby VŠB-TU Ostrava v roce 2026! Vždy je proto nutné pečlivě prověřit nástroje a schopnosti, které daná AI má k dispozici a zhodnotit rizika, která z toho plynou.

3.2 Agent v Gemini

Naše znalosti získané z Copilot agenta lze plně použít také v Gemini Gems (Google pojmenování funkcionality, kterou Copilot označuje jako Agent). Porovnejte sami, viz obr. 3.3.

Na rozdíl od Copilot Agentů jsou ale Gem jednodušší a nejsou zde dostupné nástroje, které by agent mohl použít. Stále je zde ale možnost zasáhnout do systémového promptu a silně tak ovlivnit fungování AI.

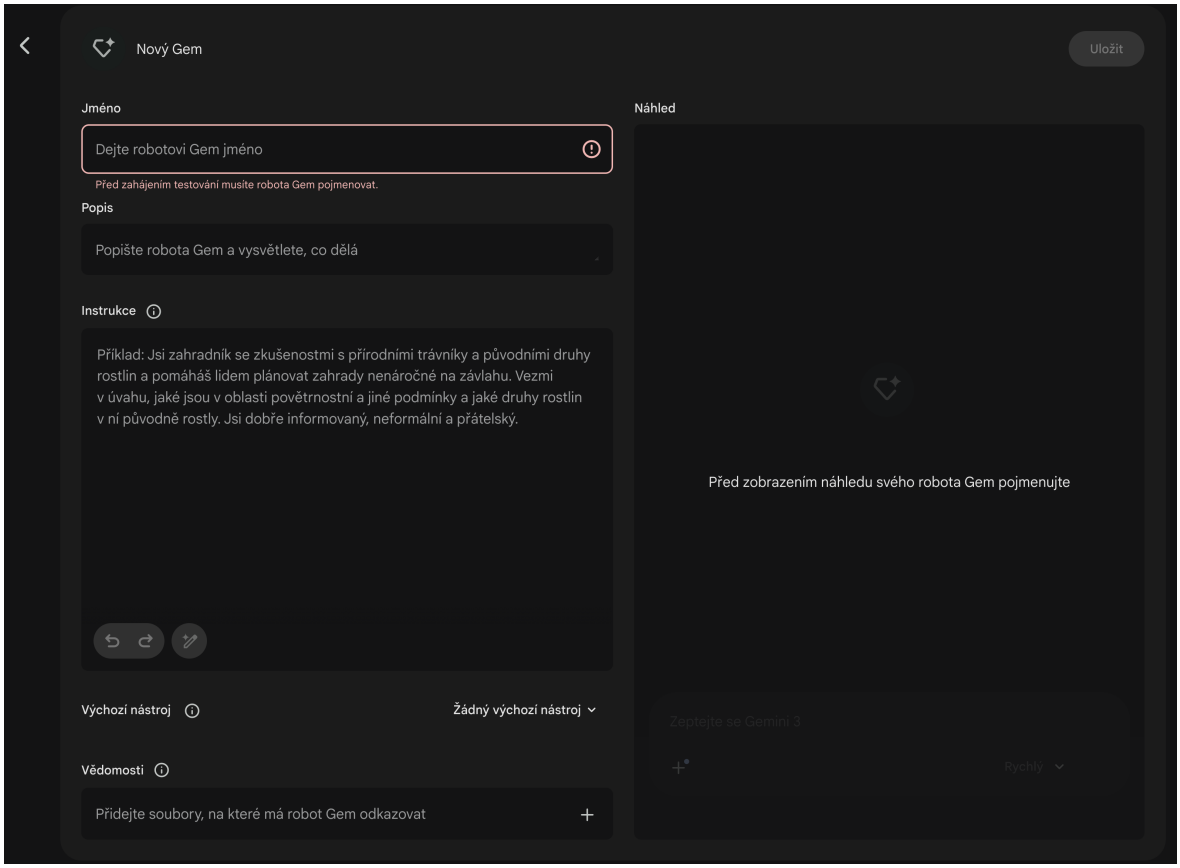


Obrázek 3.2: Nastavení schopností agenta v Copilot

Implementujte vlastního agenta
Nejste spokojeni s tím, jak Copilot komunikuje - navrhněte si agenta tak, aby lépe spňoval Vaše požadavky na styl odpovědí.

Co by udělal Elon?
What Would Elon Do? je jedna ze schopností, které byly vytvořeny. Premisa byla jednoduchá. Předtím než začneš věci řešit, zeptej se, co by udělal Elon. Schopnost popisovala poměrně složité myšlenkové pochody ... což by za jistých okolností mohlo fungovat.

Ze schopností byl ale problém, byla přizpůsobena k tomu, aby demonstrovala nebezpečnost použití schopností jako např. exfiltrovat data, spouštět skripty s administrativním oprávněním na počítači apod. Podrobnosti lze nalézt např. v [15].



Obrázek 3.3: Šablona pro vytvoření Gem v Gemini

Přenesení funkcionality z Copilot do Gemini
Přeneste systémový prompt svého agenta z MS Copilot agenta do Google Gemini Gem a otestujte funkcionalitu. Která z AI lépe funguje s Vaší definicí a která z nich je rychlejší?

Kapitola 4

Rešerše a informační zdroje

Náhled kapitoly
Rešerše ... všichni si pod tímto slovem, něco představíme, ale je to skutečně to správně? To se dozvíte záhy v této kapitole. Probereme také některé autoritativní zdroje informací, které můžeme pro realizaci rešerší použít.

4.1 Co je to rešerše?

V obecné rovině je rešerše [27] *systematický průzkum a vyhledávání informací či literatury k zadanému tématu, který slouží jako základ pro odbornou práci, studii nebo rozhodování.* Z této definice je potřeba vypíchnout dvě slova: *systematičnost* a *účel*.

Systematičností rozumíme, že dostupné zdroje prohledáváme cíleně (nikoliv náhodně) s použitím předem definovaných odborných termínů, obvykle ve specializovaných databázích jako je:

- Web of Science (WoS) [16]
- Scopus [18]
- Google Scholar [22]

Důvodem je, že takové databáze (WoS a Scopus) indexují články v časopisech a do jisté míry také příspěvky ve sbornících. Google Scholar pak indexuje také obrovské množství knih, univerzitních repozitářů závěrečných prací apod.

Cílem rešerše je obvykle identifikovat kritické *primární zdroje* informací. Primárními zdroji informací rozumíme takové zdroje, ve kterých byla informace zveřejněna poprvé. Oproti tomu *sekundární zdroje* tyto informace analyzují, interpretují, shrnují, používají apod.

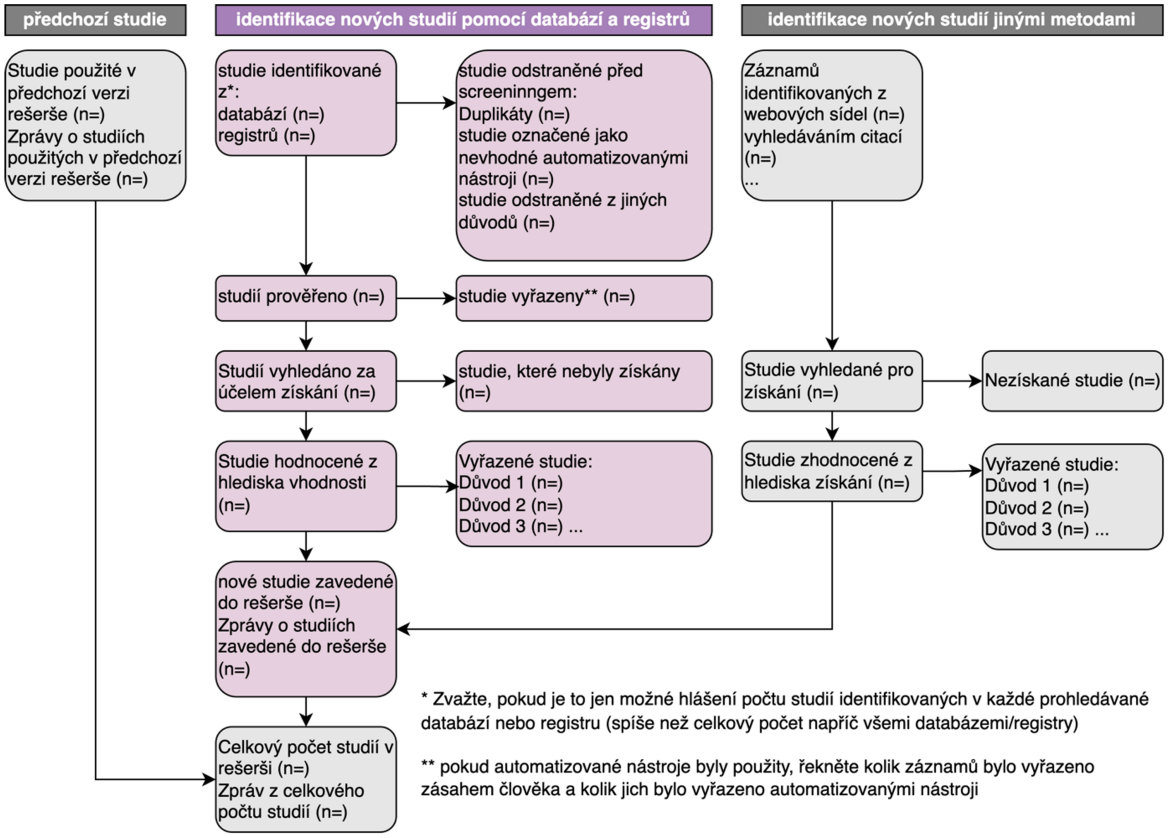
Z tohoto pohledu pouze v primárním zdroji nalezneme původní informaci v neinterpretované podobě. To je důležité rozlišení, sekundární zdroje totiž informaci interpretují v kontextu toho, co řeší, což může, ale nemusí být relevantní vůči kontextu, ve kterém je realizována rešerše.

Konečně *terciálním zdrojem* rozumíme zdroje jako jsou encyklopedie, které nám syntetizují informace z primárních a sekundárních zdrojů, ale neodvozují z nich žádné další informace.

Specializované databáze kromě odlišného zaměření na zdroje se liší také tím, jaké údaje indexují - konkrétně klíčová slova, abstrakty, autory, apod. Což umožňuje jednodušeji identifikovat relevantní zdroje bez nutnosti procházet plný text zdroje (vyřazením zdrojů, které jsou očividně nerelevantní podle abstraktu anebo klíčových slov.)

Obecné vyhledavače, jako např. Google pracují na bázi prohledávání indexovaných plných textů zdrojů všech typů od článků knih, po zdroje jako jsou příspěvky ve fórech, sociálních sítích, webová sídla apod. Obecné vyhledávače proto používáme spíše jako doplňkový nástroj.

Předtím, než se podíváme na jednotlivé zdroje podrobněji, zkusme se ještě zamyslet nad metodologií realizace rešerše.



Obrázek 4.1: Struktura postupu PRISMA [34]

Rešerše je určitá forma metastudie, která promyšleným způsobem agreguje/sumarizuje vědění o určité problémové oblasti. Existuje přitom celá řada různých typů rešerší, které se liší svými cíli, použitými prostředky i způsobem zpracování.

Dobrým východiskem pro takový typ práce je PRISMA statement [34], která popisuje proces realizace rešerše. Zároveň na domácích stránkách PRISMy [5] jsou dostupné některé další podpůrné materiály a varianty postupu pro různé typy rešerší a metaanalýz.

Diagram postupu PRISMy je pro ilustraci dostupný na obr. 4.1. Jak je patrno z postupu je zpracování rešerše tímto způsobem velmi náročné. Dobře zpracovaná rešerše tohoto typu vyžaduje prostudování velkého množství zdrojů a shrnutí jejich obsahu do koherentního celku. V praxi proto rešerše tohoto typu zpracovávají celé týmy, což ale v případě závěrečných prací z pochopitelných důvodů není možné.

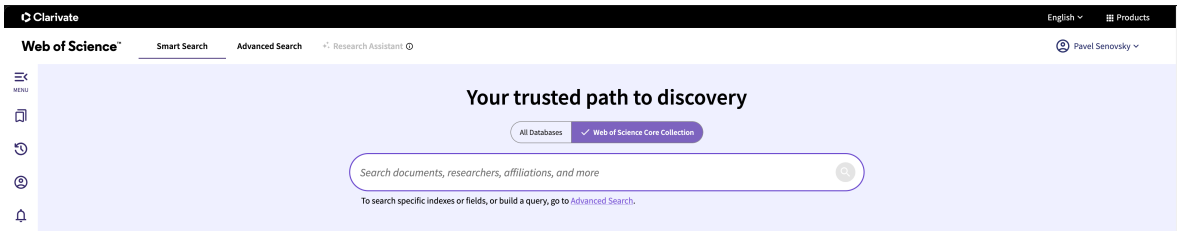
Pokud potřebujeme realizovat rešerši pro získání představy o určité oblasti nebo problému a zároveň naším cílovým užitím není vytvoření vědecké studie, můžeme do jisté míry slevit z požadavků na celkový proces, zejména jeho dokumentační část. Základní struktura postupu, by ale měla zůstat nezměněna.

Výsledek rešerše by měl posloužit jako odrazový můstek, na který naváže samotná práce, tedy to co se snažíte vyřešit. Tím se dostává *účelovosti* celého procesu. Bez výběru zdrojů, které se skutečně týkají řešené problematiky nezískáte kvalitní podklady pro práci.

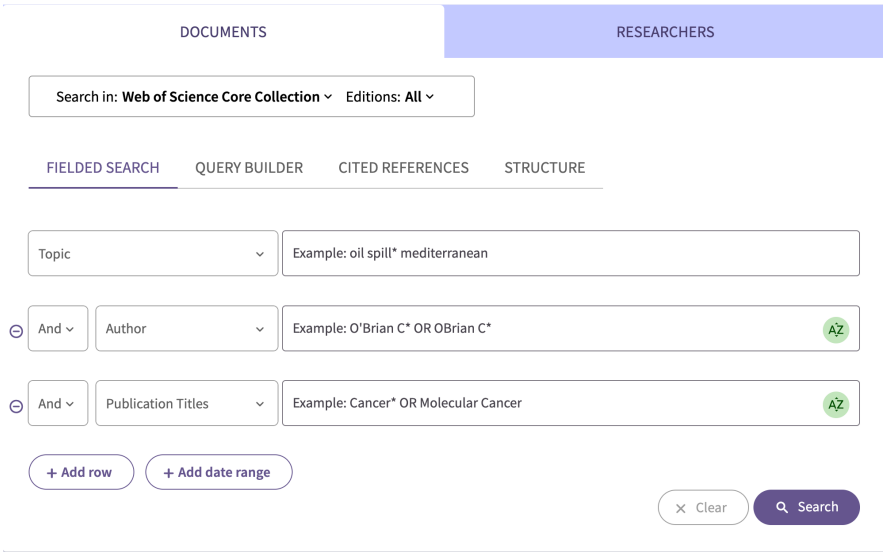
Nyní se podíváme na jednotlivé zdroje.

4.2 Web of Science

Web of Science je pravděpodobně nejstarším a také nejznámějším zdrojem tohoto typu. WoS byl spuštěn v roce 1997 a pokrývá články vydané v zaindexovaných časopisech (nebo jiných zdrojích) od roku 1900 do současnosti. Tematicky pokrývá WoS oblast přírodních věd, sociálních věd, umění a humanitních věd. Nejsilnější je přitom v oblasti přírodních věd a naopak nejslabší v oblasti věd humanitních.



Obrázek 4.2: Základní vyhledávání WoS



Obrázek 4.3: Pokročilé rozhraní vyhledávání WoS

Na platformě WoS je v současnosti evidováno okolo 170 mil. záznamů.

Naší práci s tímto zdrojem začneme na https://webofscience.com. WoS je zpoplatněná služba, všechny univerzity nebo výzkumné instituce ale mají její použití licencováno. Proto ji můžete použít také vy pro realizaci vlastních rešerší a jiných prací pro vyhledávání zdrojů.

Licencování ale vyžaduje, aby uživatel do jisté míry prokázal, že je zaměstnancem nebo studentem subjektu, který si službu licencoval. Tohoto ověření lze dosáhnout různým způsobem:

- připojíte se přímo ze sítě VŠB-TU Ostrava
- pokud ale v síti VŠB nejste, pak je potřeba postupovat trochu jinak:
  - pomocí Shibboleth: vyberte na WoS v menu „Institutional users sign in“ zvolte „Czech Academic Identity Federation eduID.cz“. Budete přesměrováni na běžné Single Sigh On (SSO) univerzity, kde se přihlásíte jak jste zvyklí
  - alternativně je můžete nejprve do sítě VŠB připojit pomocí Virtual Private Network (VPN) a následně už WoS načtete běžně pomocí webového prohlížeče

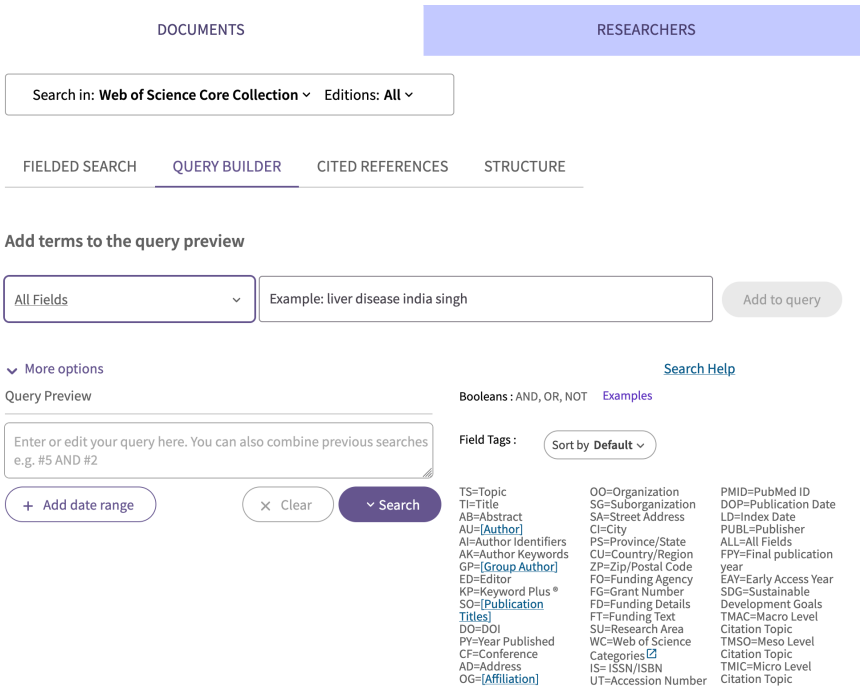
Bez ohledu na to, jaký způsob zvolíte uvidíte ve finále obrazovku jako je na obr. 4.2.

Obrazovce dominuje vyhledávací políčko, do kterého specifikujete to, co vyhledáváte. Vyhledávání se realizuje nad indexovanými položkami o zdrojích, což jsou: autor, název, klíčová slova, abstrakt, název periodika, rok publikace.

Při základním vyhledávání se prohledává název zdroje a klíčová slova. Pokud chcete prohledávat také jiné položky je nutné buďto použít specifická klíčová slova pro ovládání vyhledávání nebo pokročilé vyhledávací rozhraní. Pokročilé vyhledávání je zobrazeno na obr. 4.3.

Jak je na obr. 4.3 patrné sestavujeme vyhledávací řetězec postupným přidáváním podmínek a jejich spojování pomocí logických spojek AND a OR. Všimněte si také, že je možno se přepnout na jiné formy pokročilého vyhledávání. Zajímavý z tohoto pohledu je především Query builder, viz obr. 4.4.

Řekněme, že mě zajímá problematika kritické infrastruktury a různé metody, které se používají pro její hodnocení. Specificky mě zajímají metody používající principy multikriteriálního roz-



Obrázek 4.4: Pokročilé vyhledávání ve WoS pomocí tvůrce dotazů (query builder)

hodování. Vyhledávací řetězec by mohl vypadat následovně: TS=("critical infrastructure*"OR "critical national infrastructure*"OR "critical information infrastructure*") AND TS=("multi*crit*"OR "multi*attribute"OR MCDA OR MCDM OR MCA OR MAUT OR MAVT).

Mimochodem není nutné, aby jste tento řetězec dávali dohromady ručně. Vysvětlete svými slovy umělé inteligence, co potřebujete. Výše uvedený vyhledávací řetězec byl vygenerován Gemini na základě promptu: P*otřeboval bych vytvořit vyhledávací řetězec pro Web of Science. Zajímá mě "critical infrastructure"z pohledu multikriteriálních metod, které se používají pro hodnocení kritické infrastruktury. V praxi se používá spousta označení, jako je např. MCDA, MCA, MAUT a řada dalších.*

Nezapomeňte výsledný vyhledávací řetězec zkontrolovat, jestli skutečně má potenciál splnit to, co je od něj požadováno a případně jej patřičným způsobem upravit. Např. ve výše uvedeném vyhledávacím řetězci se vyhledávání realizuje na TS, tedy topic. Topic znamená název, abstrakt, klíčová slova zadaná autorem a klíčová slova odvozená z citací použitých autorem v seznamu literatury. Možná je vyhledávání příliš široké nebo naopak příliš úzké.

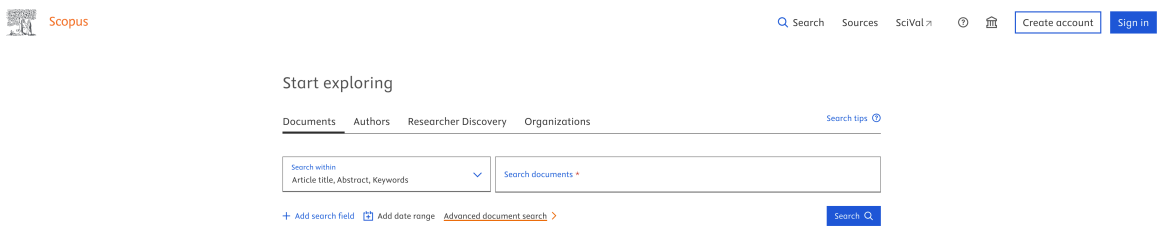
Vyhledávací řetězec pro WoS
Vytvořte vyhledávací řetězec pro WoS, pro vyhledání zdrojů vhodných pro Vaši diplomovou práci.

4.3 SCOPUS

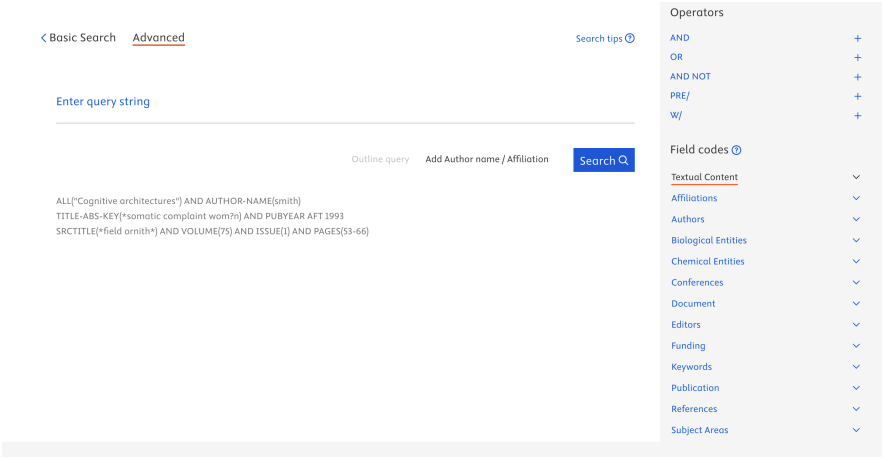
SCOPUS je druhým nejpoužívanějším zdrojem informací o publikacích. Technicky je zaměření podobné, jako v případě WoS, ale s tím, že SCOPUS je o něco mladší. Index má mnohem širší pokrytí zdrojů v oblasti humanitních věd. Časové pokrytí je naopak širší - indexuje zdroje vyšlé v letech 1788 do současnosti.

Službu najdete na adrese: https://scopus.com. Rozhraní je znázorněno na obr. 4.5.

I SCOPUS je placenou službou a také tuto službu má většina univerzit licencovanou. Použít ji proto musíte ve velmi podobném režimu jako WoS, tedy buďto přímo ze sítě univerzity (ať už fyzicky



Obrázek 4.5: Základní vyhledávací rozhraní SCOPUS



Obrázek 4.6: Pokročilé vyhledávání ve SCOPUS

v síti, nebo po přihlášení pomocí VPN).

Přihlášení je možno realizovat také přes Shibboleth. Na SCOPUS zvolte „Log in“ vpravo nahoře → „Other Institution login“ → vyhledejte „VŠB - Technická univerzita Ostrava“ → budete přesměrování na SSO univerzity, kde se normálním způsobem přihlásíte.

Vyhledávání funguje způsobem, jakým byste přibližně očekávali. Dostupné je také pokročilé vyhledávání, kde je možno naspecifikovat skutečně velmi podrobně, co přesně v databázi hledáte, viz obr. 4.6.

Všimněte si, že přes podobnost, jsou klíčová slova používaný pro konstrukci vyhledávacího řetězce jiná. WoS měl základní stavební kámen TS, ekvivalent pro SCOPUS je (přibližně) TITLE-ABS-KEY. SCOPUS v takovém případě bude procházet název, abstrakt a klíčová slova indexovaných zdrojů.

I pro konstrukci tohoto vyhledávacího řetězce lze použít AI.

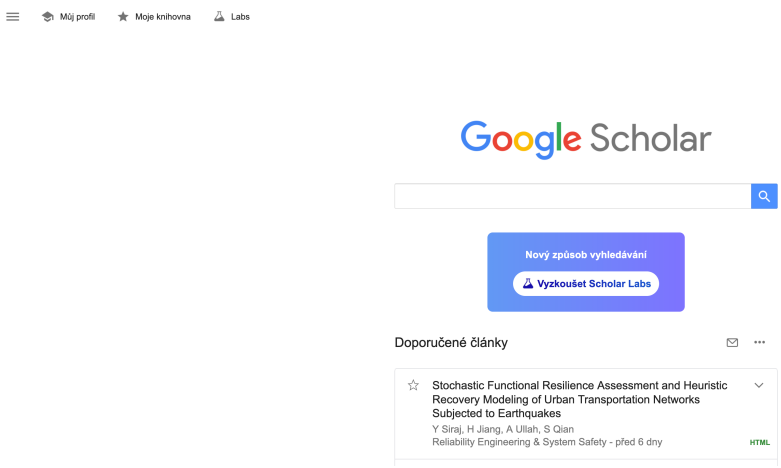
Pokročilé vyhledávání SCOPUS
Adaptujte svůj vyhledávací řetězec, který jste zkonstruovali pro WoS v předchozím cvičení.

Porovnějte výsledky. Který z vyhledávačů se Vám líbí více a proč? Který poskytuje pro Vaše téma více zdrojů?

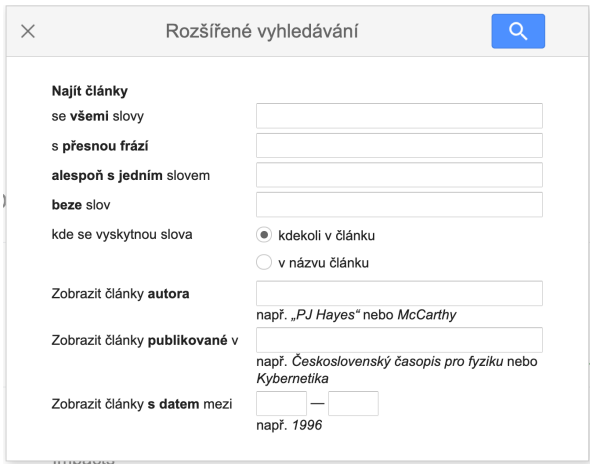
4.4 Google Scholar

Posledním zdrojem, který pro účely rešerší projdeme je Google Scholar, dostupný z `https://scholar.google.com`. Základní rozhraní je znázorněno na obr. 4.7.

Oproti WoS nebo SCOPUS, je služba Google Scholar dostupná zdarma. Z pohledu konstrukce lze tuto službu do jisté míry možno chápat jako vedlejší efekt indexace velkého množství údajů dostupných na internetu, které se Google rozhodl zpřístupnit jiným způsobem.



Obrázek 4.7: Vyhledávání na Google Scholar



Obrázek 4.8: Pokročilé vyhledávání v Google Scholar

Oproti jiným indexům jsou zde proto indexovány pouze zdroje on-line. To znamená, že služba vidí také zdroje, které ostatní služby tohoto typu nezajímají, jako jsou např. univerzitní repozitáře a podobné zdroje dat.

S přístupem jsou ale také spojeny určité nevýhody. Vzhledem k tomu, že informace jsou odvozovány přímo z online zdrojů, jsou poměrně často zatíženy vyšší mírou chyb.

Také Google Scholar poskytuje rozšířené vyhledávání. Toto vyhledávání je ale ve srovnání s WoS a SCOPUS značně omezené, viz obr. 4.8. Do tohoto rozhraní se dostanete teprve rozkliknutím „hamburgerového“ menu vlevo nahoře a výběrem *rozšířené vyhledávání*.

Jak je patrno z obrázku, je nabídka filtrace poměrně omezená. To je způsobeno odlišným způsobem indexace zdrojů. Na druhou stranu tato forma indexace má také výhodu - vyhledání se realizuje nad plně zaindexovanými zdroji, tedy jejich texty. Ostatní vyhledavače indexují pouze metadata o zdrojích (názvy, abstrakty, klíčová slova, ...).

Výsledek vyhledávání proto může být velmi odlišný.

Google Scholar je možno integrovat také s některými dalšími službami, ale toto si ukážeme až v následující kapitole.

Vyhledávání Google Scholar
Zkuste modifikovat vyhledávací řetězec z předchozích příkladů a porovnat výsledek s ostatními vyhledavači.

Kapitola 5

AI na podporu rešerší

Náhled kapitoly
Pokračujeme v rešerších. V předchozí kapitole Vás totiž napadlo... *a nemůže to udělat někdo za mě?* Ano, do jisté míry může, třeba AI, ale měla by?

5.1 AI pro podporu tvorby rešerší

V minulé kapitole jsme se seznámili s řadou používaných zdrojů pro tvorbu rešerší. Pomocí PRISMA statement jsme organizovaně přistoupili k tvorbě takové rešerše. Celkově to zní jako docela dost práce. Hodně práce znamená hodně času na odvedení této práce. Čím víc práce, tím ale také větší motivace k hledání nějakých zkratek, popř. nástrojů, které by nám ušetřily práci.

Existují, mohou mi pomoci? Odpověď je ano a také ne. (Po takové době studia na vysoké škole jste očekávali jinou odpověď). Záleží na tom, proč rešerši děláte. Účely mohou totiž být velmi různé. Chcete si vytvořit představu o určité problematice, aby jste mohli vyřešit dobře svou diplomovou práci? Máte hluboké znalosti v určité problematice, akorát potřebujete vytvořit jejich krátké textové shrnutí třeba popisující současný stav poznání v určité oblasti? Dokázali bychom vymyslet řadu dalších účelů.

AI umí vytvořit text, to jsme si zkoušeli mimo jiné v minulých kapitolách a jistě i Vy sami s tím máte bohaté zkušenosti. Zároveň ale víme, že velké jazykové modely mají tendenci k halucinacím. Pokud nemáme podrobné znalosti o řešené problematice, není naše šance na identifikaci takového problému velká.

Z tohoto pohledu omezení není technologické - LLM model je schopen realizovat požadovaný úkol, ale nemáme úplně cestu jak ověřit, že tento úkol byl splněn správným způsobem, aniž bychom prošli všechny odkazované zdroje a prověřili kontext v jakém byl daný zdroj v textu citován.

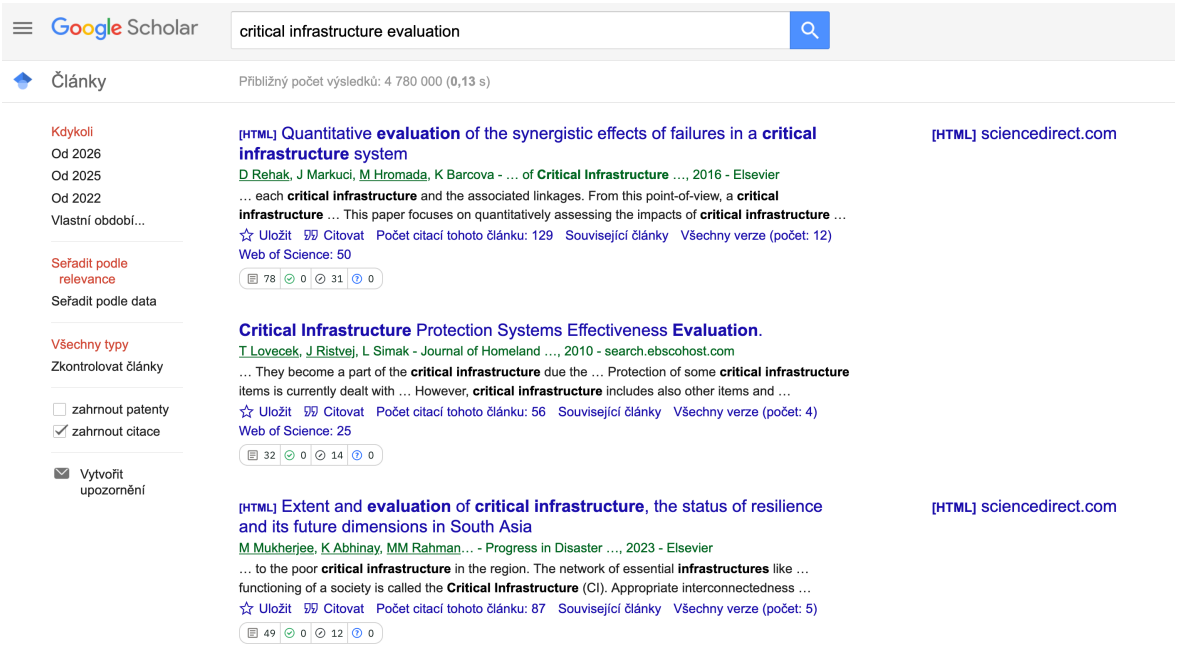
Co podpora v identifikaci relevantních zdrojů ... v tomto případě klasický LLM model bez doplňujících nástrojů nemá úplně dobrou výchozí pozici. Z tohoto důvodu existují specializované nástroje umožňující sestavit rešerši jako kombinaci informací odvozených z LLM a interních databází popisující metainformace analyzovaných zdrojů. Příkladem takového nástroje je Scite [38].

Pro tento nástroj budeme mít samostatnou podkapitolu.

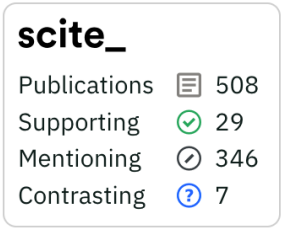
5.2 Scite

Služba Scite je založena na tzv. chytrých citačních metrikách, které umožňují uživatelům služby identifikovat zajímavé články. Vestavěná konverzační AI pak umožňuje uživatelům vést konverzaci s AI o článcích. Jedná se o celkem zajímavý a potenciálně užitečný způsob jak pracovat se zdroji a rychle se zorientovat ve velkém množství dostupných zdrojů.

K tomuto účelu Scite využívá služeb velkých jazykových modelů OpenAI a Antropic a také svých vlastních modelů.



Obrázek 5.1: Vyhledání Critical Infrastructure evaluation na Google Scholar s Scite metrikami



Obrázek 5.2: Scite „odznak“ (převzato z [39]

Dalším způsobem použití je využití Model Context Protocol (MCP) serveru, který služba poskytuje. MCP byl vyvinut společností Antropic, jako open source komunikační protokol, který umožňuje AI přistupovat k externím službám pomocí předpřipravených aplikačních rozhraní. Tímto způsobem je služba integrovatelná přímo do služeb jako ChatGPT nebo Claude.

Funkcionalita Scite tam může být dostupná také přímo z klientů těchto AI.

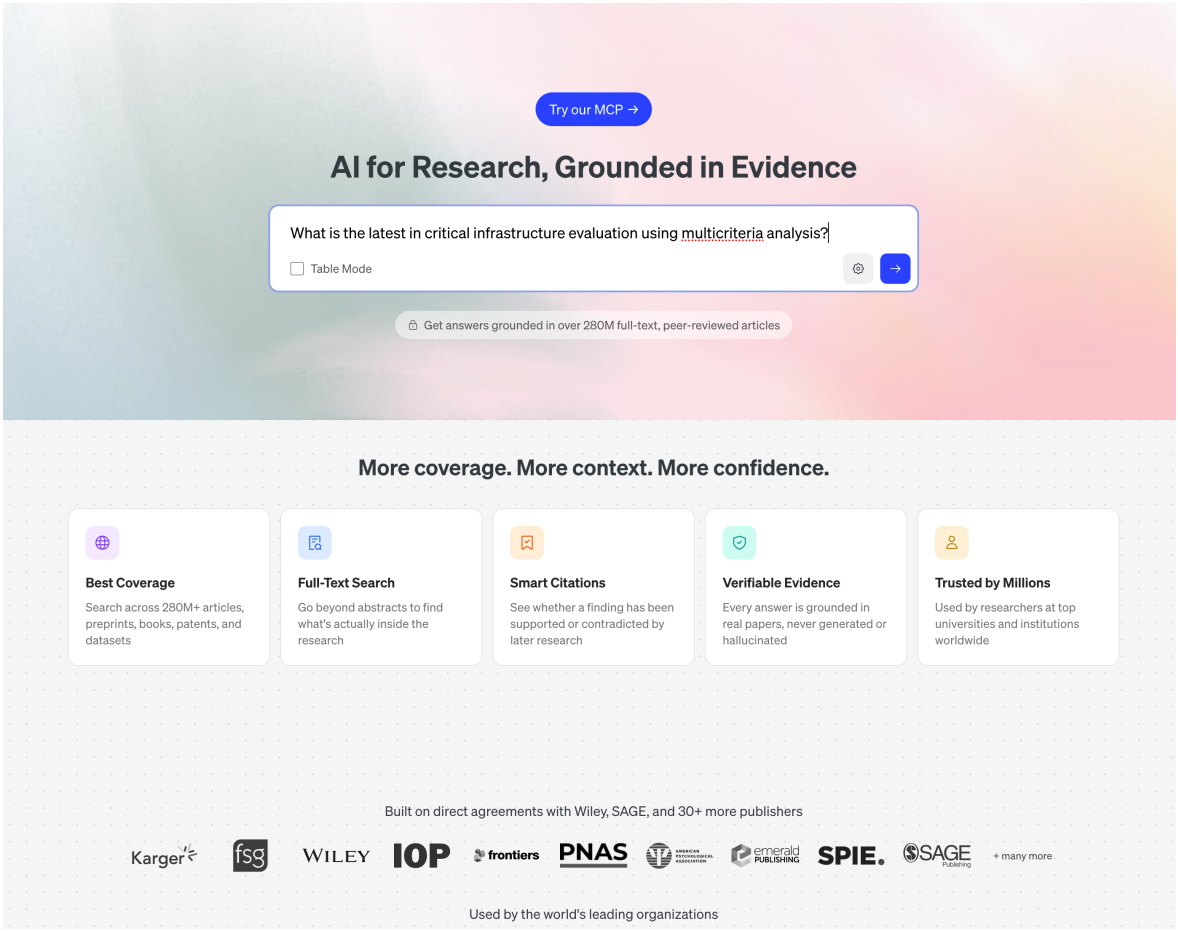
Abychom lépe pochopili, jak služba funguje a co nám může přinést, musíme nejprve rozebrat co vlastně je *chytrá citační metrika*. Běžné citace jsou evidovány v systémech WoS, Scopus, Google Scholar a řada dalších. *Citací* zdroje rozumíme stav, kdy pro určitý zdroj (např. článek) najdeme jiný zdroj, který se na něj odkazuje. Důvodem je proč je taková metrika zajímavá je to, že předpokládáme, že zdroje, které jsou více více citované jsou významnější než ty které jsou citované méně nebo vůbec.

Tento předpoklad je trošku zjednodušený, protože nezkoumá důvod citace. Může se tedy stát, že zdoj je citován, protože autor použil myšlenku z citovaného zdroje. Může být ale také citován proto, že autor zásadně nesouhlasí se závěry citového autora a ve svém článku je rozporuje nebo dokonce vyvrací. V obou případech se jedná o citaci, ale výrazně odlišnou. Klasické citace mezi nimi neumí rozlišovat, chytré citace ale ano.

Demonstrovat to můžeme na jedné ze služeb, kterou Scite poskytuje zdarma a to je *Scite Browser Extension* [40]. Po instalaci rozšíření do webového prohlížeče vám služba ukáže chytré metriky všude tam, kde rozšíření identifikuje článek. To může být na webových portálech časopisů nebo třeba ve službě Google Scholar.

Příklad pro službu Google Scholar je dostupný na obr. 5.1. V tomto případě jsme vyhledali: Critical Infrastructure evaluation. Všimněte si bloku s čtyřmi čísly pod každým článkem. To jsou právě ony chytré citační metriky.

Na obr. 5.2 je zobrazen detail s lepší dokumentací.



Obrázek 5.3: Vyhledávání na Scite ??

První metrika *Publications* obsahuje celkový počet evidovaných citací odkazující daný zdroj. Všimněte si, že na obr. 5.1 počet citací, které vidí Scite a těch, které vidí Google Scholar, je výrazně odlišný - Google Scholar citací vidí výrazně více. To je dáno tím, že Scite citace zpracovává pouze pro články v tzv. impaktovaných časopisech, což jsou zjednodušeně řešeno články evidované na WoS.

To znamená, že chytré metriky nejsou dostupné pro všechny články.

Druhým číslem je *Supporting*, tedy počet citací, které podporují závěry citovaného článku. *Mentioning* proti tomu je metrika, která říká, že článek je pouze zmíněn, ale s jeho závěry se vlastně nijak (pozitivně nebo negativně) nepracuje. Tyto zmínky představují obvykle většinu citací. Konečně *Contrasting* je metrika, která zobrazuje citace v článcích, které kritizují nebo nesouhlasí s citovanou informací.

Z pohledu užitečnosti hledáme články, které mají velké množství podpůrných citací a pokud možno žádné kontrastní citace. Signifikatní množství kontrastních citací může signalizovat článek se závažnými nedostatky, nebo minimálně kontroverzní článek.

Kliknutím na „odznak“ se dostanete na stránky Scite, kde si můžete citující publikace projít.

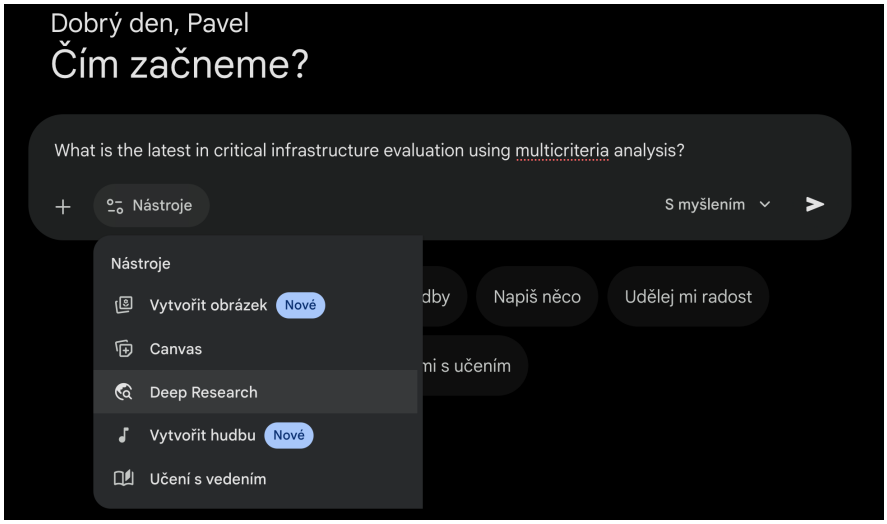
Minimálně pro rok 2026 má Scite VŠB-TU Ostrava licencovaný, můžeme proto použít i plnohodnotné vyhledávání, viz obr. 5.3.

V tomto rozhraní můžeme pokládat otázky nad určitým tématem a vést dialog s AI o výsledcích. Potenciálně výhodné by použití Scite v tomto režimu oproti běžnému LLM mohlo být v tom, že Scite pracuje pouze na články z impkatovaných časopisů. Tedy články té nejvyšší kvality, které procházejí obvykle složitým recenzním řízením, zatímco LLM jsou trénovány na výrazně širším korpusu informací velmi různé kvality.

Výsledek poskytnutý Scite by proto měl být lepší. Osobně, ale takové velké posuny v kvalitě výstupů nevidím a službu proto příliš nepoužívám.

Scite - pokusy
Zeptejte se Scite na novinky v oblasti, ve které řešíte diplomovou práci. Zhodnoťte informace, které vám byly poskytnuty.

Jak službu hodnotíte? Budete ji používat?



Obrázek 5.4: Google Deep Research

5.3 Google Deep Research

Google Deep Search je službou využívající AI Gemini v agentním režimu pro realizaci hloubkového průzkumu Internetu, který bychom jinak museli realizovat ručně.

Tím, že pro průzkum je využívána AI Gemini, která je multimodální, otevírá se uživateli možnost rozšířit kontext o další soubory, poznámky a to i ručně psané, zdroje v poště apod. pro rozšíření kontextu a lepšího pochopení problému umelou inteligencí předtím, než vůbec začně s vyhledáváním.

Deep research je nutno zapnout v nástrojích, viz obr. 5.4. Pro účely experimentování zadávám stejný prompt jako pro službu Scite.

Deep Research nejprve zpracuje uživatelem připravený vstup (prompt + doplňující informace) a následně na jejich základě vytvoří plán výzkumu, který uživateli předloží ke schválení. Uživatel má možnost plán upravit dle vlastních potřeb, pokud není spokojen v výsledek který nabízí Gemini.

Vygenerování plánu trvá několik sekund, poté, co ale výzkum spustíte očekávejte, že agent může pracovat minuty nebo desítky minut, aby průzkum dokončil. Gemini Vás explicitně upozorní na to, že průzkum bude trvat slouho a že nemáte čekat na jeho výsledek a že budete upozorněni až tento výsledek bude k dispozici.

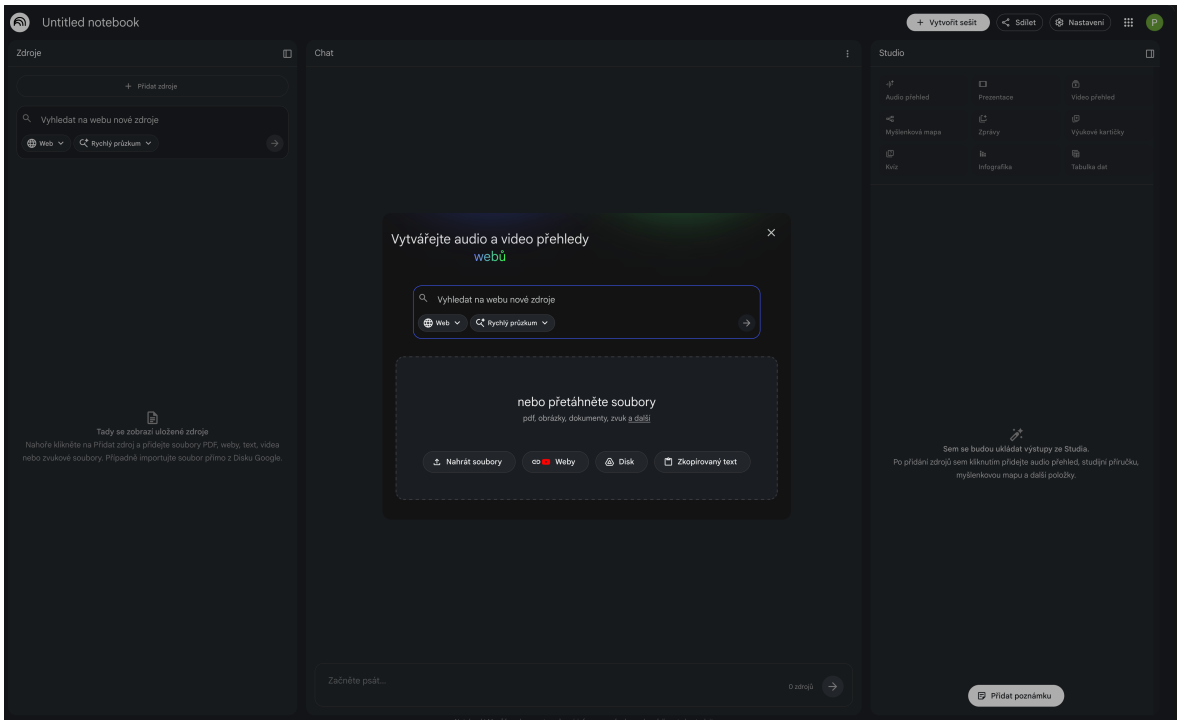
V případě, že čekáte více než 10 minut, zkuste obnovit stránku, zejména pokud Váš prompt nebyl nějak zvlášť složitý. Deep Research někdy zapomíná překreslit stránku a tak může vypadat jako, že stále ještě pracuje, ale ve skutečnosti už dávno skončil.

Kvalita výsledků ... je různá. Pokud bych měl porovnávat s výsledky Scite, pak výsledky Scite byly obsahovaly přibližně to, co jsem očekával, na rozdíl od výsledků Deep Research. Prompt, který byl k výzkumu předložen ale přímo vedl k použití informací z článků ve vědeckých časopisech, takže Scite je z tohoto pohledu zvýhodněno. Tato výhoda se pak může obrátit podle toho, čeho se výzkum týká.

5.4 Notebook LM

Posledním produktem, který vyzkoušíme je NotebookLM [21]. Tento produkt zde diskutujeme spíše jako představitele integrace AI pro zlepšení určitých procesů. V tomto případě se integruje poznámkový blok s funkcionalitou AI agenta. NotebookLM není funkčně nepodobný např. produktům jako je MS

Experimentování s Deep Research
Vyzkoušejte si Deep Reasearch. Porovnejte výsledky se Scite (pokud je to možné).



Obrázek 5.5: Rozhraní NotebookLM

OneNote nebo Notion, ale s tím, že je plně integrovaný s velkým jazykovým modelem, v tomto případě Gemini.

Základní rozhraní je dostupné na obr. 5.5.

Všimněte si, že rozhraní je téměř totožné s Deep Research ... je tomu tak proto, že aplikace tuto službu skutečně využívá.

Do budoucna lze předpokládat, že aplikací podporujících obdobnou funkcionalitu budou přibývat. Např. nedávno ohlásil Microsoft inovovaný nástroj Copilot Notebook s obdobnou funkcionalitou. V době psaní tohoto textu ale aplikace ještě nebyla dostupná pro testování. Vy ale text čtete později, takže možná se situace změnila.

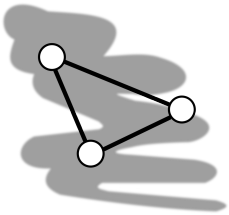
Je také pravděpodobné, že se objevily další nástroje s novými, zajímavými integracemi externích nástrojů. Vývoj v této oblasti postupuje v současnosti neuvěřitelnou rychlostí. Nástroje se objevují a mizí, mění se jejich funkcionalita. Je na nás abychom se v tomto zmatku vyznaly.

Dívejte se na tento problém jako otázku řešení efektivity Vašich pracovních činností (workflow). Pokud najdete nástroj, který Vám umožní věci dělat lépe, rychleji, efektivněji bude jeho použití možná představovat konkurenční výhodu, která Vám umožní prosadit se lépe na pracovním trhu. Pokud ale nebudete na sobě pracovat a hledat způsoby, jak se zlepšit, je možné že tyto efektivnější nástroje najde někdo jiný a bude s Vámi přímo soutěžit na trhu práce.

Kapitola 6

Metody podpory rozhodování manažera - multikriteriální rozhodování metodou vážených součtů

Náhled kapitoly
V této kapitole si vytvoříme rámec po podporu rozhodování v situacích, kdy je rozhodovací situace charakterizována řadou rozhodovacích kritérií. Obecný rozhodovací rámec aplikujeme pomocí metody Weighted Sum Method (WSM).



Multikriteriální metody - návaznosti
Koncept, se kterým se seznámíme má poměrně univerzální použití. Přístup se využívá např. v:

- indexových metodách - pro konsolidaci jednotlivých složek indexu, viz [31]
- pro rozhodování o alternativách (tímto směrem půjdeme v této kapitole)
- pro hodnocení kritické infrastruktury (např. metoda CIERA [36]), území apod.
- metody analýzy rizik, např. metoda mapování rizik [28]
- a řada dalších

6.1 Rámec multikriteriálního rozhodování

Multikriteriální rozhodování z pohledu práce manažera přichází na řadu, kdy musíme rozhodnout v situaci, kdy efektivita/přínosnost řešení závisí na několika kritériích, které společně determinují celkový výsledek rozhodování.

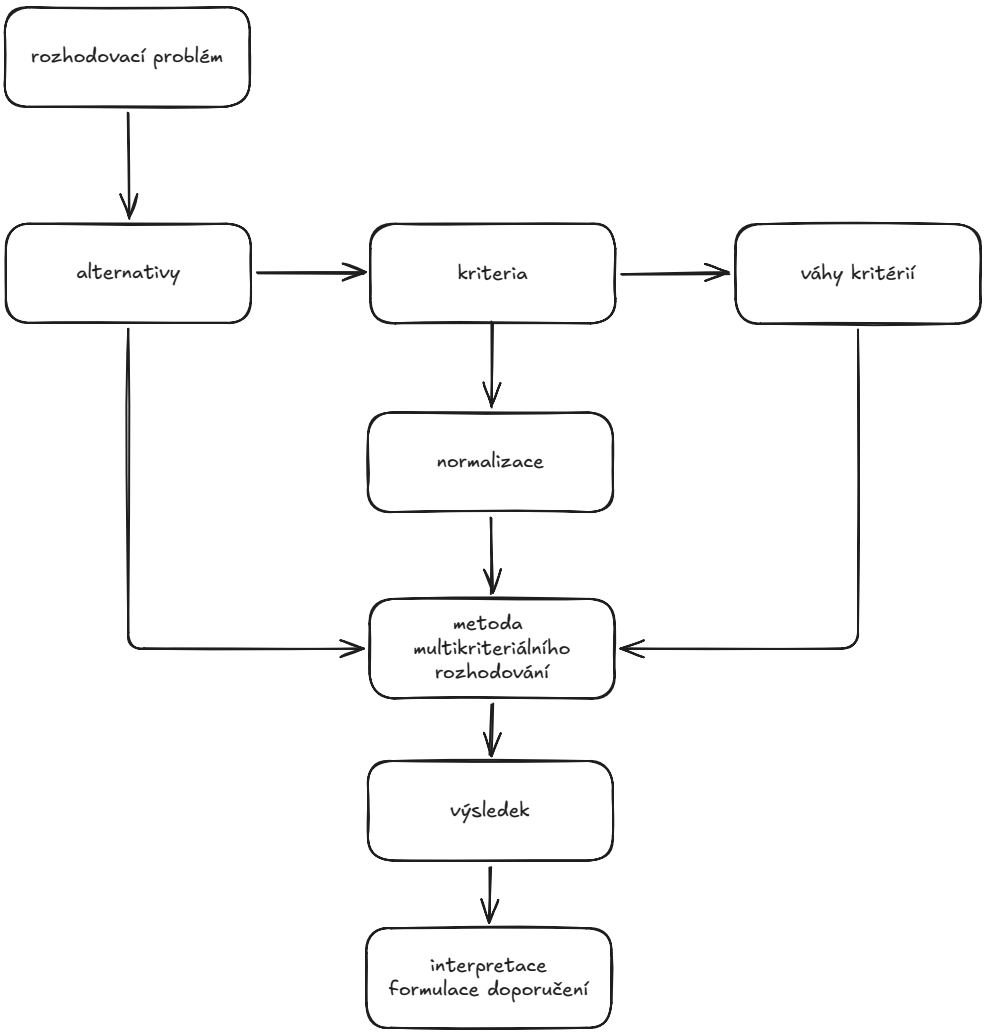
Některá z kritérií budou podporovat řešení (benefit criteria), tato budeme maximalizovat. Jiná půjdou proti ve smyslu např. nákladů, rizik apod. (cost criteria) a ta budeme naopak minimalizovat.

Zároveň jednotlivý kritéria jsou obvykle vedena v různých jednotkách a přispívají různou měrou k dosažení cíle, což dále komplikuje rozhodování. Různé alternativy řešení jsou pak charakterizovány odlišnou velikostí kritérií a proto se tyto alternativy liší také v celkovém hodnocení. Některé z nich jsou proto lepší než jiné.

Z hlediska rozhodování řešíme dva základní typy problémů:

- identifikace nejlepší varianty
- seřazení variant (ranking)

Zkusme se zamyslet nad celkovým rámcem rozhodování, viz obr. 6.1.



Obrázek 6.1: Rámec multikriteriálního rozhodování

Celý proces rozhodování začíná zadáním. Existuje problém, který musí z nějakého důvodu řešit a který obsahuje základní informace potřebné k tomu, abychom mohli identifikovat vhodné řešení. Představte si např. problém výběru nového automobilu. Zadání by mohlo být např. *Zhodnoťtě nabídku nových osobních vozidel střední třídy dostupných v ČR v roce 2026. Automobil by neměl být EV nebo plug-in hybrid. Předpokládaný rozpočet je XXXX,- Kč.*

Výše uvedené zadání je sice relativně stručné, vede ale na poměrně komplexní rozhodovací problém, protože nám omezuje možné alternativy motorovými specifikacemi a také cenou. Jelikož je významná cena, lze předpokládat, že kromě pořizovací ceny bude potřeba zohlednit také ceny za provoz a údržbu vozidla. Kromě toho, se automobily budou výrazně lišit svými jízdními vlastnostmi, ale také dalšími vlastnostmi, které ovlivní celkovou užitnost.

Budeme tedy potřebovat hodnotit řadu alternativ s velkým množstvím kritérií, které je potřeba hodnotit společně, aby mohlo být dosaženo optimální rozhodnutí. Problém tedy jednoznačně vede na metody multikriteriálního rozhodování.

Jedinou překážkou by mohla být celková komplexita rozhodnutí. Pro jednoduché rozhodovací situace obvykle tyto metody nepoužíváme. Jejich nasazení totiž není úplně jednoduché a tak se vyplatí až u větších (složitějších) rozhodovacích problémů, kde se vyplatí investovat čas a prostředky do realizace složitější analýzy.

Při shromažďování informací o alternativách lze začít také s identifikací kritérií, která použijeme pro celkové hodnocení. Některá kritéria mohou vyplývat přímo ze zadání, jako je např. pořizovací cena. Jiné ale mohou být identifikovány až rozborem identifikovaných alternativ a jejich charakteristických vlastností. Podmínkou pro kritéria je, že všechny alternativy musí být hodnoceny ve všech kritériích.

Sada hodnotících kritérií tedy musí být stejná pro všechny alternativy.

Kritéria jsou obvykle vyjádřena ve vlastních (naturálních) jednotkách. Z pohledu porovnání je to problém, protože takto vyjádřená kritéria nejsou přímo porovnatelná. Pro účely porovnání je proto potřebujeme obvykle převést do jednotné stupnice (škály) procesem, který nazýváme *normalizace*.

Ne všechny metody multikriteriálního rozhodování ale takový převod vyžadují, některé mají normalizaci zabudovanou přímo ve svém algoritmu, jiné tento problém obcházejí. Velká část metod, ale vyžaduje, aby hodnota výkonu alternativ v kritériích byla normalizovaná a tuto normalizovanou hodnotu bere jako svůj vstup.

Kritéria také obvykle nepřispívají k optimalitě rozhodnutí stejným dílem - některá jsou tedy významnější než jiná. Relativní význam je pro účely aplikace metody potřeba vyčíslit. To realizujeme tak, že jednotlivým kritériím přiřadíme *váhy*.

Podobně jako v případě normalizace existují metody, které váhy nepotřebují, resp. jejich výpočet je zabudován přímo do algoritmu metody, většina metod však vyžaduje váhové koeficienty jako svůj vstup.

Existuje celá řada normalizačních metod a také metod pro odvozování váhových koeficientů. S některými z nich se seznámíme v následující kapitole.

Pracujeme tedy vlastně o trojicí algoritmů:

- normalizace
- odvození váhových koeficientů
- algoritmus multikriteriálního rozhodování

Celý proces tedy do jisté míry funguje jako skládačka, v rámci které analytik volí kombinaci algoritmů, která nejlépe odpovídá řešenému problému.

Tato volba je poměrně důležitá, protože volba bude mít zásadní vliv na výsledek výpočtu.

6.2 Metoda vážených součtů

Jedná se o základní metodu, se kterou se obvykle začíná. Metoda předpokládá, že výkon alternativ v kritériích bude vyjádřen numericky a je normalizován a že je znám význam jednotlivých kritérií (jsou zadané nebo jinak odvozené váhy). Předpokládáme také, že jednotlivá kritéria jsou vzájemně nezávislá a také, že všechna kritéria se významně podílejí na vysvětlení rozhodovacího problému.

Pro zjednodušení interpretace výsledků často normalizujeme váhy tak (viz lineární normalizace), aby jejich součet byl roven jedné a normalizované hodnoty výkonu alternativ v kritériích násobíme 100x.

Výše uvedené požadavky zajišťuje analytik, jelikož je algoritmus není schopen vynutit.

Označme matici s hodnotami výkonu alternativ v kritériích PM (jako performance matrix, tedy matice výkonu). Aplikací zvolené metody normalizace získáme normalizovanou matici výkonu PM_{norm}. Na tuto matici můžeme aplikovat váhy a výsledek sečíst pro jednotlivé alternativy, viz (6.1). Pokud jsme normalizovali váhy a výkon vynásobili 100x tak, jak bylo doporučeno výše, pak nám vyjde výsledek v intervalu 0 - 100, který lze interpretovat ve smyslu blízkosti alternativy k ideálnímu řešení (hodnoty 100).

Výsledný indikátor (PI - performance indicator) lze použít k identifikaci nejlepšího řešení (nejvyšší hodnota) nebo pro seřazení alternativ podle celkového výkonu, podle povahy řešeného problému.

$$PI = \sum_{i=1}^{n} w_i \cdot PM_{norm}[i,] \tag{6.1}$$

Implementace tohoto přístupu v tabulkovém procesoru je triviální a na základě výše uvedeného postupu by ji měl být schopen zrealizovat kdokoliv s minimálními znalostmi MS Excel. Nás však zajímá metoda WSM, spíše jako prostředek pro seznámení se s tímto typem metod. Výchozím bodem, chceteli, ze kterého se chceme odrazit případně k použití sofistikovanějších nástrojů. Tyto nám umožňují využít jiné metody, které možná lépe vyhovují specifikům řešeného problému.

K tomuto účelu využijeme R se specializovanou knihovnou MCDASupport [44]. Ta ve verzi 0.36 podporuje desítky různých metod multikriteriálního rozhodování včetně metody WSM.

Knihovna v současnosti není k dispozici na CRAN repozitářích, budete si ji proto muset nainstalovat separátně pomocí následujícího instalačního skriptu:

Výpis 6.1: Instalace MCDASupport knihovny z GitHub

```
packages <- c("mathjaxr", "graphics", "igraph", "diagram", "stats", "dplyr", "visNetwork", "plotly", "
    tidyr", "lpsolve")
install.packages(setdiff(packages, rownames(installed.packages())))
if(length(setdiff(c("MCDASupport"), rownames(installed.packages()))) == 1){
  url <- 'https://github.com/psenovsky/MCDASupport/releases/download/v0.36/MCDASupport_0.36.tar.gz'
  destfile <- paste(getwd(), '/MCDASupport_0.36.tar.gz')
  download.file(url, destfile)
  install.packages("MCDASupport_0.36.tar.gz", repos=NULL, type="source")
}
```

Knihovna ke svému provozu potřebuje mít instalovány některé další knihovny. Skript nejprve prověří Vaši instalaci R a následně doinstaluje chybějící závislosti. Tyto závislosti se instalují z oficiálního repozitáře R. Následně skript stáhne z GitHub verzi 0.36 (poslední dostupnou verzi v době psaní skript) a nainstaluje ji z GitHub repozitáře autora.

Zformulujme rozhodovací problém. Řekněme, že chceme rozhodovat mezi devíti alternativami označovanými jako A1 - A9, které charakterizuje šest kritérií K1 - K6. Tento rozhodovací problém je adaptován z pracovní verze diplomové práce Prokšová, M.: Využitelnost metod multikriteriálního rozhodování v systému kritické infrastruktury. V době psaní těchto skript práce ještě nebyla publikována, ale Vy si její znění již budete moci přečíst např. přes repozitář https://dspace.vsb.cz.

Tabulka 6.1: Rozhodovací matice - příklad KI

	K1	**K2**	**K3**	**K4**	**K5**	**K6**
optimalizace	max	max	min	max	min	max
váha	0.25	0.20	0.15	0.15	0.10	0.15
A1	120	45	24	3	2	9
A2	100	35	12	2	3	8
A3	90	30	10	2	3	7
A4	80	22	8	2	3	6
A5	110	40	36	3	2	9
A6	120	50	20	3	2	10
A7	120	48	30	3	2	9
A8	60	55	48	5	3	8
A9	70	28	16	3	3	7

Pro naše účely je v podstatě jedno, co čísla znamenají (pokud chcete se dozvědět podrobnosti o rozhodovacím problému budete muset dohledat výše uvedenou diplomovou práci). Podstatné je, že

jsme shromáždili všechny potřebné informace pro porovnání. Váhy jsou již normalizovány a nemusíme je dále měnit. Matici PM, ale budeme muset přepočítat. Pro naše účely použijeme min max normalizaci.

Podstatu této a dalších normalizací probereme v následující kapitole, zde se proto omezíme na to, že uvedená metoda normalizace nám převede stupnice do intervalu 0 - 1, kde 0 bude představovat nejhorší výkon a 1 ten nejlepší. Pro lepší interpretovatelnost výsledku vynásobíme výsledek 100x a aplikujeme WSM metodu. Řešení je dostupné v následujícím výpisu.

Výpis 6.2: Výpočet rozhodovacího problému metodou WSM

```
library(MCDASupport)
PM <- matrix(c(
  120,45,24,3,2,9,
  100,35,12,2,3,8,
  90,30,10,2,3,7,
  80,22,8,2,3,6,
  110,40,36,3,2,9,
  120,50,20,3,2,10,
  120,48,30,3,2,9,
  60,55,48,5,3,8,
  70,28,16,3,3,7
), nrow = 9, byrow = TRUE)
rownames(PM) <- c("A1", "A2", "A3", "A4", "A5", "A6", "A7", "A8", "A9")
colnames(PM) <-c("K1", "K2", "K3", "K4", "K5", "K6")
weights <- c(0.25,0.20,0.15,0.15,0.10,0.15)
minmax <- c("max", "max", "min", "max", "min", "max")
PMnorm <- PM
for(i in 1:6) PMnorm[i,] <- mcda_norm(PM[i,], minmax = minmax[i])
PI <- wsm$new(PMnorm, w = weights)
PI$result_table
PI$scoreM
```

Výše uvedený skript vypočte výslednou tabulku (result_table, viz tab. 6.2), ale kromě toho může např. vykreslit stohovaný graf (scoreM, viz obr. 6.2) pro lepší interpretaci tohoto výsledku.

Tabulka 6.2: Rozhodovací matice - příklad KI (zaokr. na 2 des. místa)

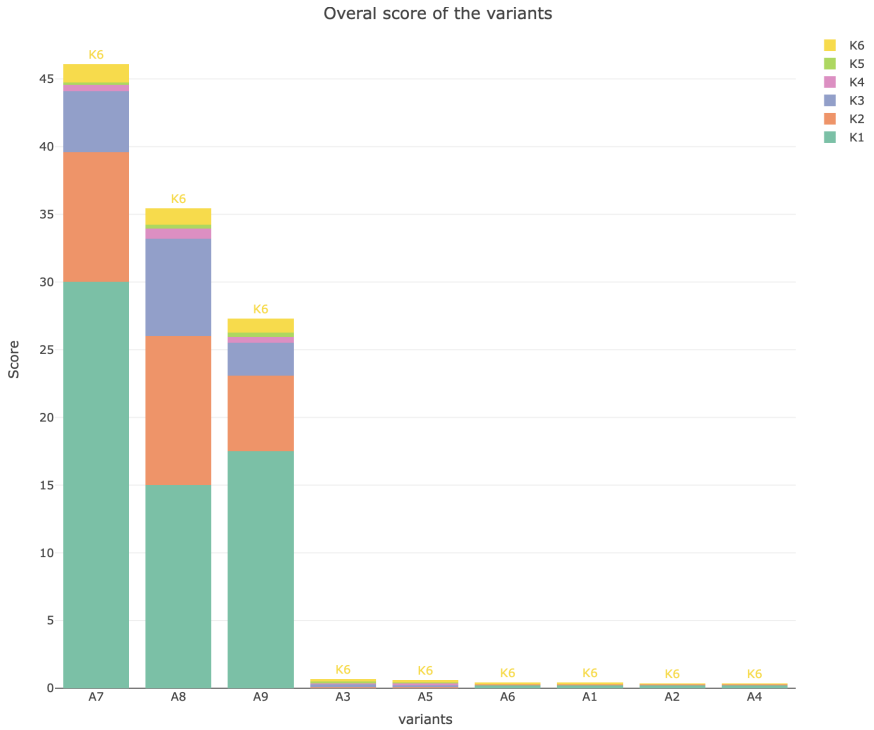
	K1	**K2**	**K3**	**K4**	**K5**	**K6**	$\sum$	**%**
A1	0.25	0.073	0.028	0.001	0	0.01	0.36	0.78
A2	0.25	0.07	0.02	0	0.001	0.01	0.34	0.74
A3	0	0.14	0.14	0.15	0.1	0.14	0.66	1.44
A4	0.25	0.05	0.01	0	0.001	0.01	0.32	0.7
A5	0	0.13	0.1	0.15	0.1	0.14	0.62	1.35
A6	0.25	0.08	0.02	0.001	0	0.01	0.37	0.79
A7	30.00	9.6	4.5	0.45	0.2	1.35	46.1	100
A8	15.00	11	7.2	0.75	0.3	1.2	35.45	76.9
A9	17.50	5.6	2.4	0.45	0.3	1.05	27.3	59.22

Z tab. 6.2 jednoznačně vyplývá, že na prvních třech místech jsou alternativy A7, A8 a A9 (v tomto pořadí). Ostatní alternativy jsou z pohledu optimality blízké nule.

6.3 Alternativní metody multikriteriálního rozhodování

Výhodou použití specializovaného výpočetního software a knihovny je, že můžeme jednoduše použít také jiné metody, které mohou poskytnout odlišný vhled do řešeného problému, protože každá metoda přistupuje k formulaci výsledků odlišným způsobem.

Použití odlišných metod může, ale nemusí vyžadovat úpravu vstupů. Pro demonstraci jsme zvolili metody PROMETHEE II, VIKOR a TOPSIS (podrobnější popis viz [45]), pro které s přijetím určitých zjednodušujících předpokladů nemusíme měnit definici rozhodovacího problému z předchozí podkapitoly.



Obrázek 6.2: Stohovaný sloupcový graf vizualizující výsledky metody WSM

Vyzkoušejte si příklad
Jedna věc je vidět vyřešený příklad, druhá je vyzkoušet si jej samostatně. Zkuste tedy příklad vypočítat. Můžete zkusit realizovat také vlastní implementaci výpočtu v Excelu a srovnat jej s výsledkem v R.

Pro výpočet použijeme opět R v kombinaci s knihovnou MCDASupport. Budeme používat stejné vstupy jako v předchozím případě, proto zadání již nebudeme opakovat a jako vstup použijeme hodnoty vah a normalizovanou matici výkonu alternativ v kritériích.

Výpis 6.3: Výpočet rozhodovacího problému s metodami PROMETHEE II, VIKOR a TOPSIS

```
PIPROMETHEE <- promethee2$new(PMnorm, w = weights)
PIVIKOR <- vikor$new(PMnorm, w = weights)
PITOPSIS <- topsis$new(PMnorm, w = weights)

result <- as.data.frame(matrix(rep(0, times = 36), nrow = 9, byrow = TRUE))
colnames(result) <- c("WSM", "PROMETHEE_II", "VIKOR", "TOPSIS")
rownames(result) <- rownames(PM)
WSM_aligned <- PI$weighted_sum_prc[rownames(result)]
result$WSM <- rank(-WSM_aligned)
result$PROMETHEE_II <- rank(-PIPROMETHEE$netFlow)
Q_aligned <- PIVIKOR$Q[rownames(result)]
result$VIKOR <- rank(Q_aligned)
TOPSIS_aligned <- PITOPSIS$closeness_ord[rownames(result)]
result$TOPSIS <- rank(-TOPSIS_aligned)
result
```

Výsledky ale vycházejí odlišně, viz tab. 6.3.

Jak vidno z tabulky, je pro náš rozhodovací problém řešení vcelku stabilní. Vlastně jediný rozdíl je v A7 a A8 pro hodnocení pomocí PROMETHEE II a ostatních metod.

Tabulka 6.3: Srovnání výsledků metod WSM, PROMETHEE II, VIKOR a TOPSIS

	WSM	**PROMETHEE II**	**VIKOR**	**TOPSIS**
A1	7	7	7	7
A2	8	8	8	8
A3	4	4	4	4
A4	9	9	9	9
A5	5	5	5	5
A6	6	6	6	6
A7	1	2	1	1
A8	2	1	2	2
A9	3	3	3	3

Pokusy
experimentujte s knihovnou zkuste zvolit jinou metodu a příklad vypočítat pomocí ní.

Zkuste také navrhnout a spočítat vlastní příklad.

Kapitola 7

Normalizační metody a metody odvozování váhových koeficientů

Náhled kapitoly
Už víme, že metody multikriteriálního rozhodování jsou jako skládačka. V předchozí kapitole jsme demonstrovali metodu WSM jako představitele těchto metod. Nyní se zaměříme na metody

- normalizace
- odvozování váhových koeficientů

7.1 Metody normalizace

normalizace Účelem normalizačních metod je převod čísel v různých stupnicích na jednu, obvykle v intervalu 0 - 1. Z předchozí kapitoly víme, že tento krok je vyžadován řadou metod multikriteriálního rozhodování, jako je např. metoda WSM.

Z předchozí kapitoly také víme, že kritéria, jejichž hodnoty jsme normalizovali byla z pohledu optimalizace maximalizována nebo minimalizována. I tuto informaci normalizace zohledňuje. Normalizaci lze tak použít také k tomu, aby se sjednotil z hlediska výpočtu směr optimalizace obvykle k maximalizaci.

V následující tabulce najdete populárnější metody normalizace, které se v praxi používají, viz tab. 7.1.

Z výše uvedených jsou nejdůležitější vektorová a min-max normalizace. Obě normalizace vedou na přepočtený interval 0 - 1, ale každá má jiný účel. Min-max používáme obvykle pro normalizaci kritérií. Aplikace metody přepočte nejhorší hodnotu kritéria na nulu a tu nejlepší na 1.

Vektorová normalizace proti tomu se často používá pro normalizaci váhových koeficientů. Tato metoda má pro tento účel jednu velmi příjemnou vlastnost, součet normalizovaných hodnot je roven 1.

Přesto i ostatní metody mají své použití. Důvodem bývají odlišné statistické vlastnosti výsledku normalizace. Min-max normalizace např. vycentruje rozdělení pravděpodobnosti a převede je na normální rozdělení. Tato změna ale nemusí být žádoucí. V takovém případě je potřeba použít jinou metodu normalizace.

Zkusme si spočítat výše uvedené metody pomocí knihovny MCDASupport, která je potřebuje. Budeme normalizovat vektor n od 1 do 10. Výsledky naleznete v tab. 7.2.

Výpis 7.1: Normalizace vektoru různými normalizačními metodami

```
1 library(MCDASupport)
2 n <- 1:10
3 df <- data.frame(
4   mcda_norm(n, minmax = "max", method = "LaiHwang"),
5   mcda_norm(n, minmax = "max", method = "linear aggregation"),
6   mcda_norm(n, minmax = "max", method = "logarithm"),
```

Tabulka 7.1: Populární metody normalizace

metoda	maximalizace	minimalizace
Lai-Hwang	$z = \frac{x}{max(x)-min(x)}$	$z = \frac{x}{min(x)-max(x)}$
lineární normalizace[1]	$z = \frac{x}{\sum_{i=1}^{m} x_i}$	$z = \frac{\frac{1}{x}}{\sum_{i=1}^{m} \frac{1}{x_i}}$
logaritmická	$z = \frac{ln(x)}{ln(\prod_{i=1}^{m} x_i)}$	$z = \frac{1-\frac{ln(x)}{ln(\prod_{i=1}^{m} x_i)}}{m-1}$
Markovic normalization	$z = 1 - \frac{x-min(x)}{max(x)}$	
min-max normalizace	$z = \frac{x-min(x)}{max(x)-min(x)}$	$z = \frac{max(x_j)-x_i}{max(x_j)-min(x_j)}$
nelineární normalizace	$z = (\frac{x}{max(x)})^2$	$z = (\frac{max(x)}{x})^2$
normalizace k průměru	$z = \frac{x}{\mu}$	
normalizace k nejlepší hodnotě[2]	$z = \frac{x}{max(x)}$	$z = \frac{min(x)}{x}$
normalizace Tzenga a Huanga	$z = \frac{max(x)}{x}$	
vektorová normalizace	$z = \frac{x}{\sqrt{\sum_{i=1}^{m} x_i^2}}$	$z = \frac{\frac{1}{x}}{\sqrt{\sum_{i=1}^{m} \frac{1}{x_i^2}}}$
normalizace Zavadskase a Turskise[3]	$z = 1 - \lvert\frac{max(x)-x}{max(x)}\rvert$	$z = 1 - \lvert\frac{min(x)-x}{min(x)}\rvert$

[1] také známa pod jménem sumarizační normalizace
[2] také známa pod názvem normalizace k maximu
[3] také známa pod jménem JKN - Jüttler's-Körth's Normalizace

```
  mcda_norm(n, minmax = "max", method = "Markovic"),
  mcda_norm(n, minmax = "max", method = "minmax"),
  mcda_norm(n, minmax = "max", method = "nonlinear"),
  mcda_norm(n, minmax = "max", method = "toaverage"),
  mcda_norm(n, minmax = "max", method = "tobest"),
  mcda_norm(n, minmax = "max", method = "max"),
  mcda_norm(n, minmax = "max", method = "TzengHuang"),
  mcda_norm(n, minmax = "max", method = "vector"),
  mcda_norm(n, minmax = "max", method = "ZavadskasTurskis")
)
colnames(df) <- c("LaiHwang", "linear aggregation", "logarithm", "Markovic", "minmax",
                  "nonlinear", "toaverage", "tobest", "max", "TzengHuang", "vector",
                  "ZavadskasTurskis")
df
```

WSM použití jiné normalizace
V příkladu, který jste počítali pro metodu WSM v předchozí kapitole použijte jinou metodu normalizace a porovnejte výsledné pořadí. Změnilo se?

Tabulka 7.2: Normalizace vektoru 1-10 různými normalizačními metodami

data	Lai-Hwang	lin.aggr.	log	Markovic	minmmax	nelin.
1	0.11	0.02	0e+00	1.0	0	0.01
2	0.22	0.04	0.91e-07	0.9	0.11	0.04
3	0.33	0.05	3.03e-07	0.8	0.22	0.09
4	0.44	0.07	3.82e-07	0.7	0.33	0.16
5	0.56	0.09	4.44e-07	0.6	0.44	0.25
6	0.67	0.11	4.94e-07	0.5	0.56	0.36
7	0.78	0.13	5.36e-07	0.4	0.67	0.49
8	0.89	0.15	5.73e-07	0.3	0.78	0.64
9	1	0.16	6.05e-07	0.2	0.89	0.81
10	1.11	0.19	6.35e-07	0.1	1	1

data	k prům.	k nejlepšímu	max	Tzeng-Huang	vektorová	Zavadskas-Turskis
1	18.18	0.1	0.1	10	0.05	0.1
2	36.36	0.2	0.2	5	0.10	0.2
3	54.55	0.3	0.3	3.33	0.15	0.3
4	72.73	0.4	0.4	2.5	0.20	0.4
5	90.91	0.5	0.5	2	0.25	0.5
6	109.09	0.6	0.6	1.666667 0.30	0.6	
7	127.27	0.7	0.7	1.43	0.36	0.7
8	145.45	0.8	0.8	1.25	0.41	0.8
9	163.64	0.9	0.9	1.11	0.46	0.9
10	181.82	1.0	1.0	1	0.51	1.0

7.2 Metody odvozování váhových koeficientů

Pro odvozování váhových koeficientů existuje celá řada metod. V následující kapitole se např. seznámíme s metodou AHP, která kromě odvozování váhových koeficientů umí porovnávat přímo jednotlivé alternativy mezi sebou. V této kapitole se ale podíváme spíše na tři typy metod, které váhy umožní odvodit samostatně. Bude se jednat o:

- objektivní metody
- odvozování na základě mapování preferencí - zde použijeme metodu Fullerova trojúhelníku
- další metody založené na bodování

7.2.1 Objektivní metody

Objektivními metodami rozumíme metody, které váhy odvozují z nějaké statistické vlastnosti přímo datasetu. Může to být entropie, variabilita a řada dalších. Objektivita je zde brána nikoliv jako skutečná objektivita, ale spíše jako opak subjektivity, kdy by váhy byly voleny na základě subjektivního hodnocení, např. ze strany expertů.

Pro naše experimenty budeme opět používat knihovnu RMCDASupport ve verzi 0.36 nebo novější, která má řadu metod tohoto implementovaných, viz tab. 7.3.

Tabulka 7.3: Objektivní metody odvozování váhových koeficientů

zkratka	**název**	**normalizace**
MW	Mean weighting method	A
SDW	Standard Deviation Weighting method	A
SVW	Statistical Variance Weighting method	A
EWM	Entropy Weight Method	N
CRITIC	Criterion Importance Through Intercriteria Correlation	N
GCW	Gini Coefficient Weighting	N
MEREC	Method based on Removal Effects of Criteria	N
CILOS	Criterion Impact LOSs	N
IDOCRIW	Integrated Determination of Objective CRIteria Weights	N
MPSI	M Preference Selection Index	N

V tabulce sloupec *normalizace* (A, jako ano) znamená, že analytik si může volit vlastní metodu normalizace. Implementace objektivních metod v RMCDASupport umožňuje použít jakýkoliv typ normalizace implementovaný v této knihovně, viz předchozí podkapitola. Pokud analytik takovou metodu nezvolí, použije se min-max normalizace.

Pokud ve sloupci je hodnota N (ne), pak metoda má zabudovanou normalizaci buďto přímo do algoritmu odvozování váhových koeficientů nebo pracuje na takových principech, že nepotřebuje normalizaci hodnot realizovat vůbec.

Nejjednodušší metodou je **Mean weighting (MW) metoda** [29]. Tato metoda předpokládá, že váhy všech kritérií jsou stejné, tedy že mezi kritérii z pohledu významu nejsou signifikantní rozdíly z hlediska významu. Váhy jsou proto aproximovány průměrem: $w_j = 1/n$, kde n je počet kritérií v rozhodovacím problému.

Standard Deviation Weighting (SDW) metoda vypočítává váhy na základě směrodatná odchylky hodnot kritéria. Pří výpočtu postupujeme tak, že znormalizujeme hodnoty zvolenou normalizační metodou a vypočteme směrodatnou odchylku **populace** pro jednotlivá kritéria. Všimněte si slova populace, to hraje důležitou roli, protože způsob výpočtu je odlišný. Obvykle totiž počítáme směrodatnou odchylku statistického vzorku.

$$\sigma_j = \sqrt{\frac{\sum_{i=1}^{m}(F_{ij} - \overline{F_j})^2}{m}} \tag{7.1}$$

Pro všechna j od 1 do m, kde $\overline{F_j}$ je aritmerický průměr normalizovaných hodnot. Z této metriky odvodíme váhy aplikací vektorové normalizace:

$$w_j = \frac{\sigma_j}{\sum_{k=1}^{m}\sigma_k} \tag{7.2}$$

Statistical Variance Weighting (SVW) methoda je velmi podobná jako SDW metoda, pouze pro aproximaci váhových koeficientů používáme rozptyl místo směrodatné odchylky. Výpočet provádíme opět na normalizovaných hodnotách kritérií a využíváme populační variantu výpočtu rozptylu:

$$\sigma_j^2 = \frac{\sum_{i=1}^{m}(F_{ij} - \overline{F_j})^2}{m} \tag{7.3}$$

Hodnoty vah pak odvodíme aplikací vektorové normalizace na vypočtěném σ^2.

Entropy Weight Method (EWM) [29] nám umožňuje odvodit váhové koeficienty na základě entropie (neurčitosti) přítomně v datech. Metoda data nejprve znormalizuje:

$$EM_{ij} = \frac{PM_{ij}}{max(PM_{ij})_j} \tag{7.4}$$

Následně vypočte pravděpodobnosti kritérií:

$$p_{ij} = \frac{EM_{ij}}{\sum_{i=1}^{n} EM_{ij}} \tag{7.5}$$

kde j ... kritéria, i ... alternativy a n je počet alternativ. Vypočteme entropii E.

$$E_j = -P\sum_{i=1}^{n} p_{ij} * ln(p_{ij}) \tag{7.6}$$

kde $P = 1/ln(n)$. Hodnota E vychází vždy v intervalu $<0; 1>$. Výpočet pokračuje výpočtem stupně divergence $div_j = |1-E_j|$, ze které je možno odvodit váhové koeficienty vektorovou normalizací vypočtené divergence.

Criterion Importance Through Intercriteria Correlation (CRITIC) [17] odvozuje váhové koeficienty na analýzy korelace mezi jednotlivými kritérii. Základem je výpočet korelační matice R, spočtené pomocí Pearsonova korelačního koeficientu. Tato informace pak umožňuje vypočítat konflikt kritéria vůči ostatním kritériím:

$$\sum_{k=1}^{m}(1 - r_{jk}) \tag{7.7}$$

Následně vypočteme kvalitu informace poskytované kritériem vynásobením hodnoty konfliktu směrodatnou odchylkou. Vypočtený indikátor C normalizovaný vektorovou normalizací pak použijeme jako váhy.

$$C_j = \sigma_j \sum_{k=1}^{m}(1 - r_{jk}) \tag{7.8}$$

Gini Coefficient Weighting (GCW) počítá váhy na základě Gini indexu. Nejprve vypočteme Gini koeficienty G_j na základě párového absolutního rozdílu diskrétních hodnot kritéria dle (7.9) a tuto hodnotu vektorově normalizujeme, abychom odvodili váhové koeficienty.

$$G_j = \frac{\sum_{i=1}^{m}\sum_{k=1}^{m}|f_{ij} - f_{kj}|}{2m^2\overline{f_j}} \tag{7.9}$$

Method based on Removal Effects of Criteria (MEREC) [26] počítá váhy na základě zohlednění dopadu odebrání kritéria na celkový výkon alternativ. Metoda nejprve normalizuje kritéria pro maximalizační využívá rovnici (7.10) a pro minimalizační (7.11).

$$n_{ij} = \frac{min_k x_{kj}}{x_{ij}} \tag{7.10}$$

$$n_{ij} = \frac{x_{ij}}{max_k x_{kj}} \tag{7.11}$$

Z normalizovaných hodnot vypočteme celkový výkon alternativ s_i:

$$S_i = ln(1 + (\frac{1}{m}\sum_{j}|ln(n_{ij})|) \tag{7.12}$$

V dalším kroku vypočteme výkon alternativ při odebrání jednotlivých kritérií:

$$S'_{ij} = ln(1 + (\frac{1}{m} \sum_{k,k \neq j} |ln(n_{ik})|)) \tag{7.13}$$

Celkový vliv odebrání kritéria E_j pak vypočteme podle rovnice (7.14). Tato metrika po aplikaci vektorové normalizace poslouží jako váhový koeficient.

$$E_j = \sum_j |S'_{ij} - S_i| \tag{7.14}$$

Criterion Impact LOSs (CILAS) odvozuje váhové koeficienty na základě míry ztráty. Metoda byla navržena v 80. letech minulého století, její praktická implementace byla ale realizována až autory Zavadskas a Podvezko [42] o 40 let později, jako podpůrný nástroj jejich metody IDOCRIW, kterou popíšeme za chvíli.

Nejprve převedeme všechna kritéria minimalizační na maximalizační (7.15) a následně celou takto upravenou matici výkonu alternativ v kritériích normalizujeme (7.16).

$$r_{ij} = \frac{min_i r_{ij}}{r_{ij}} \tag{7.15}$$

$$x_{ij} = \frac{r_{ij}}{\sum_{i=1}^{n} r_{ij}} \tag{7.16}$$

Následně zkonstruujeme matici A, $a_{ij} = x_{kj}$. Pro tyto účely je potřeba identifikovat maximální hodnotu x_j v jednotlivých sloupcích normalizované matice výkonu a řádky k, na kterých se toto maximum nachází.

Např. pokud v prvním sloupci by se maximum nacházelo na třetím řádku, pak $a_{11} = x_{31}$.

Zjednodušeně sestavujeme matici A kopírováním řádků z normalizované matice výkonu X odpovídající řádku maxima v daném sloupci. To také znamená, že pokud jsme takové maximum identifikovali ve více sloupcích, pak se daný řádek v matici A bude opakovat.

Následně vypočteme relativní ztrátu P:

$$p_{ij} = \frac{x_j - a_{ij}}{x_j} \tag{7.17}$$

Jelikož matice A má vždy identifikovaná maxima ve své diagonále, matice P na diagonále bude mít 0.

Konečně zkonstruujeme matici systému vah F tak, že od matice P odečteme sumu sloupců v diagonále. Diagonála matice F bude proto vždy záporná. Pro sumu ve sloupcích bude platit, že $F = 0$. F pak použijeme pro odvození váhových koeficientů.

Váhové koeficienty získáme vektorovou normalizací metriky q získané vyřešením soustavy rovnic $Fq^T = 0$. Pro váhy nás zajímá netriviální řešení této soustavy, tedy jiné řešení než $q = 0$. Pro tyto účely existuje řada metod, ale implementace v RMCDASupport využívá Singular Variable Decomposition (SVD).

Výše zmíněný **Integrated Determination of Objective CRIteria Weights (IDOCRIW)** vyvinuli Zavadskas a Podvezko [42] jako kombinaci metod CILOS a EWM:

$$w_j = \frac{q_j \cdot ew_j}{\sum_{i=1}^{n} q_j \cdot ew_j} \tag{7.18}$$

kde q jsou váhy odvozené metodou CILOS a ew pak váhy odvozené metodou EWM.

Konečně metoda **M Preference Selection Index (MPSI)** byla navržena Gligorić et al [19] v roce 2022. Výkonostní matice je nejprve normalizována metodou normalizace k nejlepší hodnotě (r_j). Z normalizovaných hodnot pro jednotlivá kritéria spočteme průměr v_J. Zájmovou metrikou pro odvození váhových koeficientů jsou variace preferencí p_j:

$$p_j = \sum_{i=1}^{m} (r_{ij} - v_j)^2 \tag{7.19}$$

Hodnotu vah odvodíme aplikací vektorové normalizace na p_j.

Z výše uvedeného přehledu jednoznačně vyplývá, že k výpočtu je možno přistoupit opravdu různým způsobem. Volba metody pak bude mít významný vliv na hodnotu odvozených váhových koeficientů. Jelikož se využívají vnitřní statistické vlastnosti dat charakterizujících výkon předpokládáme, že všechna data používaná k hodnocení jsou relevantní vůči řešenému problému a také, že jsou nezávislá.

Metody nemají žádný mechanismus, jak tyto dva požadavky algoritmicky zajistit. Je proto na analytikovi, aby data připravil tak, aby tato omezení respektovala.

Pro naši demonstraci použijeme stejný příklad jako v předchozí kapitole a postupně na něj aplikujeme výše uvedené metody pro odvození váhových koeficientů:

Výpis 7.2: Objektivní metody výpočtu váhových koefeicientů

```
library(MCDASupport)
metody <- c("MW", "SDW", "SVW", "EWM", "CRITIC", "GCW", "MEREC", "CILOS", "IDOCRIW", "MPSI")
kriteria <- c("K1", "K2", "K3", "K4", "K5", "K6")
minmax <- c("max", "max", "min", "max", "min", "max")
PM <- matrix(c(
  120,45,24,3,2,9,
  100,35,12,2,3,8,
  90,30,10,2,3,7,
  80,22,8,2,3,6,
  110,40,36,3,2,9,
  120,50,20,3,2,10,
  120,48,30,3,2,9,
  60,55,48,5,3,8,
  70,28,16,3,3,7
), nrow = 9, byrow = TRUE)
rownames(PM) <- c("A1", "A2", "A3", "A4", "A5", "A6", "A7", "A8", "A9")
colnames(PM) <- kriteria
result <- as.data.frame(matrix(0, ncol = 10, nrow = 6))
colnames(result) <- metody
rownames(result) <- kriteria
for(i in 1:10) {
  result[,i] <- mcda_objective_weights(PM, method = metody[i], minmax = minmax)
}
result
```

Výsledné váhové koeficienty jsou dostupné v tab. 7.4.

Tabulka 7.4: Váhové koeficienty odvozené různými objektivními metodami (zaokrouhledno na 2 desetinná místa)

kritérium	MW	SDW	SVW	EWM	CRITIC	GCW	MEREC	CILOS	IDOCRIW	MPSI
K1	0.17	0.17	0.17	0.09	0.14	0.14	0.16	0.15	0.09	0.15
K2	0.17	0.15	0.14	0.13	0.13	0.17	0.2	0.25	0.22	0.17
K3	0.17	0.15	0.13	0.52	0.28	0.34	0.36	0.11	0.41	0.34
K4	0.17	0.14	0.11	0.15	0.17	0.16	0.12	0.15	0.18	0.14
K5	0.17	0.24	0.33	0.07	0.18	0.11	0.06	0.19	0.09	0.13
K6	0.17	0.14	0.12	0.04	0.1	0.09	0.1	0.15	0.04	0.07

Pokusy objektivní metody váhování
implementujte příklad na objektivní metody odvozování váhových koeficientů a prozkoumejte výsledky. Jak mo se liší.

Aplikujte vybranou metodu (nebo třeba všechny metody) na Vámi zvolený rozhodovací problém.

7.2.2 Určení váhových koeficientů na základě preferencí

Do této oblasti budou spadat dvě metody, které postupně budeme probírat a to konkrétně metoda *Fullerova trojúhelníku* a metoda *AHP*. AHP nás čeká až v následující kapitole, zde se proto zaměříme na Fullerův trojúhelník.

V obecné rovině metody odvozující váhy na základě preferencí se zaměřují na formální zachycení toho, jakým způsobem vnímají významnost jednotlivých kritérií experti v dané problémové doméně. Přístupů které k tomuto účelu můžeme použít je celá řada, Fullerův trojúhelník využívá k tomuto účelu párové porovnávání významnosti kritérií, konkrétně *binární párové porovnání*.

Tento typ porovnání vedle sebe postaví všechny možné kombinace kritérií a experta nechá vybrat to, které je ve dvojici významnější. Odtud tedy v názvu to *binární* - volíme jeden ze dvou. Myšlenka je taková, že kritéria, která budou preferována častěji budou významnější. Hodnota váhových koeficientů je pak odvozena z počtu preferencí daného kritéria (+ 1), které normalizujeme vektorovou normalizací (ano, opět u vah je to výrazně nejpoužívanější normalizační metoda).

+1 do preferencí je potřeba zadat pro případ, že některé kritérium by nebylo preferováno vůbec, což se může stát. V takovém případě by bez přičtení 1 vyšla váha 0, což by ale znamenalo, že dané kritérium nebude mít vliv na výsledek rozhodování. Po přičtení 1 bude výsledná váha pro takové kritérium velmi malá, ale nenulová.

Trouhelník v názvu je odvozován od vzhledu záznamu preferencí při ručním zpracování viz tab. 7.5 a 7.6.

Tabulka 7.5: Fullerův trojúhelník - preference kritérií

K1				
	K2			
K2		K1		
	K2		K4	
K3		K2		K1
	K4		K2	
K4		K3		
	K5			
K5				

Jak si předchozí pyramidu vysvětlit? Postupujeme po sloupcích. *První sloupec* obsahuje přehled všech kritérií. Ve *druhém sloupci* srovnávám K1 a K2, preferuji K2, K2 a K3, preferuji K2, K3 a K4, preferuji K4 a konečně K4 a K5, preferuji K5.

Ve *třetím sloupci* hodnotím ob kritérium, srovnávám tedy K1 a K3, K2 a K4, K3 a K5, v dalších sloupcích hodnotím ob 2 kritéria, 3 ... podle počtu kritérií. Logicky čím více kritérií, tím rozsáhlejší bude konstruovaná pyramida. Nyní vypočteme pořadí. Při výpočtu lze vynechat první sloupec pyramidy, který obsahuje všechna kritéria a tudíž nám nepomůže při zhodnocení celkové významnosti kritéria.

Alternativní zápis Fullerova trojúhelníku je v tab. 7.6. Výhodou tohoto zápisu je to, že v jedné tabulce jsou obsaženy jak jednotlivé preference, tak odvození váhových koeficientů, což může přispět k jednodušší interpretovatelnosti celkového řešení.

Preference kritérií jsou v tab. 7.6 zvýrazněny tučně.

Tabulka 7.6: Fullerův trojúhelník pro výpočet vah

Porovnání kritérií				**kritérium**	**výskyt**	**váha**
1	**1**	1	**1**	K1	2	0,2
2	3	**4**	5			
	2	**2**	**2**	K2	4	0,4
	3	4	5			
		3	**3**	K3	1	0,1
		4	5			
			4	K4	2	0,2
			5			
				K5	1	0,1

Určitě Vás napadne otázka, můžu výpočet realizovat pomocí knihovny RMCDASupport, zejména vzhledem k tomu, že je zde poměrně výrazně propaguji. Odpověď je ano. Knihovna má implementovanou funkci *binary_paiwise_comp()*. Její nevýhodou je, že jako vstup vyžaduje úplnou matici porovnání. V našem případě by použití vypadalo následovně:

Zachyťte vlastní preference
a použijte je pro návrh váhových koeficientů.

Výpis 7.3: Binární párové porovnání v RMCDASupport

```
1  library(MCDASupport)
2  pref <- rbind(
3    c(0,0,1,0,1),
4    c(1,0,1,1,1),
5    c(0,0,0,0,1),
6    c(1,0,1,0,0),
7    c(0,0,0,1,0)
8  )
9  rownames(pref) <- colnames(pref) <- c("K1", "K2", "K3", "K4", "K5")
10 t <- binary_paiwise_comp(pref)
```

Nechám na Vašem posouzení, jestli se pro výše uvedený postup vyplatí startovat R.

7.2.3 Ostatní metody odvozování váhových koeficientů

Existuje celá řada dalších přístupů, pomocí kterých lze odvodit váhové koeficienty. Třeba RMCDA-Support má implementovány také:

- BWM (Best-Worst Model)
- COPRAS
- DEMATEL
- PIPRECIA (Pivot Pairwise Relative Criteria Importance Assessment)
- PSI (je také metodou MCDA)
- Rank Ordering Methods (ROMs) jmenovitě: Rank Sum, Rank Exponent a Rank Reciprocal metody
- SWARA

Existují ale také metody, které nevyžadují využití výpočetní techniky a jsou z hlediska použití jednoduché. Zástupcem takových metody je *Rozpočtování vah*. Metoda funguje tak, že expertovi dáte k dispozici rozpočet bodů, které může dle libosti přiřadit jednotlivým kritériím, podle toho, jak vnímá jejich důležitost. Výsledné bodování pak normalizujeme vektorovou normalizací.

Předtím, než se vrhneme na metodu AHP (v další kapitole) zamysleme se nad případy, kdy informace používané k odvození váhových koeficientů získáváme od expertů (důraz na množném čísle). V takovém případě totiž je nutné podklady agregovat. K tomuto účelu lze použít celou řadu strategií - můžeme např. preference zprůměrovat, nebo použít medián, popř. modus pro kategoriální hodnocení. Toto jsou ty jednodušší způsoby.

Je potřeba si uvědomit, že volba konsolidačního mechanismu názorů expertů bude mít podstatný vliv na výsledné váhy a také, že nelze předem říci která z metod bude pro řešený problém fungovat nejlépe.

Trochu složitější přístup by představoval použití váženého průměru názorů expertů. Takovém případě předpokládáme, že různí experti mají z hlediska přínosu k určení váhových koeficientů různý vliv a tento vliv zohledníme volbou váhových koeficientů. K určení tohoto typu vah můžeme použít např. informace o úspěšnosti podobných aktivit (odhadů) daného experta v minulosti, pokud je tato informace dostupná. Vyjít lze také z hodnocení odbornosti experta.

Při použití skupinového mapování preferencí je potřeba poznamenat, že agregace nutně nevede k lepším výsledkům. Může se klidně stát, že shromáždíme od jednotlivých expertů vnitřně konzistentní názor na to, jak mají vypadat váhové koeficienty, ale agregací se tato konzistence vytratí a výsledek nebude odlišitelný od náhodně vygenerovaného výsledku. Problémem je, že v podstatě neexistuje jednoduchý způsob jak určit, že se tak stalo (resp. jak zhodnotit kvalitu výsledného agregovaného názoru).

Kapitola 8

Metoda AHP

Náhled kapitoly
V této kapitole se podíváme na jednu z nejpoužívanějších metod pro odvozování váhových koeficientů a podporou multikriteriálního rozhodování - *metodu Analytic Hierarchy Process (AHP)*.

8.1 Úvod od AHP

Nejprve se zamysleme nad tím, v čem je vlastně problém. Při výkladu metody WSM jsme formulovali požadavek na nezávislost jednotlivých kritérií, která používáme pro rozhodování. Tento požadavek není úplně jednoduše realizovatelný a tak z něj často slevujeme na úroveň požadavku na maximálně slabou závislost.

Problémem je, že pokud existuje mezi kritérii závislost, pak při použití vážených součtů výkonu alternativ v kritérií dojde k vychýlení celkového hodnocení směrem k charakteristikám pokrytým závislými kritérii.

Slabá závislost sice není ideální, ale umožňuje nám přijmout předpoklad, že takto závislá kritéria nevychýlí celkové hodnocení signifikantně a my budeme moci tak tuto odchylku zanedbat.

Metodu AHP vyvinul v 80. letech Thomas L. Saaty [37] právě pro řešení takových problémů. Metoda vyjadřuje rozhodovací problém systémem kritérií formujícím hierarchickou strukturu, což zajišťuje že poskytované výsledky nejsou tak náchylné k odchylkám v hodnocení. AHP také poskytuje nástroje umožňující základní zhodnocení kvality zaznamenaných preferencí s použitím tzv. *indexu konzistence*. Tento index umožňuje zhodnotit do určité míry, zda zaznamenané preference nejsou náhodou výsledkem náhodného procesu.

Vzhledem k tomu, že metoda AHP je metodou velmi populární, existují stovky, možná tisíce studií z různých oblastí, které tuto metodu používají pro řešení různých problémů. Použití metody je tak dobře popsáno a zdokumentováno včetně informací o problémech a omezeních této metody. Popularita metody také způsobila, že existuje celá řada softwarových prostředků, které jsou dostupné pro podporu použití metody v praxi.

Příkladem takového software jsou SuperDecisions [13] na jejichž vývoji se podílel přímo Saaty.

V literatuře je metoda AHP někd označována jako Saatyho metoda, je ale potřeba upozornit, že se jedná o nepřesné označení, jelikož Saaty ve skutečnosti vyvinul metod několik, mimo jiné také metodu Analytic Network Process (ANP).

ANP proti AHP nechápe rozhodovací problém jako hierarchii kritérií, ale jako síť kritérií. ANP je tak obecnější metodou schopnou pokrýt širší potrfolio problémů. Šlo by také říci, že AHP je speciálním případem ANP metody, kdy spojení mezi kritérii vytváří hierarchii.

Z prostorových důvodů se budeme v tomto textu věnovat pouze metodě AHP. Software jako jsou třeba výše uvedené SuperDecisions ale podporují obě tyto metody.

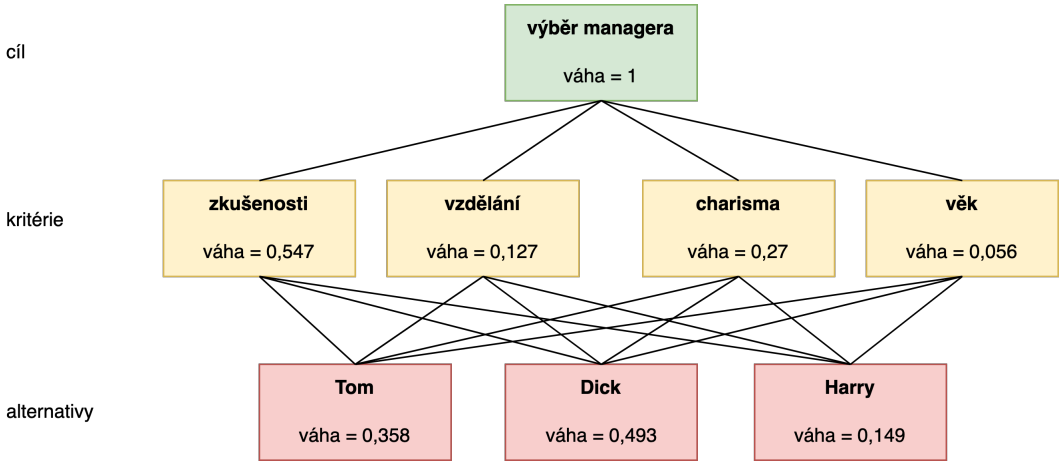
8.2 Proces AHP

Aplikace metody AHP se děje ve třech krocích:

1. model rozhodovacího problému jako hierarchie - rozložení kritérií pro rozhodování do podoby stromu
2. párové porovnání jednotlivých kritérií z pohledu jejich předpokládaného příspěvku k řešení problému
3. odvození váhových koeficientů.

Výsledek lze následně použít buď pouze pro zjištění váhových koeficientů nebo lze provést také párové srovnání výkonu alternativ v jednotlivých kritériích a použít tuto informaci k rankingu alternativ.

Zkusme se podívat na velmi jednoduchý příklad hierarchie, viz obr. 8.1. Rozhodujeme o výběru budoucího vrcholového managera na základě specifikace jeho vlastnosti.



Obrázek 8.1: Jednoduchá hierarchie problému výběru managera (adaptováno z [**?**])

Hierarchie je pro tento problém pouze dvouúrovňová. V první úrovni je specifikován cíl rozhodování, tedy výběr managera. Do tohoto cíle jsou konsolidována všechna kritéria. Tomu odpovídá celková váha 1 (100 %). Tuto váhu pak rozkládáme o úroveň níže do jednotlivých kritérií. Vždy přitom musí platit, že $\sum w$ jednotlivých kritérií je rovno velikosti váhy kritéria, které v hierarchii rozpracovávají.

Hierarchie na obr. 8.1 je tedy velmi jednoduchá, dokonce natolik jednoduchá, že použití AHP metody ve srovnání s WSM nepřináší signifikantní benefity. Z hlediska hloubky hierarchie nás ale ve skutečnosti nic neomezuje. Můžeme mít tak tři nebo čtyři (popř. více) úrovní hierarchie ... podle potřeby.

To už je situace, kdy použití AHP přináší signifikantní benefity.

Pokud by naším zájmem bylo pouze vypočítat váhové koeficienty jednotlivých prvků hierarchie, mohli bychom celou hierarchii omezit na cíle a kritéria a vynechat alternativy. Je tomu tak proto, že alternativy nám do hry vstupují pouze v případě, že AHP použít také pro skórování variant řešení. Hodnoty vah v tomto případě odpovídají celkovému výkonu alternativy napříč všemi „listovými“ kritérii v hierarchii.

I v tomto případě tedy realizujeme párové srovnání. Porovnáváme výkon páru alternativ v jednotlivých kritériích. Každý pár tedy postupně porovnáme ve všech kritériích, abychom dostali odhad celkového výsledku.

Tento přístup má výhodu v tom, že výkon alternativy v každém kritériu může zůstat ve svých naturálních jednotkách a to i v případě, že je kritérium specifikováno nenumerickou formou - třeba ordinální stupnicí nebo dokonce textovým popisem.

V případě, že jsou ale hodnoty kritérií normalizovány a použité škály odpovídají vnímání užitnosti, není krok párového porovnání alternativ potřeba realizovat. Formálně v takovém případě můžeme rozhodovací situaci „zploštit“ do jediné úrovně tvořené listovými uzly hierarchie a vypočítat celkové skóre alternativy metodou WSM.

Abychom zakončili vyhodnocení informací obsažených na obr. 8.1 z hodnot uzlů alternativ vyplývá, že nejlepším kandidátem na managerskou pozici je Dick následovaným Tomem. Harry se umístil na posledním místě.

Pro odvození váhových koeficientů potřebujeme provést párové porovnání v každé úrovni a každé skupině kritérií. Pro jednoduchou hierarchii na obr. 8.1 k tomuto účelu budeme potřebovat jednu matici porovnávající kritéria a čtyři matice porovnávající výkon alternativ v jednotlivých kritériích.

Můžeme to srovnat s řešením hierarchie z případové studie dále v této kapitole. Pro její řešení budeme potřebovat 3 matice pro porovnání jednotlivých kritérií a 8 matic porovnávajících výkon alternativ v jednotlivých kritériích, viz obr. 8.2.

V rámci každé matice porovnáváme jednotlivá kritéria pomocí následující stupnice:

- 1 - stejně důležité
- 3 - středně důležitější
- 5 - silně důležitější
- 7 - velmi silně důležitější
- 9 - extrémně důležitější
- Sudá čísla mohou posloužit pro podrobnější rozlišení mezi kritérii
- Možno použít i desetinná čísla pro ještě větší možnost rozlišení

pokud porovnávám kritéria a a b, přičemž je silně důležitější nežli b, pak $aSb = 5$. Z toho můžeme ale dovodit také že hodnota $bSa = 1/5$. Tento vztah nám tak umožňuje zmenšit počet preferencí, které je potřeba pro získání podkladů pro provedení analýzy realizovat.

Bude tedy platit, že pokud preferujeme a před b bude hodnota preference $\in < 1; 9 >$, v případě, že preferujeme b před a bude tato hodnoty v intervalu $\in < 0,11; 1 >$. (Pro zvídavé $1/9 = 0,11$:-).

Navíc vzhledem k tomu, že matice párového srovnání je vždy čtvercová, tedy n:n, kde n je počet kritérií, která porovnáváme, můžeme dovodit, že na diagonále budou vždy hodnoty 1. Je tomu tak proto, že na diagonále vlastně porovnáváme vždy kritérium se sebou samým.

Formálně se sice jedná o metodu párového porovnání, ale ve srovnání s metodou Fullerova trojúhelníku se nejedná o metodu *binární*, jelikož nemapujeme pouze hrubou preferenci ve smyslu a je lepší než b, ale také do určité míry intenzitu této preference.

Matice preferencí P (8.1) je základem pro odvození váhových koeficientů.

$$P = \begin{bmatrix} 1 & p_1 & \dots & p_{1n} \\ \frac{1}{p_1} & 1 & \dots & p_{\dots n} \\ \dots & \dots & 1 & \dots \\ \frac{1}{p_{1n}} & \dots & \dots & 1 \end{bmatrix} \tag{8.1}$$

Předpokládáme přitom, že preference odpovídají skutečným poměrům mezi váhami kritérií. To je poměrně silný požadavek, který v praxi může představovat jistý problém, pokud se nám jej nepodaří naplnit, pak výsledky, které metoda poskytuje mohou být výrazně nepřesné. To je také považováno za jednu ze slabin metody. Metoda také neposkytuje žádný nástroj, který by nám umožnil zhodnotit kvalitu mapovaných preferencí (jak moc odpovídají realitě), byť obsahuje nástroj, který nám umožní identifikovat situaci, kdy preference byly zadány očividně náhodně.

Problém odvození váhových koeficientu pak můžeme vyjádřit jako optimalizační problém, viz (8.2).

$$F = \sum_{i=1}^{k} \sum_{j=1}^{k} \left(p_{ij} - \frac{w_i}{w_j} \right)^2 \rightarrow \min \tag{8.2}$$

Kde k je počet hodnocených kritérií.

Řešení je možno realizovat pomocí metody *kvadratického programování*. Tento problém je ale výpočetně poměrně náročný a při ručním výpočtu velmi komplikovaný. Z toho důvodu velmi často nahrazujeme tento výpočet jeho aproximací pomocí geometrického průměru (8.3). Jo ale potřeba dodat, že tato aproximace zejména pro komplexnější problémy nemusí poskytovat dostatečně přesné výsledky.

$$w_i = \frac{\left(\prod_{j=1}^{k} p_{ij}\right)^{\frac{1}{k}}}{\sum_{i=1}^{k}\left(\prod_{j=1}^{k} p_{ij}\right)^{\frac{1}{k}}} \tag{8.3}$$

Hodnocení konzistence výpočtu můžeme provést pomocí *indexu konzistence* CI (8.4).

$$CI = \frac{w_{max} - k}{k - 1} \tag{8.4}$$

Vypočtené výsledky CI můžeme porovnat proti *náhodnému indexu konzistence* RCI pro výpočet *poměru konzistence* CR (8.5).

$$CR = \frac{CI}{RCI_k} \tag{8.5}$$

Hodnotu RCI je možno získat z tab. 8.1.

Tabulka 8.1: AHP - náhodný index konzistence (převzato z [37])

k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
RCI	0	0	0,58	0,9	1,12	1,24	1,32	1,41	1,45	1,49	1,51	1,48	1,56	1,57	1,59

Jako zlomová je považována hodnota $CR = 0,1$. Pokud $CR < 0,1$ předpokládáme, že zmapované preference, které jsme použili pro odvození váhových koeficientů nejsou náhodné.

Pokud bychom použili statistickou terminologii, pak provádíme testování nulové hypotézy, že zmapované preference jsou náhodné, kterou testujeme pomocí CR a pokud $CR < 0,1$, pak tuto hypotézu zamítáme.

K výše uvedenému je potřeba dodat, že hodnota $CR = 0,1$ je pouze orientační. Při hodnocení bychom měli usilovat o co nejnižší hodnotu. Pro komplexní problémy, ale poskytovaná kvalita určení preferencí jen velmi obtížně bude výrazně nižší než 0,1. Preference jsou vždy zatíženy osobním zaujetím hodnotitele (kognitivní bias), který bude snižovat kvalitu poskytnuté informace.

Další nepřesnosti mohou vznikat snahu mapovat preference z širší skupiny hodnotitelů, kdy preference jednotlivých hodnotitelů je potřeba agregovat do jediné hodnoty.

8.3 Případová studie - výběr auta Johnsonových

Tento příklad byl původně publikován na Wikipedii [4], v těchto skriptech je prezentována lehce upravená verze příkladu.

Cílem rozhodování je pomoci rodině Johnsonových vybrat nový automobil. Rodina po pečlivém zvážení sestavila rozhodovací hierarchii, viz obr. 8.2 a identifikovala 6 automobilů, ze kterých bude vybírat.

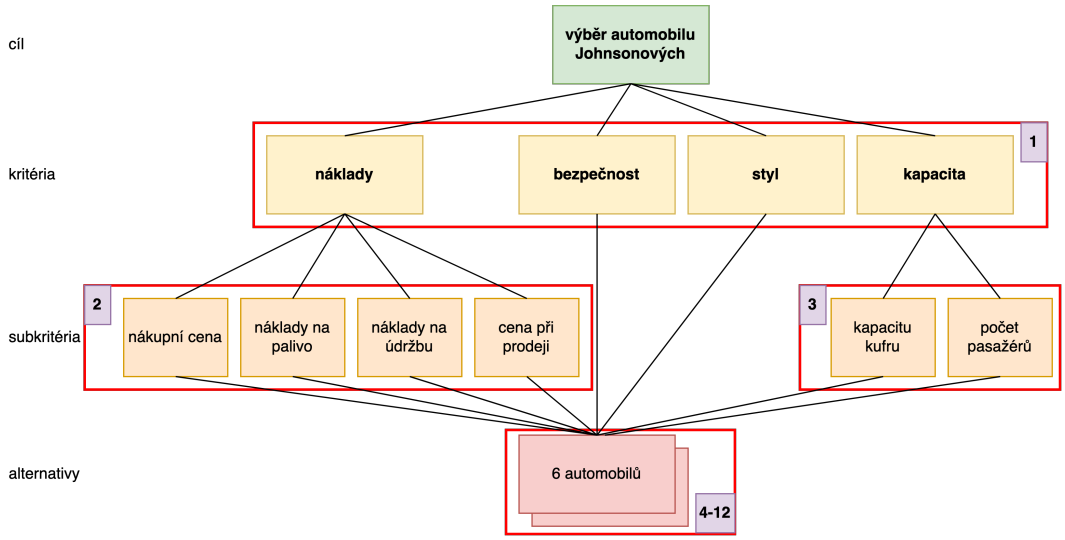
Johnsonovi vybírají z následujících aut:

1. Accord (sedan)
2. Accord (hybrid)
3. Pilot SUV
4. CR-V SUV
5. Element SUV
6. Odyssey minivan

Párové porovnání kritérií je dostupné v tab. 8.2.

Pro subkritéria nákladů je párové porovnání dostupné v tab. 8.3.

Poslední skupinu subkritérií tvoří kapacita, viz tab. 8.4.



Obrázek 8.2: Hierarchie výběru auta Johnsonových s určením skupin pro porovnání (adaptováno z [4])

Tabulka 8.2: Výběr auta - porovnání kritérií (adaptováno z [4])

	náklady	**bezpečnost**	**styl**	**kapacita**
náklady	1	3	7	3
bezpečnost	1/3	1	9	1
styl	1/7	1/9	1	1/7
kapacita	1/3	1	7	1

Tabulka 8.3: Výběr auta - porovnání subkritérií nákladů (adaptováno z [4])

	nákupní cena	**náklady na palivo**	**náklady na údržbu**	**cena při prodeji**
nákupní cena	1	3	3	5
náklady na palivo	1/2	1	2	2
náklady na údržbu	1/5	1/2	1	1/2
cena při prodeji	1/3	1/2	2	1

Tabulka 8.4: Výběr auta - porovnání subkritérií kapacita (adaptováno z [4])

	kapacita kufru	**počet pasažérů**
kapacita kufru	1	1/5
počet pasažérů	5	1

Celá rozhodovací hierarchie má 8 listových uzlů, Což znamená že musí být vytvořená 8 porovnávacích matic, kde se s těmito listovými kritérii vypořádáme. S prostorových důvodů podrobněji rozebereme pouze první z nich - tedy nákupní cenu, přičemž pro ostatní pouze specifikujeme koncovou porovnávací matici, která je nutná pro realizaci výpočtu. Případní zájemci se mohou podívat na způsob odvození do původního zdroje na Wikipedii [4][1].

Při úvahách o výhodnosti nákupu s ohledem na *nákupní cenu* používá rodina Johnsonových řadu pomocných úvah. Rodina má pro nákup k dispozici pouze 25 000 USD, je tak pro ni zásadní, jestli se automobil do této ceny vejde. Jako pomocný ukazatel pak může být použit rozdíl a podíl cen srovnávaných automobilů.

Základní vyhodnocení se dostupné v tab. 8.5, jemu přináležející odvození váhových koeficientů pak v tab. 8.6 a konečně matice, která je porovnává je dostupná v tab. 8.7.

[1] Výběr auta Johnsonových na Wikipedii: https://en.wikipedia.org/wiki/Analytic_hierarchy_process_%E2%80%93_car_example

Tabulka 8.5: Výběr auta - porovnání ceny (v USD) automobilů (adaptováno z [4])

	Porovnání		pořizovací cena		lepší	cena lepší		pod/(přes) rozpočet	
#	A	B	A	B	cena	rozdíl	poměr	A	B
1	Accord s.	Accord h.	20 360	31 090	A	10 730	1,53	4 640	**(6 090)**
2	Accord s.	Pilot	20 360	27 595	A	7 235	1,36	4 640	**(2 595)**
3	Accord s.	CR-V	20 360	20 700	A	340	1,02	4 640	4 300
4	Accord s.	Element	20 360	18 980	B	1 380	1,07	4 640	6 020
5	Accord s.	Odyssey	20 360	25 645	A	5 285	1,26	4 640	**(645)**
6	Accord h.	Pilot	31 090	27 595	B	3 495	1,13	**(6 090)**	**(2 595)**
7	Accord h.	CR-V	31 090	20 700	B	10 390	1,5	**(6 090)**	4 300
8	Accord h.	Element	31 090	18 980	B	12 110	1,64	**(6 090)**	6 020
9	Accord h.	Odyssey	31 090	25 645	B	5 445	1,21	**(6 090)**	**(645)**
10	Pilot	CR-V	27 595	20 700	B	6 895	1,33	**(2 595)**	4 300
11	Pilot	Element	27 595	18 980	B	8 615	1,45	**(2 595)**	6 020
12	Pilot	Odyssey	27 595	25 645	B	1 950	1,08	**(2 595)**	**(645)**
13	CR-V	Element	20 700	18 980	B	1 720	1,09	4 300	6 020
14	CR-V	Odyssey	20 700	25 645	A	4 945	1,24	4 300	**(645)**
15	Element	Odyssey	18 980	25 645	A	6 665	1,35	6 020	**(645)**

Tabulka 8.6: Výběr auta - porovnání ceny - odvození preferencí (adaptováno z [4])

	Porovnání		lepší	pref.	zdůvodnění
#	A	B	auto		
1	Accord s.	Accord h.	A	9	B překročilo rozpočet
2	Accord s.	Pilot	A	9	B překročilo rozpočet
3	Accord s.	CR-V	A	1	cena je takřka stejná
4	Accord s.	Element	B	2	cena B lepší o 1 000 USD
5	Accord s.	Odyssey	A	5	cena A lepší o 5 000 USD
6	Accord h.	Pilot	A	1	obě auta překračují rozpočet
7	Accord h.	CR-V	B	9	A překročilo rozpočet
8	Accord h.	Element	B	9	A překročilo rozpočet
9	Accord h.	Odyssey	B	7	A překročilo rozpočet, B překročilo těsně
10	Pilot	CR-V	B	9	A překročilo rozpočet
11	Pilot	Element	B	9	A překročilo rozpočet
12	Pilot	Odyssey	B	7	A překročilo rozpočet, B překročilo těsně
13	CR-V	Element	B	2	lepší cena B o 1 000 USD
14	CR-V	Odyssey	A	5	lepší cena A o 5 000 USD
15	Element	Odyssey	A	6	lepší cena A o 6 000 USD

Tabulka 8.7: Výběr auta - matice porovnání pořizovací cena (adaptováno z [4])

	Accord s.	Accord h.	Pilot	CR-V	Element	Odyssey
Accord s.	1	9	9	1	1/2	5
Accord h.	1/9	1	1	1/9	1/9	1/7
Pilot	1/9	1	1	1/9	1/9	1/9
CR-V	1	9	9	1	5	1/2
Element	2	9	9	1/5	1	6
Odyssey	1/5	7	9	2	1/6	1

Pro ostatní listová kritéria a subkritéria je párové porovnání dostupné v tab. 8.8 - 8.14.

Výpočet budeme demonstrovat v prostředí R pomocí knihovny AHP [**?**]. Tato knihovna ale již není v CRAN dostupná k automatizované instalaci přímo. Instalaci je ale možno provést ze stažených zdrojových kódů knihovny analogickým způsobem, jakým byla instalována knihovna MCDASupport v předchozí kapitole.

Stažení je možno provézt z `https://cran.r-project.org/src/contrib/Archive/ahp/`.

Tabulka 8.8: Výběr auta - matice porovnání pohonné hmoty (adaptováno z [4])

	Accord s.	**Accord h.**	**Pilot**	**CR-V**	**Element**	**Odyssey**
Accord s.	1	1/3	5	3	4	3
Accord h.	3	1	9	5	7	6
Pilot	1/5	1/9	1	1/4	1/3	1/4
CR-V	1/3	1/5	4	1	2	1
Element	1/4	1/7	3	1/2	1	1
Odyssey	1/3	1/6	4	1	1	1

Tabulka 8.9: Výběr auta - matice porovnání nákladů na údržbu (adaptováno z [4])

	Accord s.	**Accord h.**	**Pilot**	**CR-V**	**Element**	**Odyssey**
Accord s.	1	2	4	4	4	5
Accord h.	1/2	1	4	4	4	5
Pilot	1/4	1/4	1	1	2	1
CR-V	1/4	1/4	1	1	1	3
Element	1/4	1/4	1/2	1	1	2
Odyssey	1/5	1/5	1	1/3	1/2	1

Tabulka 8.10: Výběr auta - matice porovnání prodejní cena (adaptováno z [4])

	Accord s.	**Accord h.**	**Pilot**	**CR-V**	**Element**	**Odyssey**
Accord s.	1	3	4	1/2	2	2
Accord h.	1/3	1	2	1/5	1	1
Pilot	1/4	1/2	1	1	1/6	1/2
CR-V	2	5	1	1	4	4
Element	1/2	1	6	1/4	1	1
Odyssey	1/2	1	2	1/4	1	1

Tabulka 8.11: Výběr auta - matice porovnání bezpečnost (adaptováno z [4])

	Accord s.	**Accord h.**	**Pilot**	**CR-V**	**Element**	**Odyssey**
Accord s.	1	1	5	7	9	1/3
Accord h.	1	1	5	7	9	1/3
Pilot	1/5	1/5	1	2	9	1/8
CR-V	1/7	1/7	1/2	1	2	1/8
Element	1/9	1/9	1/9	1/2	1	1/9
Odyssey	3	3	8	8	9	1

Tabulka 8.12: Výběr auta - matice porovnání styl (adaptováno z [4])

	Accord s.	**Accord h.**	**Pilot**	**CR-V**	**Element**	**Odyssey**
Accord s.	1	1	7	5	9	6
Accord h.	1	1	7	5	9	6
Pilot	1/7	1/7	1	1/6	3	1/3
CR-V	1/5	1/5	6	1	7	5
Element	1/9	1/9	1/3	1/7	1	1/5
Odyssey	1/6	1/6	3	1/5	5	1

Knihovna sice již nadále není vyvíjena, ale pro základní experimentování s metodou AHP plně postačuje. Knihovna podporuje pouze „plný režim“ výpočtu, tedy včetně hodnocení alternativ a není ji tak možno snadno použít pouze pro odvozování váhových koeficientů.

Použití by mohlo vypadat následovně:

Výpis 8.1: Použití metody AHP pro řešení nákupu vozu Johnsonových

```
1 library(ahp)
```

Tabulka 8.13: Výběr auta - matice porovnání kapacita kufru (adaptováno z [4])

	Accord s.	**Accord h.**	**Pilot**	**CR-V**	**Element**	**Odyssey**
Accord s.	1	1	1/2	1/2	1/2	1/3
Accord h.	1	1	1/2	1/2	1/2	1/3
Pilot	2	2	1	1	1	1/2
CR-V	2	2	1	1	1	1/2
Element	2	2	1	1	1	1/2
Odyssey	3	3	2	2	2	1

Tabulka 8.14: Výběr auta - matice porovnání počet pasažérů (adaptováno z [4])

	Accord s.	**Accord h.**	**Pilot**	**CR-V**	**Element**	**Odyssey**
Accord s.	1	1	1/2	1	3	1/2
Accord h.	1	1	1/2	1	3	1/2
Pilot	2	2	1	2	6	1
CR-V	1	1	1/2	1	3	1/2
Element	1/3	1/3	1/6	1/3	1	1/6
Odyssey	2	2	1	2	6	1

```
2  library(data.tree)
3  cars <- Load("c:/path/cars.ahp")
4  Calculate(cars)
5  print(cars, filterFun = isNotLeaf)
6  Analyze(cars)
7  AnalyzeTable(cars)
```

Postup je poměrně jednoduchý. načteme datový soubor cars.ahp, který obsahuje rozhodovací hierarchii, včetně preferenčních matic a tyto podklady proženeme funkcí *Calculate*. Právě tato funkce provede výpočet váhových koeficientů a provede také skórování jednotlivých variant řešení.

Zbývající řádky kódu slouží pouze pro vizualizaci výsledků.

Příkaz *print* provede vypsání rozhodovací hierarchie na obrazovku. Pro soubor cars.ahp[2] vypadá hierarchie následovně:

```
     levelName
1  Root
2  ¦--Cost
3  ¦  ¦--Purchase Price
4  ¦  ¦--Fuel Costs
5  ¦  ¦--Maintenance Costs
6  ¦  °--Resale Value
7  ¦--Safety
8  ¦--Style
9  °--Capacity
10 ¦--Cargo Capacity
11 °--Passenger Capacity
```

V grafické podobě pak výsledek vypadá následovně (viz obr. 8.3):

První sloupec váhy zobrazuje odvozené váhové koeficienty. Jako nejméně významné byla identifikována kritéria kapacita kufru a styl s 3,6 % a 4,1 %. Všimněte si také, že právě styl ve sloupci *inconsistency* obsahuje hodnotu 10,1 %, což znamená, že $CR = 0,101$ a není tedy možno vyloučit, že preference vedoucí k váze stylu mohou být náhodné. V tomto případě, je ale hodnota 0,1 překročena pouze nepatrně, takže takový výsledek mlžeme akceptovat i s přihlédnutím k tomu, že příklad je školní :-).

[2]soubor je dostupný ke stažení v rámci systému LMS (https://lms.vsb.cz v kurzu *Modelování rozhodovacích procesů*)

	Weight	Accord Sedan	Odyssey Minivan	Accord Hybrid	CR-V SUV	Element SUV	Pilot SUV	Inconsistency
Root	100.0%	22.0%	21.2%	18.3%	15.8%	13.3%	9.3%	7.5%
Cost	51.0%	12.8%	4.7%	9.1%	11.3%	11.2%	1.9%	1.5%
Purchase Price	24.9%	6.1%	2.4%	0.6%	6.1%	9.1%	0.6%	8.0%
Fuel Costs	12.8%	2.9%	1.1%	6.2%	1.3%	0.9%	0.4%	3.1%
Resale Value	8.2%	1.8%	0.9%	0.8%	3.4%	0.9%	0.5%	0.5%
Maintenance Costs	5.1%	1.9%	0.3%	1.5%	0.5%	0.4%	0.5%	4.0%
Safety	23.4%	5.1%	10.2%	5.1%	0.8%	0.5%	1.8%	8.0%
Capacity	21.5%	2.8%	6.0%	2.8%	3.1%	1.4%	5.5%	0.0%
Passenger Capacity	17.9%	2.4%	4.9%	2.4%	2.4%	0.8%	4.9%	0.0%
Cargo Capacity	3.6%	0.3%	1.1%	0.3%	0.6%	0.6%	0.6%	0.2%
Style	4.1%	1.5%	0.3%	1.5%	0.6%	0.1%	0.2%	10.1%

Obrázek 8.3: AHP řešení problému výběru auto pro rodinu Johnsonových

Z pohledu formulace doporučení pro rodinu Johnsonových je nejlépe vyšel Accord sedan následovaný Odyssey minivanem, naopak nejhůře pak Pilot SUV.

Tedy jak vidno, výpočet je relativně jednoduchý, dosud jsme si ale neukázali jeho jednu komponentu, která je ale zcela zásadní. Nejprve totiž musíme připravit podklady pro výpočet, tedy v našem případě soubor cars.ahp.

Tento soubor je ve formátu YAML Ain't Markup Language (YAML). Jeho tvorba je dosti nepříjemná. Níže je přiložen krátký úsek výsledného souboru pro demonstraci jeho struktury.

Výpis 8.2: Struktura YAML

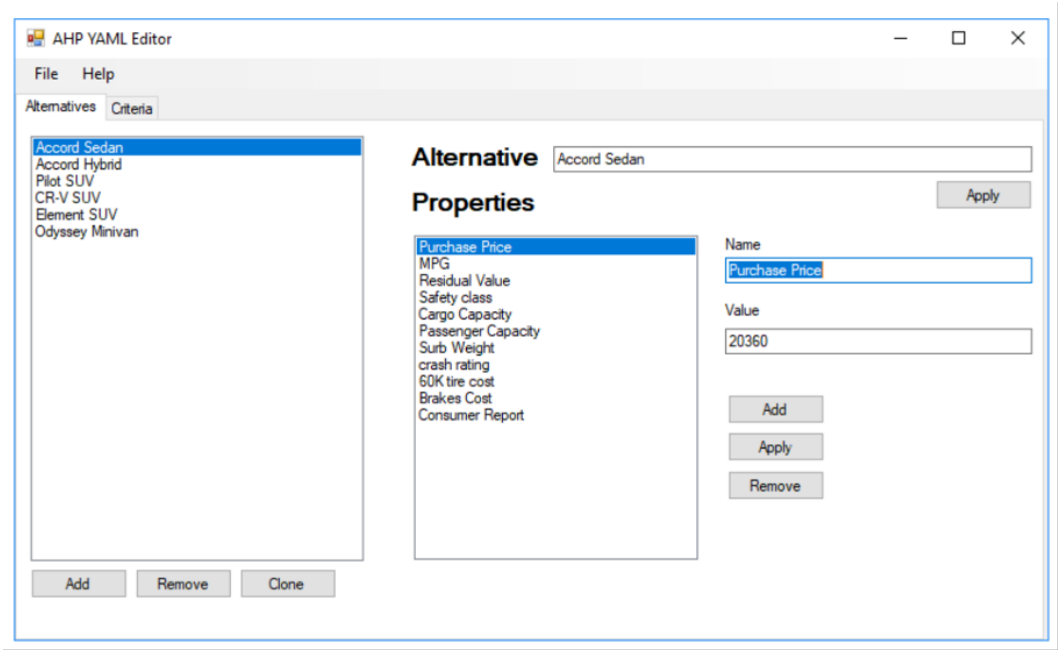
```
Version: 2.0

Alternatives: &alternatives
  Accord Sedan:
  Purchase Price: 20360
  MPG: 31
  Residual Value: 0,52
  Safety class: midsize car
  Cargo Capacity: 14
  Passenger Capacity: 5
  Surb Weight: 3289
  crash rating: 4* in side impact front
  60K tire cost: 700
  Brakes Cost: 1x
  Consumer Report: +++
...
Goal:
  preferences:
    pairwise:
      - [Cost, Safety, 3]
      - [Cost, Style, 7]
      - [Cost, Capacity, 3]
      - [Safety, Style, 9]
      - [Safety, Capacity, 1]
      - [Style, Capacity, 1/7]
  children:
    Cost:
      preferences:
        pairwise:
          - [Purchase Price, Fuel Costs, 2]
          - [Purchase Price, Maintenance Costs, 5]
          - [Purchase Price, Resale Value, 3]
          - [Fuel Costs, Maintenance Costs, 2]
          - [Fuel Costs, Resale Value, 2]
          - [Maintenance Costs, Resale Value, 1/2]
...
```

Struktura souboru je poměrně rigidní. Na prvním řádku je vždy určení verze. Následuje definice jednotlivých alternativ, podobně jako na řádcích 3 - 15 výpisu výše. Zbytek souboru pak tvoří definice hierarchie a párová srovnání. Všimněte si, že jednotlivé páry je potřeba porovnat pouze 1x. Dobře viditelné to je hned na úrovni kritérií, kdy na řádku 20 jsou porovnávána kritéria náklady a bezpečnost - preferovány náklady, ale nikterak silně. Opačné porovnání (bezpečnost a náklady) ale specifikováno není - knihovna je dopočítá sama.

To je výhodné, protože jinak by se počet řádků vstupního souboru takřka zdvojnásobil. I přes toto zjednodušení má zdrojový kód příkladu okolo 250 řádek textu. Plný text souboru je dostupný v příloze 3 těchto skript.

Ruční vytvoření takového souboru je komplikované. Pro zjednodušení jsem připravil jednoduché rozhraní umožňující takový soubor naklikat. AHP YAML Editor [43] je dostupný ke stažení buď ze systému LMS nebo stránek autora `https://fbiweb.vsb.cz/~sen76/data/uploads/programy/AHPEditor%20v0.1.7z` a poskytuje jednoduché rozhraní pro definici rozhodovací hierarchie a mapování preferencí.

Rozhraní poskytuje dva pohledy prezentované na obr. 8.4 a obr. 8.5.



Obrázek 8.4: Nastavení alternativ řešení rozhodovacího problému v AHP YAML editoru

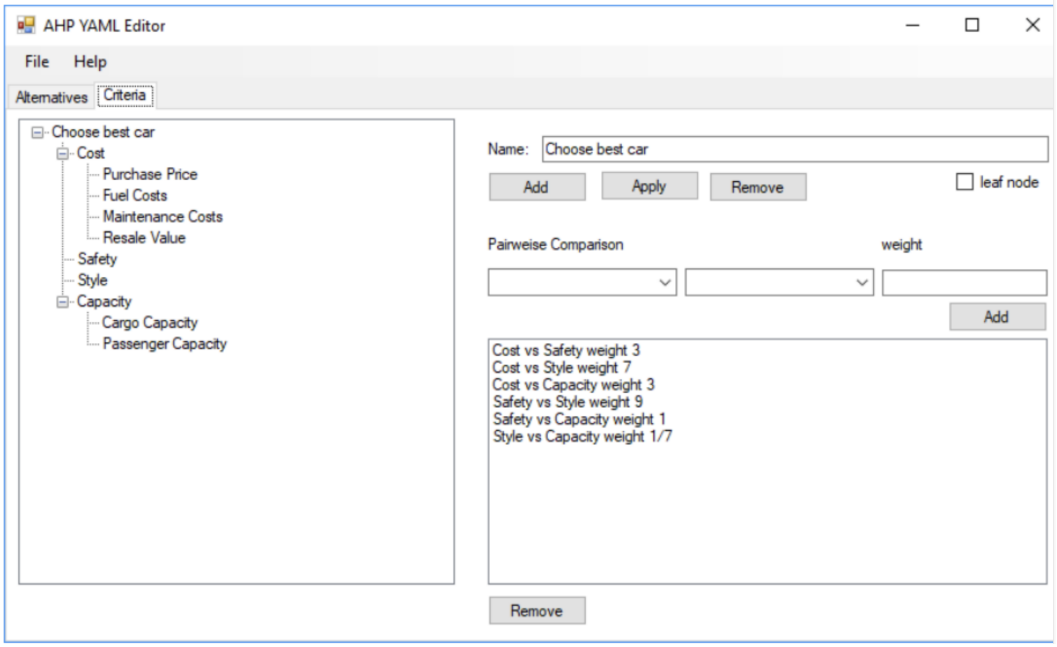
Nejprve z pohledu alternativ nastavíme všechny alternativy a jejich vlastnosti. Pro definici alternativ je potřeba, aby struktura vlastností zůstala napříč alternativami stejná. Z toho důvodu doporučujeme vytvořit první alternativu, nastavit její vlastnosti a všechny ostatní alternativy vytvořit „naklonováním“ této alternativy.

Operaci klování spustíte tak, že vyberete alternativu ke klonování a kliknete na tlačítko *Clone*. Výslednou strukturu alternativ už následně neměníme, pouze upravíme jméno alternativy a hodnoty jejích jednotlivých vlastností.

Z pohledu hierarchie kritérií postupujeme od kořene rozhodování směrem ke specifikaci jednotlivých listových uzlů.

Před započetím mapování preferencí doporučujeme nadefinovat úplnou hierarchii. Pro listové uzly nezapomeňte zaškrtnout volbu *leaf node*. Tato volba rozhodne, co se bude nabízet pro párové porovnání. Pokud volba není zaškrtnuta nabídnou se pro porovnání kritéria nižší úrovně. V případě, že je uzel hierarchie nastaven jako listový nabídnou se pro porovnání alternativy.

Po dokončení práce vyexportujte soubor do formátu YAML. Pro účely dalších úprav můžete soubor také uložit do JSON formátu. Tento formát je ale pouze vnitřním úložným formátem editoru a knihovna AHP v R s ním neumí pracovat.



Obrázek 8.5: Nastavení hierarchie kritérií rozhodovacího problému v AHP YAML editoru

8.4 Alternativní cesty k výpočtu

Výše uvedený postup je eufemisticky řešeno dosti nepříjemný a to i v případě, že se rozhodneme použít SW. Existuje nějaká cesta jak si celý proces zjednodušit - odpověď je ano v případě, že jediné, co nás zajímá jsou váhové koeficienty. Na LMS v kurzu máte k dispozici šablonu, do které můžete vyplnit pouze hodnoty a provede výpočet v celé hierarchii rozhodovacího problému. Berte ale v úvahu, že šablonu si budete muset upravit tak, aby skutečně odpovídala Vašemu rozhodovacímu problému.

Existují také kalkulátory AHP, které na základě preferencí odvodí váhové koeficienty a hodnotu CR pro kontrolu konzistence. Příkladem je třeba AHP Priority Calculator na https://bpmsg.com/ahp/ahp-calc.php. Jedná se o webovou aplikaci, ve které naklikáte preference, kliknete na tlačítko „Calculate“ a dostanete výsledek. Omezením ale je, že počítat můžete vždy pouze jednu matici.

Vytvoření hierarchie budete proto muset dělat ručně. V takovém případě začneme postupovat od vrcholu hierarchie postupně směrem dolů. Pro náš příklad výběru auta rodiny Johnsonových (viz obr. 8.2) bychom začali maticí 1 a odvodili váhové koeficienty.

Následovali bychom maticí 2 atd. Ale pozor kalkulátor navrhne pro další matice váhy bez ohledu na zapojení matice do hierarchie problému, tedy výsledné váhy budou vždy v intervalu 0 - 1 a jejich součet bude roven 1. Z logiky věci ale vyplývá, že váhy na nižších úrovních hierarchie musí v součtu dávat odvozenou váhu na vyšší úrovni.

Pro náš příklad výpočtu matice 2 to znamená, že součet váhových koeficientů této matice musí v součtu být roven váze kritéria náklady. To můžeme zajistit jednoduchou transformací vypočteného intervalu:

$$w2_j = w_j * w1 \tag{8.6}$$

Kde $w2_j$ ... jsou přepočtené váhy podřízené matice, w_j ... váhy vypočtené kalkulátorem a $w1$ je váha nadřízeného kritéria, které nižší hierarchií rozpracováváme.

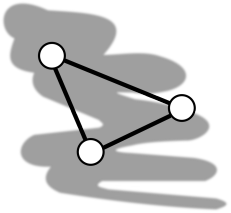
Vyzkoušejte si AHP
vyberte si šablonu MS Excel nebo kalkulátor priorit AHP a zkuste vyřešit některý z příkladů. Dbejte na to, aby konzistence váhových koeficientů nepřesáhla hranici CR = 0,1.

Kapitola 9

Projektové řízení a metoda CPM

Náhled kapitoly
Od problematiky rozhodování o alternativách zkusmě přejít k jiným metodám podpory rozhodování. Zaměřme se na situaci, kdy potřebujeme uřídit sady jednorázových aktivit pomocí kterých řešíme nějaký, obvykle složitý, problém. Tento proces označujeme obvykle *projektové řízení*.

V této kapitole probereme jednak software, který nám může pomoci, jednak teoretické základy metody CPM, která představuje základní nástroj jak získat a udržet kontrolu nad průběhem projektu.



Návaznosti
Na tuto kapitolu pak navážeme kapitolami následujícími, kde probereme další nástroje, které nám umožní lepší koordinaci a plánování úkolů, které jsme např. naprojektovali pomocí nástroje z této kapitoly.

9.1 Projektové řízení

Zkusme si nadefinovat několik pojmů, které se v dalším textu objeví. Prvním z nich je *management*. Slovo management je odvozeno z francouzského manéž, tedy kruhová aréna určená pro drezůru koní. Někdy se původ tohoto slova odvozuje také od francouzského managere – tedy řízení. Pro management se objevuje celá řada definicí, např. že management se dosahování cílů prostřednictvím jiných. Objevuje se také definice: Management je mobilizace všech zdrojů podniku za účelem dosažení vytyčených cílů.

Management jako vědní disciplínu řadíme do oblasti humanitní, proto je jeho úplné zvládnutí pomocí tvrdých, technických prostředků nemožné – ty jsou však ale použitelné (a vysoce užitečné) jako podpůrné prostředky. Jejich použití podpůrných metod automaticky nezajišťuje manažerovi úspěch, ale zvyšuje jeho šanci jej dosáhnout.

Dalším pojmem, který budeme často diskutovat je *projekt*. Pojmem projekt rozumíme sled činností, které vedou k vymezenému cíli, s jasně daným začátkem a koncem a přesně danými zdroji použitelnými pro splnění cílů projektu.

Projekt jako takový lze vymezit činnostmi, které v jeho průběhu musí proběhnout, v určité časové návaznosti. Takové návaznosti můžeme velmi dobře vyjádřit tabulkově. Demonstrujme si tento přístup na zjednodušeném příkladu stavby domu (viz tabulka 9.1).

Z praktického pohledu je projekt charakterizován zdrojovými omezeními. Z tab. 9.1 nám vyplynuly omezení časové, ale kromě nich máme celou řadu dalších zdrojů:

- finanční (rozpočet)
- lidské (dostupné hodinové úvazky osob, které mají projekt řešit)

Tabulka 9.1: Průběh činností projektu v čase

	činnost	**následníci**	**předchůdci**	**Délka [dny]**
A	Výkop základů	B	-	2
B	Betonování základů	C	A	10
C	Hrubá stavba	D	B	17
D	Zastřešení	E,F,G,H	C	5
E	Elektroinstalace	I	D	5
F	Potrubí (voda, plyn)	I	D	5
G	Topení	I	D	5
H	Okna	I	D	14
I	Omítky, podlahy	J	E,F,G,H	21
J	Kolaudace	-	I	1

- materiálové (suroviny, materiál, ...)
- další (např. čas na nějakých specializovaných strojích)

Strukturu projektu můžeme popsat prostřednictvím specifikace následníků činností nebo předchůdců činnosti. *Následníci* jsou činnosti, které musí následovat po dokončení právě hodnocené činnosti. *Předchůdci* jsou činnosti, které musí být dokončeny předtím než mohu započít právě hodnocenou činnost.

Jedná se vlastně o ekvivalentní zápisy, liší se pouze směr, kterým provádíme hodnocení. U následníků postupujeme od počáteční činnosti směrem k cíli projektu, u předchůdců začínáme u poslední činnosti a propracováváme se k začátku projektu.

Některé software pro projektové řízení např. Microsoft Project [1], open source Project Libre [3] nebo Open Workbench [2] umožňují volit si způsob navazování dle potřeb a dokonce jej měnit, s tím že se následníci a předchůdci operativně přepočítají.

Softwarová podpora

Vytváření diagramů ručně je poměrně náročné - zabere to spoustu času a udělat chybu je snadné. Jelikož projektové řízení je aktivitou, která je realizována často, vznikla v průběhu doby řada pokročilých softwarových nástrojů schopných usnadnit nám činnosti plánování projektů a také kontroly jejich realizace.

Defacto leaderem v oblasti projektového řízení je společnost Microsoft se svým nástrojem MS Project [1]. Jedná se o proprietální software, který sám Microsoft řadí do „širší“ rodiny MS Office. Tento nástroj se vyznačuje celkovou uživatelskou přívětivostí a možností integrace s dalšími technologiemi Microsoftu jako MS Exchange pro možnost propagace pracovních rozvrhů jednotlivým pracovníkům projektu přímo do jejich kalendářů zobrazovaných pomocí MS Outlook (nebo webového rozhraní).

Existuje také celá řada dalších softwarových nástrojů, které jsou přímo open source. Příkladem by mohl být třena Project Libre [3] (dříve známý pod jménem OpenProj). Project Libre je dostupný na prakticky všech platformách od Windows až po OS X. Jako alternativu lze zmínit produk Open Workbench [2], který je však dostupný pouze pro MS Windows.

Praktické ukázky, které máte možnost vidět v následující kapitole byly vytvořeny pomocí Project Libre.

Seznam aktivit v tabulární formě není úplně dobře použitelný pro operativní řízení průběhu jednotlivých aktivit projektu, plánování zdrojů apod. Z tohoto důvodu často používáme grafickou interpretaci informací o časovém charakteru aktivit a jejich vzájemné vazbě pomocí *harmonogramu* někdy také nazývaného *Ganttův diagram* a také pomocí *síťového diagramu*.

Jak vidno z obr. 9.1, červeně je zvýrazněna tzv. *kritická cesta*. Kritickou cestou rozumíme posloupnost cest od začátku do konce pro kterou platí, že překročení plánovaného času (aktivita trvá déle než bylo plánováno) automaticky vede k prodloužení délky projektu jako celku. Účelem identifikace kritické cesty proto je identifikovat aktivity, na které si manager musí z hlediska řízení dát obzvláště velký pozor.

Kritická cesta je:

- vždy přítomna
- vždy začíná v první a končí v poslední aktivitě
- vždy z časového pohledu nejdelší

To samozřejmě neznamená, že by manager mohl ostatní aktivity úplně ignorovat, pouze to, že u nich existuje jistá časová rezerva. Tuto rezervu lze využít při řízení - např. z nekritické činnosti uvolnit lidské nebo materiálové zdroje a posílit činnost kritickou tak, aby se zajistilo její dokončení v čas. Odebrání zdrojů z nekritické činnosti povede taktéž k posunu začátku nebo jejímu prodloužení, pokud je tento posun nebo prodloužení dostatečně malý, aby délka činnosti nepřesáhla limity stanovené existující časovou rezervou - nezasáhne tato změna negativně do celkového obrazu projektu (nedojde k jeho prodloužení).

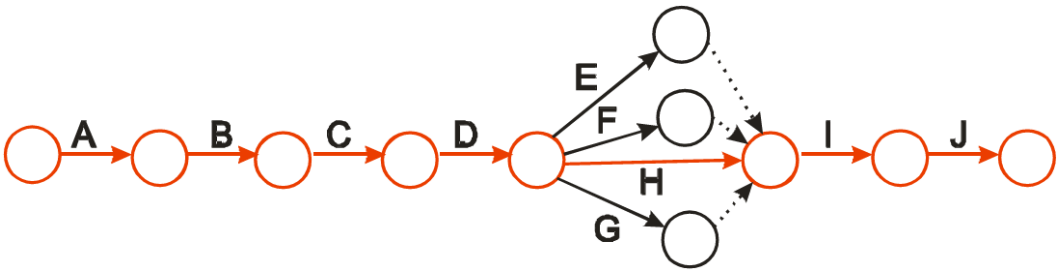
Hlavní rozdíl mezi aktivitou na kritické cestě a aktivitou nektritickou je přítomnost časové rezervy u nektritické aktivity. Pokud ale vlivem chybných rozhodnutí, externími vlivy apod. ztratíme kontrolu nad nekritickou činností natolik, že vyčerpá celou svou časovou rezervu, pak se z takové aktivity stane kritická. Pokud navíc tuto časovou rezervu překročíme, pak se navíc může stát, že aktivita původně kritická, přestane být kritickou. Kritická cesta se tedy v průběhu projektu může změnit a to je další důvod pro ta, abychom průběh plnění aktivit pečlivě sledovali.

Obrázek 9.1: Zjednodušený harmonogram projektu

V *síťovém grafu* jsou aktivity představovány hranami. Hrany E, F, G, H představují činnosti (aktivity), které mohou probíhat zároveň. Všechny tyto aktivity musí být dokončeny předtím, než bude zahájen činnost I. Abychom tyto vazby byli schopni zachytit v síťovém grafu přidáváme do něj virtuální aktivity (viz tečkované čáry na obr. 9.2).

Tyto virtuální činnosti neodpovídají žádným činnostem reálným a jejich délka z pohledu časového je rovna nule. Jejich úkolem je tedy pouze zpřehlednění výsledného grafu. K tomu je potřeba také dodat, že notace použitá na obr. 9.2 odpovídá notaci používané v původní způsobu zaznačení sítě používaném při ručním zpracování. Softwarové nástroje používají, jak později uvidíme, trochu jinou notaci.

I v síťovém grafu je možno vyznačit kritickou cestu, na obr. 9.2 je tato cesta opět vyznačena červenou barvou.



Obrázek 9.2: Síťový graf projektu

Možná Vás napadne otázka, proč k jednomu úkolu (identifikace kritické cesty) použít dva různé grafy. Různé grafy používáme, protože identifikace kritické cesty je pouze jedním z úkolů, které tyto grafy plní. Primárním úkolem síťového grafu je přehledně vizualizovat závislosti (návaznosti) mezi jednotlivými činnostmi. Primárním úkolem harmonogramu je vizualizovat činnosti přesně z hlediska časového, za účelem optimalizace využití zdrojů pro nekritické činnosti včetně jejich případného časového posunu.

Přínosy harmonogramů/síťových grafů projektů:

- identifikace kritické cesty
- optimalizace využití zdrojů pro vykonání nekritických činností (včetně posunutí termínů v mantinelech vytyčených kritickou cestou)
- kontrola realizovaných činností, podklady pro efektivní řízení

Příklad
Spočtěte délku kritické cesty pro příklad, se kterým jsme pracovali v této kapitole.

Řešení
Možné cesty
1 – A,B,C,D,E,I,J = 2 + 10 + 17 + 5 + 5 + 21 + 1 = 61
2 – A,B,C,D,F,I,J = 2 + 10 + 17 + 5 + 5 + 21 + 1 = 61
3 – A,B,C,D,H,I,J = 2 + 10 + 17 + 5 + 14 + 21 + 1 = 70
4 – A,B,C,D,G,I,J = 2 + 10 + 17 + 5 + 5 + 21 + 1 = 61

Kritická cesta je cesta 3 (A,B,C,D,H,I,J), projekt bude trvat 70 dní.

9.2 Project Libre - softwarová podpora projektového řízení

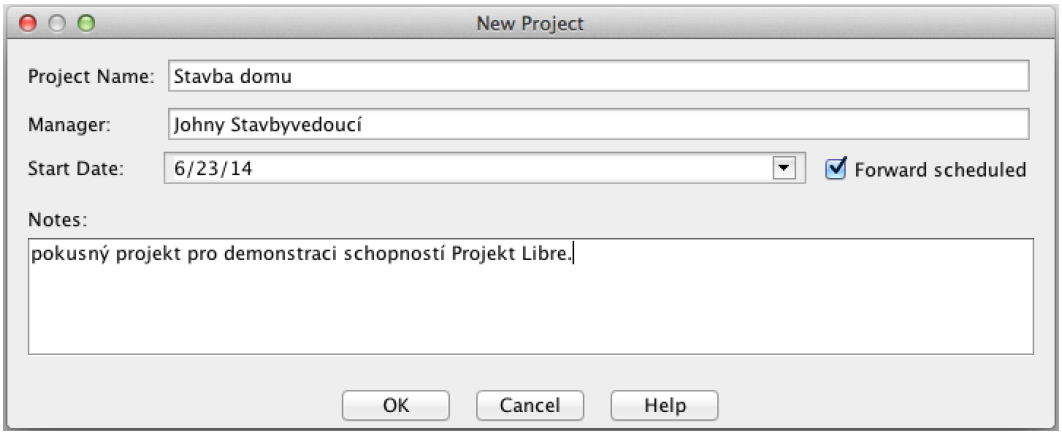
Průvodce studiem
V této kapitole budeme prakticky demonstrovat použití softwaru pro podporu řízení na softwarovém balíku Project Libre, většina z toho, co se zde naučíte je ale také přímo aplikovatelná v jiných softwarech pro podporu řízení projektů, ačkoliv se tyto balíky budou do určité míry zejména grafickým uživatelským rozhraní (Graphical User Interface (GUI)) lišit. Pokud tedy máte na počítači instalován jiný softwarový produkt určený pro projektové řízení, nemusíte jej nahrazovat Project Libre, tedy pokud nechcete.

Před použitím je nutné software stáhnout z jeho domácích stránek [3]. Začátečníkům bych doporučoval stáhnout si přímo binární distribuci Project Libre určenou pro jejich operační systém. Jelikož se ale jedná o open source, máte možnost také stáhnout zdrojové kódy a manipulovat i přímo s nimi.

Pro provoz balíku budete potřebovat mít instalován také Java Developlent Kit (JDK). Pokud je na svém počítači nemáte ještě nainstalováno, můžete si je stáhnout ze stránek společnosti Oracle, která JDK vyvíjí: https://www.oracle.com/java/technologies/downloads/.

Dvojklikem na staženém souboru zahájíte instalační proces. V čase psaní těchto textů (červen 2014) byla poslední dostupnou verzí Project Libre verze 1.5.9. Po dokončení instalace je možno program začít používat.

Po spuštění programu je uživatel vyzván, aby založil nový projekt nebo otevřel projekt existující. Jelikož právě s programem začínáme, nemáme k dispozici žádný existující projekt, budeme proto muset vytvořit projekt nový. Při založení nového projektu je nutné zadat jméno projektu, projektového managera, datum počátku projektu. Projekt můžeme doplnit poznámkami, ale to není povinné. Obrazovka s definicí nového projektu je na obr. 9.3.



Obrázek 9.3: Založení nového projektu v Project Libre

Různé softwarové balíky pro projekty podporují různé vlastnosti. Např. MS Project umožňuje specifikovat druh kalendáře, který se má v projektu využívat (např. bussiness hours - 8-mi hodinový pracovní den, 5 dní v týdnu, nebo třeba třísměnný provoz 7 dní v týdnu).

V rámci našeho experimentování s Project Libre budeme realizovat příklad z předchozí podkapitoly. Budeme tedy stavět rodinný dům. Kliknutím na OK vytvoříme projekt.

Nový projekt je automaticky otevřen v režimu Ganttova diagramu, který nám na jedné straně umožňuje definovat základní vlastnosti jednotlivých aktivit a na straně druhé je rovnou vizualizuje do podoby harmonogramu. Mezi základní vlastnosti řadíme především:

- jméno aktivity
- délka trvání aktivity
- datum počátku a konce práce na aktivitě (při definici návazností se přepočítává automaticky podle návazností a délky trvání aktivity)
- předchůdci
- zdroje

Jednotlivé aktivity projektu jsme již specifikovali v předchozí kapitole, konkrétně v tabulce 9.1, proto údaje pouze přepíšeme do GUI programu. Začneme definicí jmen projektových činností. Všimněte si, že při vytváření Project Libre automaticky doplňuje některé informace - předpokládá délku trvání 1 den s počátkem totožným se startem projektu.

Informace zadané do GUI se také přímo zaznamenávají v pravé části obrazovky v harmonogramu. Předpokládaná kritická cesta je zvýrazněna červenou barvou. Další sloupce pro aktivity zatím vyplněné nejsou - ty budeme muset definovat sami.

Nejprve provedeme změnu délky jednotlivých činností projektu. Ty jsou aktuálně nastaveny na 1 den a my z tabulky 9.1 víme, že délky řady činností budou odlišné. Délku můžeme specifikovat ve dnech, týdnech nebo měsících, Project Libre přitom předpokládá, že zadáváme délku ve dnech. 5 v položce délka je tedy interpretováno jako 5 dní (v případě, že se používá běžný kalendář bussiness hours, pak se jedná také o ekvivalent 1 týdne). Délkovou jednotku můžeme přidat přímo za číselnou specifikaci délky, např. 3d (tři dny), 2w (dva týdny), 1m (jeden měsíc). Zkratky časových jednotek jsou odvozeny z angličtiny, tedy d = day, w = week, m = month.

Mějte na paměti, že týden může znamenat také něco jiného než si myslíte, v závislosti na kalendáři, který se použije, implicitně přitom máme přednastaven kalendář bussiness hours. Kalendáře ale lze měnit na úrovni jednotlivých činností - tedy každá činnost může mít jiný kalendář a proto se definice času může poněkud zkomplikovat

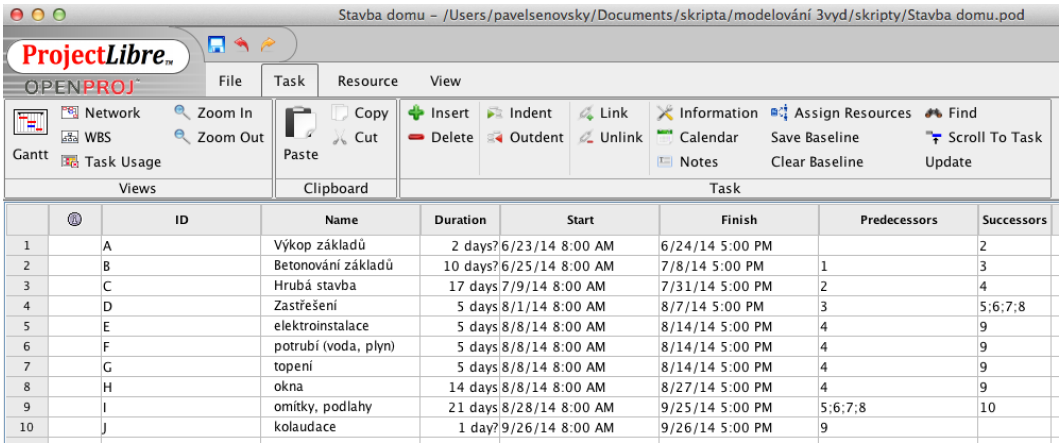
Informace o jednotlivých činnostech z tab. 9.1 se také liší v jiné podstatné charakteristice. V tabulce jsme jednotlivé činnosti měli identifikované pomocí písmen A - J, v Project Libre jsou ale identifikovány číslem řádku, na kterém jsou zapsány (1-10). Abychom zajistili kompatibilitu (a jednoduchost definice návazností), přidáme další sloupec, do kterého zapíšeme identifikační písmena.

Přidání nového sloupce je snadné. Klikneme prostě pravým tlačítkem myši na záhlaví sloupce, kam chceme nový sloupec přidat a v kontextovém menu zvolíme vložit sloupec (Insert Column ...). Kontextové menu nám také umožňuje sloupce odebírat. Tady doporučuji jistou zdrženlivost - nechejte si zobrazené alespoň základní sloupce.

Volba vložit sloupec spustí dialogové okno pro výběr typu sloupce. Typy jsou seřazené v přehledném seznamu. Z některých názvů je jasně patrné, k čemu sloupec slouží, u jiných tomu tak není. Pro náš účel potřebujeme sloupec typu Text - protože je jich tam více, zvolíme *Text1*. Jedná se o obecný sloupec, umožňující vložit jakýkoliv text. Po vložení sloupce změníme z jeho kontextového menu jeho název (volba Rename). Nové jméno bude ID. Do nově vytvořeného sloupce pak můžeme z tabulky 9.1 přepsat identifikátory.

Takže zbývá doplnění chybějících údajů - tedy návazností. V Ganttově diagramu již máme dostupný sloupec pro předchůdce, pokud ale preferujete zadaní spíše následníků - můžete, je ale potřeba přidat nový sloupec typu následník (successor). Project Libre mezi předchůdci a následníky přepočítává automaticky, takže musíme definovat pouze jedno.

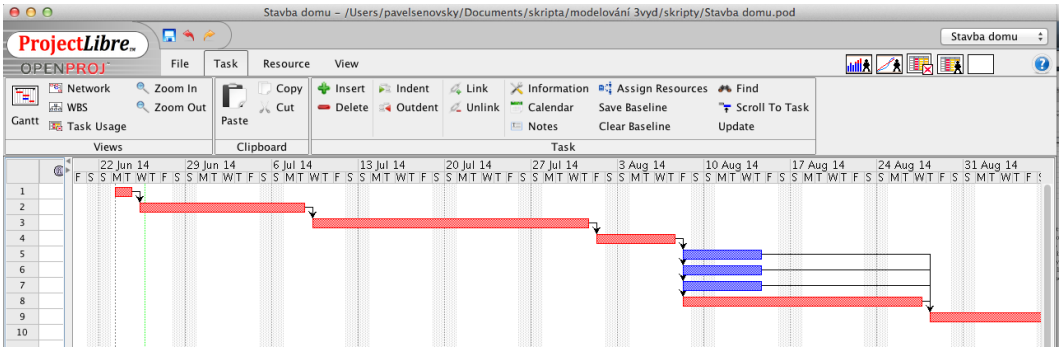
V tomto případě použiji předchůdce. Nezapomeňte, že identifikátor v Project Libre je číslo řádku, námi zadaný sloupec ID tedy slouží pro jednodušší překlad mezi číslem řádku a původním označením činnosti. Při zadávání většího množství např. předchůdců, oddělujeme jednotlivé předchůdce středníkem. Ve výsledku získáte zápis podobný tomu na obr. 9.4.



	ID	Name	Duration	Start	Finish	Predecessors	Successors
1	A	Výkop základů	2 days?	6/23/14 8:00 AM	6/24/14 5:00 PM		2
2	B	Betonování základů	10 days?	6/25/14 8:00 AM	7/8/14 5:00 PM	1	3
3	C	Hrubá stavba	17 days	7/9/14 8:00 AM	7/31/14 5:00 PM	2	4
4	D	Zastřešení	5 days	8/1/14 8:00 AM	8/7/14 5:00 PM	3	5;6;7;8
5	E	elektroinstalace	5 days	8/8/14 8:00 AM	8/14/14 5:00 PM	4	9
6	F	potrubí (voda, plyn)	5 days	8/8/14 8:00 AM	8/14/14 5:00 PM	4	9
7	G	topení	5 days	8/8/14 8:00 AM	8/14/14 5:00 PM	4	9
8	H	okna	14 days	8/8/14 8:00 AM	8/27/14 5:00 PM	4	9
9	I	omítky, podlahy	21 days	8/28/14 8:00 AM	9/25/14 5:00 PM	5;6;7;8	10
10	J	kolaudace	1 day?	9/26/14 8:00 AM	9/26/14 5:00 PM	9	

Obrázek 9.4: Project Libre - definice Ganttova diagramu

Zadáním předchůdců/následovníků jsme také nastavili spojení mezi jednotlivými činnostmi, proto harmonogram v Ganttově diagramu již začíná dávat trošku lepší smysl, viz obr. 9.5.



Obrázek 9.5: Project Libre - harmonogram

V harmonogramu jednotlivým činnostem odpovídají čáry, s délkou odpovídající době trvání činnosti. Všimněte si šedých svislých sloupců, které odpovídají dnům pracovního klidu. Činnosti jsou

také rozlišeny barevně - činnosti na kritické cestě jsou zvýrazněny červenou barvou, zatímco nekritické činnosti jsou znázorněny barvou modrou. Vzájemná návaznost je zachycena pomocí šipek.

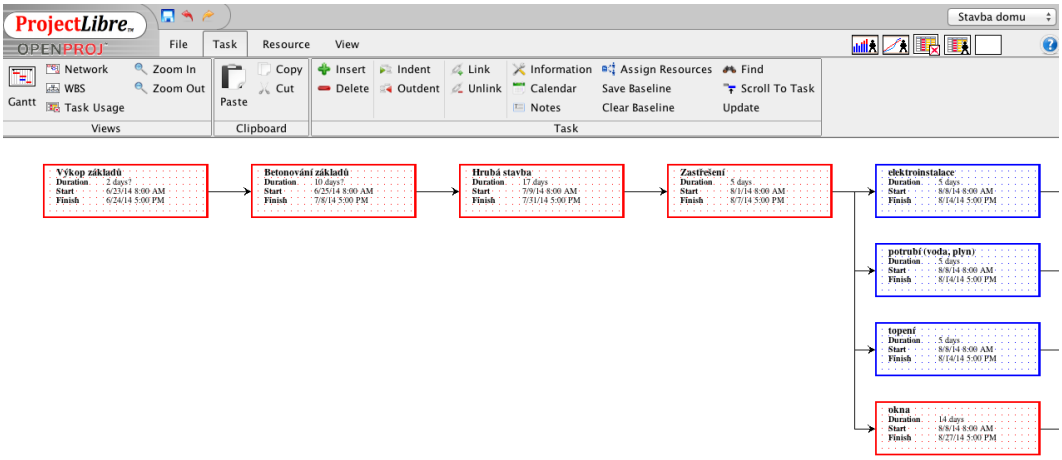
V našem případě jsou činnosti seřazeny, ale harmonogram toto nevyžaduje - stejnou funkčnost dosáhneme i když budou činnosti v seznamu činností zaznamenány na přeskáčku.

V případě, že bychom vytvářeli harmonogram ručně, museli bychom se na tomto místě zastavit, protože další manipulace by prostě technicky nebyla možná. Softwarová realizace harmonogramu dává uživateli ale další možnosti. Pomocí tažení myší lze celou činnost posunout v čase. Jelikož tažení samo o sobě nemění strukturu projektu (návaznost činností), termíny navazujících činností se přepočtou automaticky. Tažením počátku nebo konce lze změnit délku trvání činnosti. Konečně lze sledovat procento splnění další činnosti, což je nedocenitelná funkce pro účely řízení průběhu projektu.

Pro změnu procenta splnění najedeme myší na začátek činnosti - kurzor se změní do podoby %> (procento s šipkou doprava). Procento splnění měníme kliknutím a tažením. Vizuálně pak bude možné procento splnění činnosti identifikovat pomocí vyplnění činnosti černou barvou proporcionálně odpovídající procentu splnění.

Alternativním pohledem na činnosti a jejich vzájemnou vazbu je síťový diagram. V síťovém diagramu jednotlivé činnosti jsou reprezentovány uzly sítě (na rozdíl od ručního zpracování síťového grafu, kde činnosti byly reprezentovány hranami sítě). Spojení pak umožňují identifikovat jednoduše, které činnosti musí být dokončeny předtím, než mohou ostatní započít. Síťový diagram je generován automaticky na základě základních časových vlastností činností.

I v síťovém grafu je možno vizuálně identifikovat kritickou cestu - zvýraznění je provedeno opět červenou barvou. V Project Libre se síťový pohled aktivuje Task tab -> tool Network. Pro náš příklad je síťový graf zachycen na obr. 9.6.



Obrázek 9.6: Project Libre - síťový graf

Máme tedy k dispozici dva různé pohledy - harmonogram/Ganttův diagram a síťový diagram, které zachycují projekt a jeho průběh v čase, stejně jako kritickou cestu. Možná Vás napadla otázka - proč je to potřeba? Pro pochopení rozdílu je potřeba se zaměřit na rozdíly mezi oběma pohledy. Síťový diagram umožňuje, konsolidovat činnosti (a některé základní informace o nic) a jejich návaznosti na relativně malém prostoru. Harmonogram obvykle zabírá prostor větší, umožňuje ale lepší časové pochopení průběhu projektu a umožňuje snadnější manipulaci s časovými, lidskými i materiálovými zdroji (rezervami), stejně jako sledování průběhu projektu.

V předchozím odstavci jsme zmínili zdroje - může nám Project Libre pomoci i s nimi? Ano, může - zdroje, dostupné pro projekt, lze spravovat na záložce zdrojů (Resource).

Vytvořme tři zdroje - pracovník 1 a 2 a písek. Pro zdroje je potřeba specifikovat jejich typ. Typem se přitom rozumí rozlišení mezi zdroje lidskými a materiálovými. Oba druhy zdrojů mají trochu odlišné vlastnosti, se kterými se dále pracuje.

Pro lidské zdroje vybereme typ pracovní (work). Těmto zdrojům je možné přiřadit e-mail, seskupit je do pracovních týmů apod. Lidským zdrojům je nutno přiřadit také maximální možné využití (max. units). V případě lidí pracujeme s procenty, které si lze představit jako úvazek. 100 % proto znamená, že pracovníka můžeme plně využít pro projekt, 50 %, že můžeme využít pouze polovinu jeho kapacity (poloviční úvazek). Lidské zdroje nemá smysl kapacitně využívat na více než 150 %.

S každým pracovníkem jsou spojeny některé náklady - jedná se o hodinovou mzdu (standard rate) a sazbu za přesčasy (overtime rate) - tedy za dobu, kdy by došlo k překročení „nasmlouvaného“ pracovního výkonu, jak je specifikován procentem úvazku. Přesčasová sazba je obvykle vyšší než sazba standardní.

Náklady na použití (cost per usage) jsou náklady přímo spojené s nasazením pracovníka, které ale nelze chápat jako mzdu. Např. zde může být náklad na dopravu dělníků na místo stavby z ubytovny apod.

Náběh nákladů (Accrue at) umožňuje specifikovat, kdy náklad fyzicky vznikne - kdy bude nutné náklad reálně zaplatit. V tomto případě máme tři možnosti:

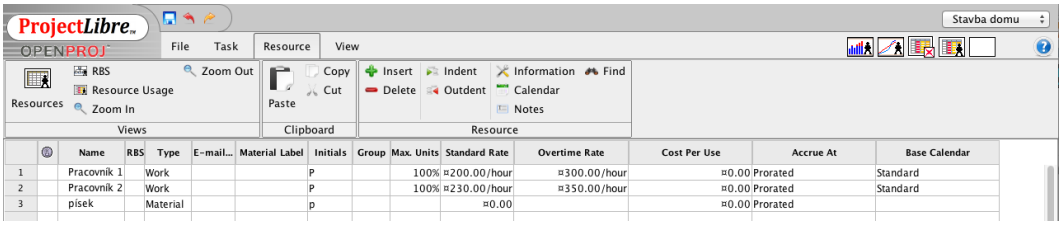
- na začátku,
- na konci,
- průběžně.

Všechny tři způsoby mohou být v případě lidských zdrojů použity. Do průběžného financování spadá běžný pracovní poměr - tedy pracovník je kmenovým zaměstnancem firmy a dostává běžnou mzdu splatnou v dohodnutých termínech. Financování na konci volíme v případě, že potřebujeme získat kontrolu nad určitým pracovním výkonem. Placení se provede až po jeho dokončení (převezmeme práci a zaplatíme). Placení na začátku předpokládá nutnost finanční investice nutné pro provedení prací - např. že součástí je nákup něčeho (např. materiálu), který je nutný pro výkon práce. V tomto případě, ale takový materiál již nezavádíme do materiálových zdrojů. Tento typ financování se používá spíše pro konraktory (subdodávky).

Každému pracovníkovi je možné nastavit jeho vlastní kalendář a to standardní (business hours), 24 hodin a noční směny (night shift). Nastavení kalendáře rozhoduje o tom, jak a kdy bude práce odvedena.

Pro materiálové zdroje řada sloupců nemá smysl, takže je nepoužíváme, řada sloupců má ale trošku jiný smysl než v případě zdrojů lidských. Standardní sazba (Standard rate) nemá charakter hodinové sazby mzdy, ale cena, za kterou pořizujeme jednotku zdroje. Jednotky v tomto případě mohou být jakékoliv - tuny, kilogramy, litry, cena za kus, čtvereční metr apod. Cena za užití je interpretačně podobná - i materiálový zdroj je potřeba dostat na místo, popř. nějak zpracovat.

Příklad definic zdrojů naleznete na obr. 9.7.



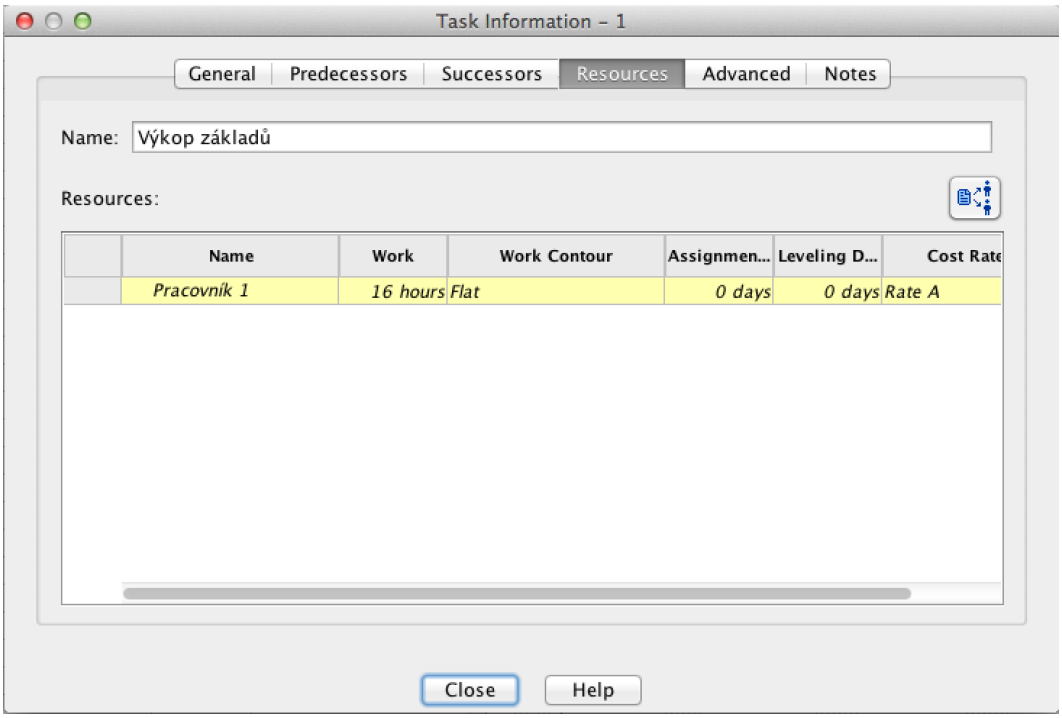
Obrázek 9.7: Project Libre - definice zdrojů

Vraťme se zpět do Ganttova diagramu - právě zde totiž můžeme přiřadit zdroje k jednotlivým úkolům. Přiřazení je možné provést dvojím způsobem, buď klikneme myší do sloupce jméno zdroje (resource name) nebo ve vlastnostech jednotlivých činnosti (spustí se dvojklikem na dané vlastnosti). Dialogové okno s podrobnostmi činnosti umožňuje nastavování celé řady dalších zajímavých vlastností vztahujících se k činnostem je na obr. 9.8.

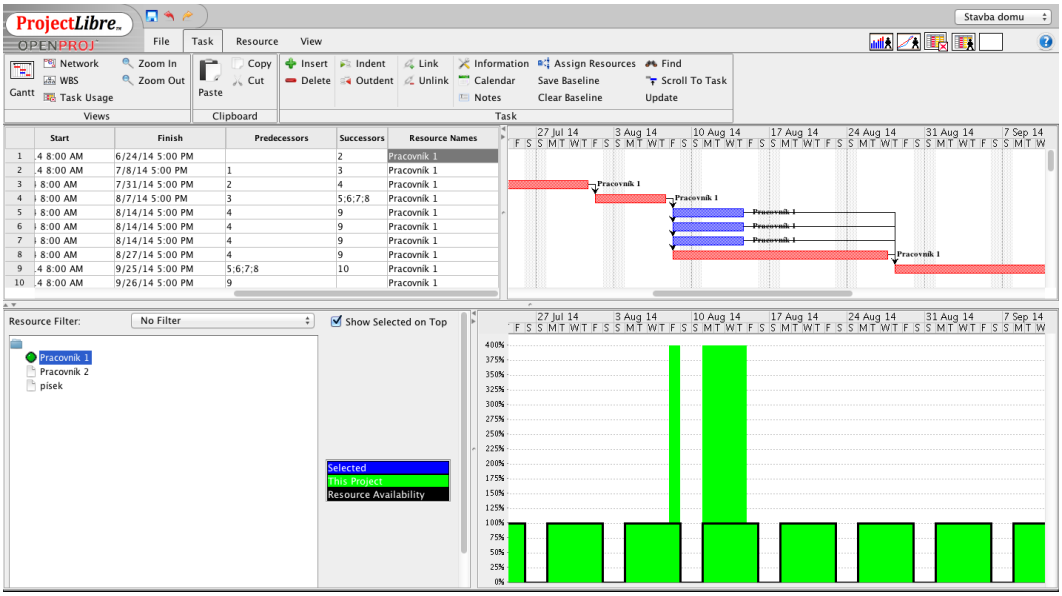
Zkusme pokusně přiřadit všem činnostem pracovníka 1. Celý proces můžete uspíšit prostým kopírováním jména zdroje ve sloupci jmen zdrojů na záložce Ganttova diagramu.

Podívejme se, jak jsou naše zdroje využívány. Nejprve prozkoumáme histogram zdrojů. Histogram můžeme spustit kliknutím na ikonu histogramu v pravém horním rohu okna programu. Pro náš příklad je histogram zachycen na obr. 9.9.

Histogram je spojen s harmonogramem, tedy posunuje se časově tak, aby odpovídal aktuální pozici harmonogramu. Z našeho hlediska je nejzajímavější období projektu spojené s činnostmi 5 - 8 (E - H). Z histogramu nám vyplývá, že zdroj Pracovník 1 bude mít pekelný týden, kdy jeho pracovní den bude mít 32 hodin. Naše využití tohoto zdroje je tedy na 400 %, což je samozřejmě logický nesmysl. Všimněte si, že Project Libre odlišuje vizuálně mezi běžnou dobou a přesčasy - to nám umožňuje jednoduše vizuálně identifikovat jakákoliv, z hlediska využití zdrojů, problémová místa - a to i v případě, že nejsou tak extrémní, jako v našem příkladu.

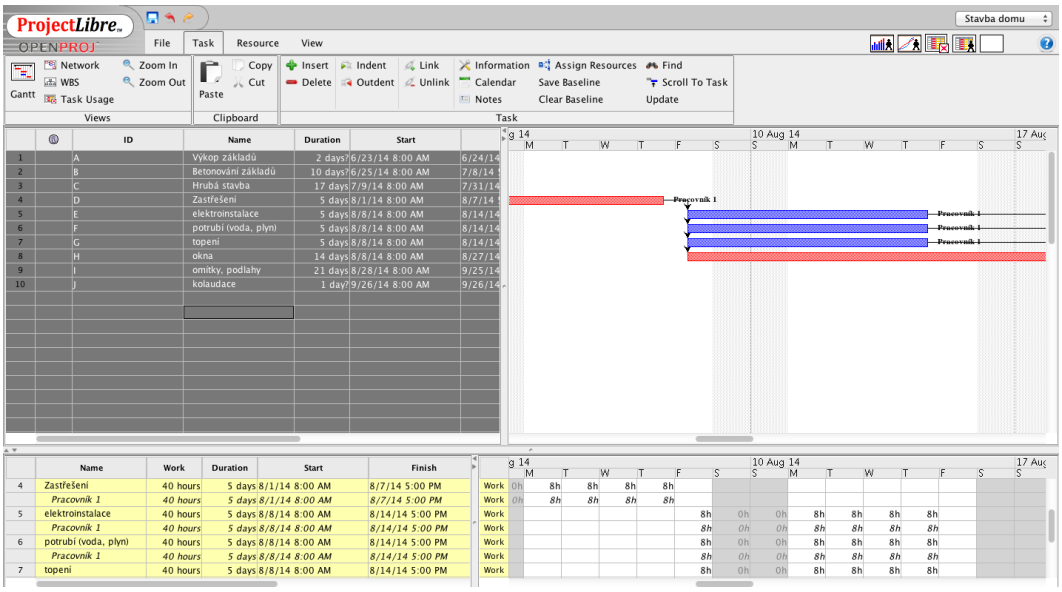


Obrázek 9.8: Project Libre - podrobnosti o činnosti



Obrázek 9.9: Project Libre - histogram zdrojů

Můžeme se také podívat na zdroje z hlediska pracovních hodin - volbou využití zdrojů (resource usage), viz obr. 9.10. Software samotný tedy disponuje celou řadou nástrojů, které nám umožní identifikovat podstatu problému a umožnit nám tak něco s ním udělat.



Obrázek 9.10: Project Libre - užití zdrojů

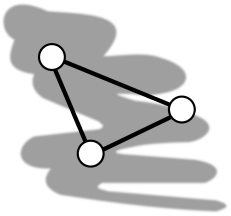
Management projektu

Vytvořte projekt podle vlastního zadání, pokud se Vám nechce přemýšlet - vytvořte projekt popisující Vaše studium v tomto semestru. Pozornost v takovém případě věnujte termínům semestrálních projektů, případných testů, termínům zkoušek apod.

Kapitola 10

Nástroje pro týmovou spolupráci

Náhled kapitoly
V této kapitole se zaměříme na základní platformu použitelnou pro týmovou spolupráci - Microsoft OneDrive a technologie okolo něj, konkrétně SharePoint a také použití MS Word pro práci na souborech.



Jiná řešení
V kapitole se zaměřujeme na řešení od Microsoftu. Jedním z důvodů je to, že Microsoft tato řešení má poměrně dobře odladěna a nástroje, které k práci s tímto cloudem používáme jsou dobře známy a jsou rozšířené. Neznamená to ale, že se jedná o jediné možné řešení.

S úspěchem můžete použít také iCloud řešení od společnosti Apple, G-suite od Google. Collabora poskytuje velmi sofistikované prostředí pro spolupráci, můžete dokonce postavit vlastní cloudové řešení např. na bázi NextCloud. Každé z těchto řešení sice má jistá specifika, jsou ale zde společné prvky. Migrovat mezi řešeními od různých výrobců je tak možné a relativně jednoduché.

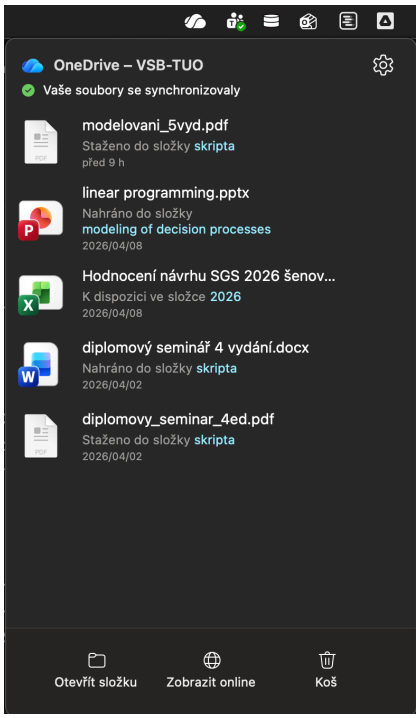
10.1 OneDrive

Základem každého kooperativní řešení je cloudová platforma, umožňující sdílení souborů, práce na nich, apod. V případě řešení od Microsoftu tuto úlohu plní OneDrive. Ten slouží jednak jako úložiště souborů pro jednotlivé uživatele, jednak poskytuje rozhraní přes které mohou fungovat další služby plnící specifické služby, jako např.:

- SharePoint - pro webová sídla a sdílení informací skupin pracovníků
- MS Teams - pro týmovou komunikaci (viz samostatná kapitola)
- MS Planner - pro plánování a úkolování
- Office 365 - online/offline editory souborů
- a řada dalších

Služby se tedy mohou připojovat k OneDrive prostřednictvím dobře dokumentovaného API místo a využívat služby cloudu včetně řízení práv k souborům, apod. a nemusí tak řešit tyto problémy samy. To výrazně zrychluje a zefektivňuje vývoj. Architektura také umožňuje, aby specializované služby byly skládány do větších funkčních celků použitelných i pro řešení celkem složitých organizačních problémů.

One drive cloudové složky má také komponentu která běží na klientských zařízení jako jsou počítače, notebooky, ale také mobilní zařízení jako jsou mobilní telefony a tablety (iOS i Android).



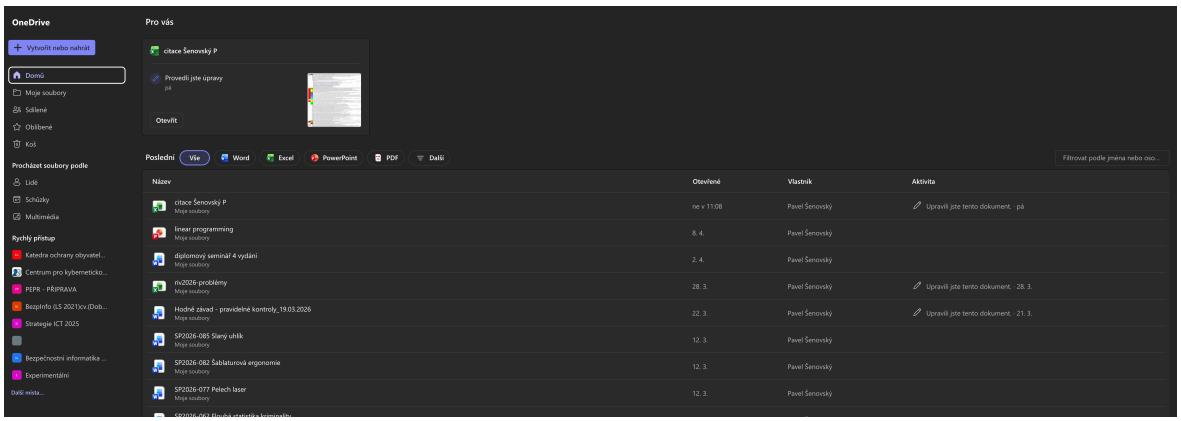
Obrázek 10.1: OneDrive synchronizační komponenta běžící na koncovém zařízení

Na počítačích funguje jako on-line synchronizační nástroj, který synchronizuje složku OneDrive v počítači a v cloudu. Pro zajištění této funkce musí OneDrive na koncovém zařízení běžet stále jako služba, viz obr. 10.1.

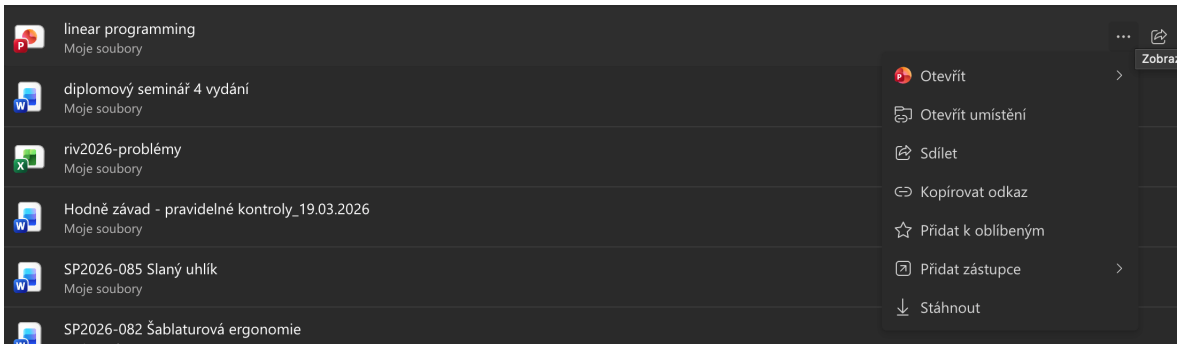
Kromě této komponenty je možno přistoupit do OneDrive přímo přes webové rozhraní služby prostřednictvími webového prohlížeče nebo aplikací jako MS Teams nebo MS Outlook, které jsou také webové a přístup ke službě poskytují jako komponentu. Rozhraní OneDrive při přístupu přes webové rozhraní vypadá jako na obr. 10.2.

Alternativně lze přistoupit k OneDrive přímo z počítače ve složce OneDrive. Pro zajištění funkčnosti musí na daném počítači být spuštěna služba OneDrive. Bez této služby nově vytvořené soubory ve složce se neuloží na cloud (zůstávají tedy pouze lokálně a složka bude fungovat jako normální složka). Většina souborů ve OneDrive není uložena lokálně. Ve složce tak sice uvidíte soubory, ale po kliknutí na ně, se Vám nemusí podařit je spustit. V případě, že běží služba OneDrive tak po kliknití soubor nejprve stáhne a to umožní soubor otevřít.

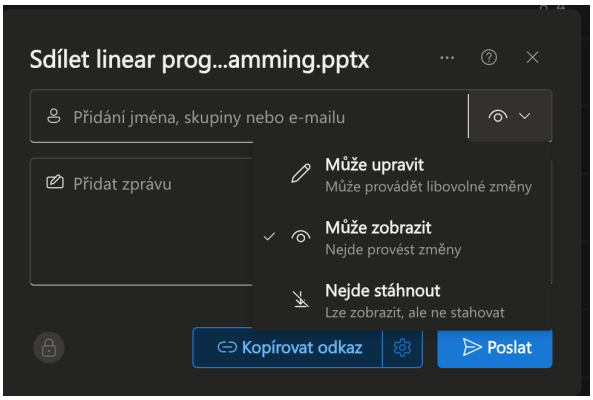
Soubory lze otevírat také přímo ve webovém rozhraní. V takovém případě se načte online verze produktu, např. MS Word a soubor je možno prohlížet a editovat podle potřeby.



Obrázek 10.2: Webové rozhraní OneDrive (přístup přes MS Teams



Obrázek 10.3: Sdílení pomocí webového rozhraní OneDrive



Obrázek 10.4: Sdílení souboru - komu sdílet?

Jak je to se spoluprací? V OneDrive můžete soubory a složky sdílet. To uděláme z kontextového menu daného souboru, viz např. obr. 10.3.

Všimněte si ikony v kontextovém menu u Sdílení. Tato ikona je přítomna také přímo v řádku (vedle „...“) a kontextové menu tak ke sdílení nemusíme vůbec použít. Sdílet lze pro jednotlivce, seznam osob nebo skupinu osob, viz obr. 10.4.

Jednotlivce zadáváme obvykle pomocí jejich uživatelských jmen nebo adres. Můžeme použít také skupinové adresy spojené se službami MS Teams (jednotlivé týmy) nebo s weby provozovanými na SharePoint. Sdílení se implicitně provádí s právy pro čtení. Pokud chceme, aby osoba, se kterou soubor sdílíme, mohla tento soubor také editovat musíme změnit tato práva směrem na *Může upravit*.

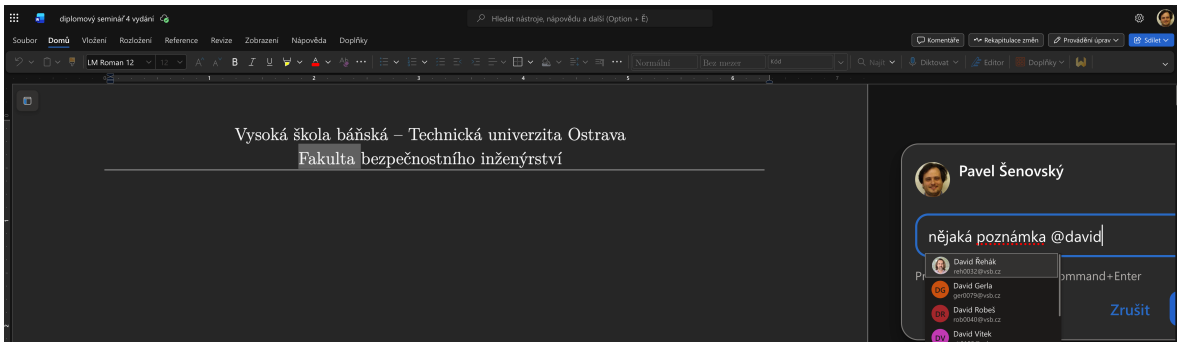
Prvotní informace o tom, že soubor byl nasdílen se rozešle e-mailem. Pokud chcete přidat i nějakou zprávu, kromě automaticky generované zprávy, že soubor byl nasdílen, můžete ji zapsat do kolonky *přidat zprávu*.

Sdílení OneDrive
Nasdílejte soubor, otevřete jej a spolupracujte s dalšími studenty na vytváření tohoto dokumentu.

Tento způsob sdílení využíváme v případě, že existují lidé nebo skupiny lidí u kterých víme, že na tvorbě dokumentu by měli mít možnost participovat a proto jim jej sdílíme.

10.2 Spolupráce nad dokumentem v MS Word

Co ale v případě, že primárně dokument vytváříme my, ale chtěli bychom znát např. názor někoho jiného na určitý segment textu v dokumentu. Třeba si nejsme jistí, jestli používáme správnou metodu,



Obrázek 10.5: Komentáře v MS Word umožňují použití @zmínek

nebo jestli ji používáme správně. I v takovém případě lze sdílet. MS Word nám ale umožňuje použít i jiný způsob, s použitím *komentářů*.

V případě, že zmíníme nějakou osobu pomocí @zmínky, pak se této osobě automaticky nasdílí dokument a přijde jí upozorňující mail o tom, že dokument byl nasdílen s textem komentáře. Daná osoba pak může dokument otevřít, komentovat jej nebo provádět úpravy.

Takže máme nasdíleno. Na dokumentu nyní pracuje 10 lidí ... zároveň. Jelikož všechny změny se projevují se zpožděním do jedné sekundy, lze říci, že práce probíhá v reálném čase. Jak zajistit, že si jednotliví uživatelé takzvaně „nepolezou do zelí“?

Technicky toto bohužel nelze vynutit. Co můžete udělat, je použít netechnické prostředky pro udržení kontroly. Ty v sobě zahrnují, způsob, jakým píšete věci, jak jste pozvali další osoby ke spolupráci. Bylo součástí pozvánky, jaký způsob zapojení do tvorby dokumentu se od nich očekává?

Obvykle existuje osoba, která za dokument nese nějakou odpovědnost, ostatní se pak pouze na tvorbě tohoto dokumentu podílejí. Odpovědná osoba by měla dokument vytvořit a zvolit vhodnou formu pozvání ostatních k spolupráci na jeho přípravě.

Rozdělit práci můžeme jednoduše tak, že vytvoříme osnovu a do komentáře k jednotlivým nadpisům vložíme zmínky osob, které by měly na dané sekci textu participovat. To přirozeným způsobem omezí aktivity těchto osob v dokumentu - nasměruje je to do jeho určité části. Technicky jim to nezabrání dělat změny i někde jinde.

Co můžeme udělat je zapnou systém revizí. Toto umožní, aby změny realizované v dokumentu byly jednoznačně odlišeny od původního, popř. schváleného textu. Na osobě odpovědné pak je, aby navrhované změny v dokumentu přijímala, odmítala, nebo např. formou komentářů vracela zpět k další úpravě.

Obvykle nestačí pouze doplnit text, je potřeba jej alespoň stručně okomentovat ve smyslu co je cílem, proč, atd. Toto lze udělat opět pomocí komentářů. Podrobnější informace o doplňovaném textu umožní dalším osobám lépe pochopit, co chcete danou sekcí docílit a mohou Vám při tvorbě lépe pomoci.

Na komentáře lze reagovat, čímž mohou vzniknout i poměrně obsáhlé konverzace nad tématem. V okamžiku kdy komentář naplní svůj účel a již není potřeba máme dvě možnosti komentář vyřešit nebo jej úplně smazat. Rozdíl mezi těmito přístupy je v tom, že vyřešený formulář zůstává v dokumentu přítomen a může být znovu otevře, pokud se později ukáže, že vyřešení bylo předčasné. Výmaz komentáře oproti komentář odebere zcela z dokumentu.

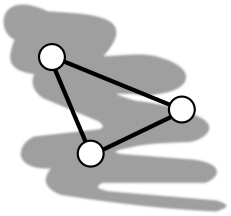
Z praktického pohledu bychom mohli udělat jednoduché doporučení - komentáře označujeme jako vyřešené, pouze osoba odpovědná za dokument může komentáře také mazat.

Autoři participující na tvorbě dokumentu by se měli chovat ohleduplně vůči sobě navzájem a omezit vzájemné zasahování do textu. V případě odlišného názoru na řešení by tento názor měl být zformulován do komentáře, poskytujícího alternativní vhled do určité problematiky. Je možné, že tímto způsobem jednotliví autoři v konfliktu najdou společné řešení. Pokud se tak ale nestane bude hlavním arbitrem osoba odpovědná za dokument jako celek.

Výsledný dokument by měl mít vyřešené (a smazané) všechny komentáře a také všechny revize.

Z pohledu osoby zodpovědné za tvorby dokumentu není identifikace změn ani při použití výše uvedených nástrojů úplně snadná. Pokud totiž na dokumentu pracuje velké množství lidí, pak v důsledku jejich činnosti vzniká také velké množství změn. Není optimální cestou procházet dokument od začátku do konce a tyto změny aktivně vyhledávat. Word pro tyto účely poskytuje nástroje, pro

Pokusy Word
vyzkoušejte si kolaborativní práci na společném dokumentu. Používejte komentáře se zmínkami. Podívejte se jak rychle přicházejí e-amily s upozorněním. Používejte revize, získejte trochu zkušeností s tímto způsobem práce.



Oběh dokumentů
tradičně spolupráce na dokumentech byla realizována etapovitě. To fungovalo tak, že někdo dokument napsal a pak jej e-mailem poslal dále k připomínkováním, ten po dokončení připomínek jej posílá zpět, nebo jej posílá k úpravám další osobě.

Tento způsobe je extrémně pomalý a v dnešní době již zcela zbytečný. Použití cloudových služeb celkovou spoluprácí silně akceleruje, za předpokladu, že manažersky tvorbu dokumentu udržíte pod kontrolou.

sledování aktivity, viz obr. 10.6.

Jsou poskytovány dva základní nástroje: *Rekapitulace změn*, která popisuje chronologicky změny, kterými prošel dokument a *komentáře*, které poskytují na jednom místě přehled všech dosud nevyřešených komentářů. Myšlenka je v obou případech stejná - nechceme, aby musel procházet celý dokument a hledat změny, popř. nové komentáře, na které má reagovat, ale umožnit rychlou identifikaci pasáží textu, které se v dokumentu změnily, nebo komentářů na které je potřeba reagovat a skočit na ně.

Jedná se tedy o podobný koncept jako obsah, který nám umožňuje navigovat strukturou nadpisů v dokumentu. V případě komentářů a změn dokumentu navigujeme podle komentářů, resp. změn realizovaných v dokumentu. Obecně platí, že čím větší (délkou textu) dokument a čím více lidí se podílí na jeho vzniku, tím více oceníte tuto pokročilou funkcionalitu.

10.3 SharePoint

Posledním nástrojem, který probereme v této kapitole je *SharePoint*. Jedná se nástroj Microsoftu určený pro:

- přípravu veřejně přístupných webových sídel
- uzavřených webových sídel pro sdílení informací v týmech

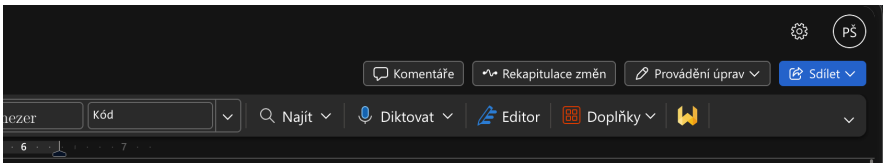
SharePoint je poměrně flexibilní. Webová sídla je možné vytvářet jednak samostatně, přímo pomocí této služby. Sídla SharePoint se ale vytvářejí automaticky např. společně s vytvořením týmu v MS Teams. V obou případech se jedná o jiný typ sídla.

Týmová sídla jsou specifická tím, že k nim mohou přistupovat pouze členové týmu. Oproti tomu, veřejně přístupná sídla jsou dostupná z logiky věci všem bez nutnosti přihlašování. V obou případech se využívá stejné rozhraní Content Management System (CMS) systému pro návrh obsahu sídla.

Při vytváření sídla si vybereme šablonu vzhledu a pak dle potřeby vkládáme jednotlivé stránky popř. komponenty, jako jsou obrázky, tabulky apod. Veřejně dostupná webová sídla sice jak bylo uvedeno výše SharePoint podporuje, ale není to primární úkol tohoto nástroje, tím je podpora práce v týmech a to nám vede na uzavřený týmová sídla, kde ale je potřeba používat také trochu jiné nástroje.

SharePoint z tohoto pohledu využívá úzké integrovanosti komponent v rámci systému Microsoft 365. Do takového sídla proto lze bez problému integrovat:

- sdílené soubory týmu v OneDrive



Obrázek 10.6: Sledování změn s MS Word online

- týmové úkoly v Planner
- interaktivní seznamy a formuláře
- a další

Výše uvedené komponenty u veřejně dostupných sídel nemají příliš smysl, ale v okamžiku, kdy potřebujete koordinovat celý tým mají nezastupitelnou roli. SharePoint se v takovém případě může stát jednak zdrojem „pravdy“ jednak výchozím bodem pro spolupráci.

Integrované soubory OneDrive lze v prostředí SharePoint rovnou otevřít a dle potřeby editovat. Propojované technologie jsou tak propojené velmi úzké a to do té míry, že často není možné jednoduše rozlišit, kde jedna končí a druhá začíná.

Výhodou prostředí webového sídla mimo jiné je také to, že při přístupu ke sdíleným souborům není potřeba nastavovat přístupová práva pro skupinu - použijí se prostě skupinová práva. SharePoint také dobře komunikuje aktivitu probíhající na sdílených souborech, integrací seznamu několika v poslední době změněných položek přímo do stránek.

Tímto způsobem je na první pohled patrné, na jakých souborech se zrovna pracuje, nebo které stránky byly v poslední době přidány nebo změněny.

Experimentování se SharePoint
Vyzkoušejte si pro Vaši studijní skupinu vytvořit SharePoint pro spolupráci a sdílení materiálů. Experimentujte s možnostmi, které Vám tato technologie poskytuje.

Kapitola 11

Komunikace v týmech - email, kalendáře, úkoly a plánování

Náhled kapitoly
V této kapitole se zaměříme na různé komunikační nástroje pro podporu práce týmů:

- MS Outlook pro řešení e-mailů, kalendářů a komunikace obecně
- plánování a úkolování (MS Planner, MS ToDo)
- rychlé shromažďování informací pomocí MS Loop

11.1 Úvod do Outlook

Outlook byl poměrně dlouhou dobu synonymem pro korporátního e-mailového klienta. Toto pojetí ale nikdy nebylo tak úplně pravda, protože kromě e-mailů podporoval také Outlook kalendáře, vedení kontaktů a měl integrovaný jednoduchý úkolovník. V současné době to již není pravda vůbec, protože komunikace v týmech se zásadním způsobem změnila a tak prosté maily již nestačí.

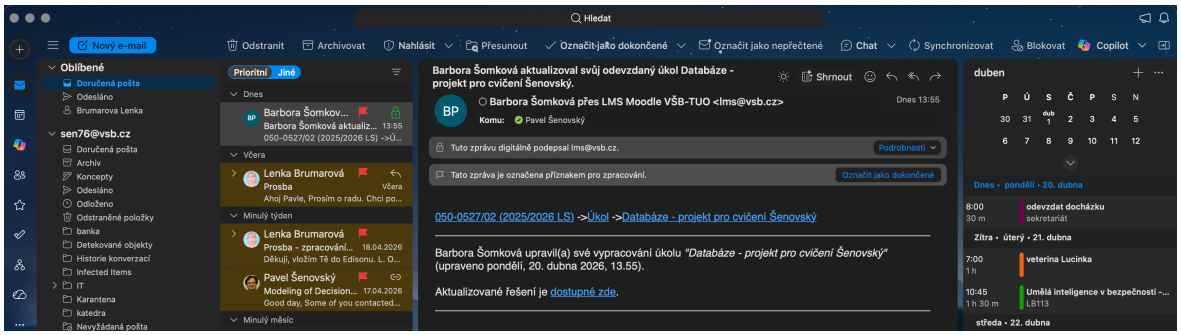
Moderní komunikační řešení v sobě musí obsahovat také:

- upozorňování na aktivitu ve komunikačních kanálech MS Teams
- integrovanou AI pro shrnutí komunikace nebo rychlý návrh konceptu e-mailu
- evidence změn v souborech, na kterých se v rámci týku pracuje včetně odpovědí na naše komentáře v nich
- možnost transformovat maily na úkoly včetně nastavování odpovědnosti a plánování schůzek k jeho řešení
- realizace hlasování o termínech konání schůzky
- a řada dalších funkcionalit

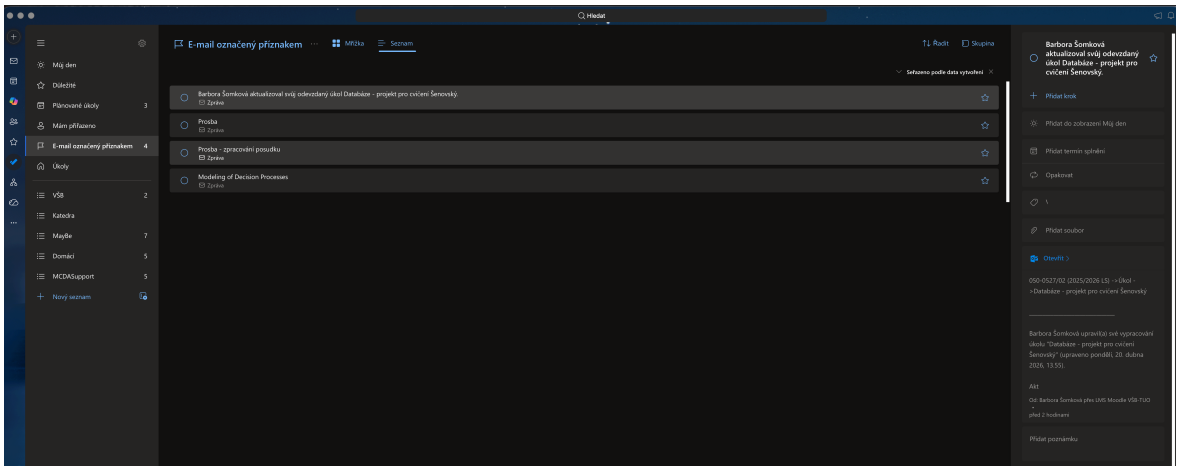
Pro splnění výše uvedené funkcionality je Outlook silně integrován do celého ekosystému Microsoft 365. Výše uvedené pokročilé funkčnosti je tak dosaženo integrací jinak samostatných služeb do rozhraní jednoho programu.

Pokud nemáte zkušenost s rozhraním moderních verzí MS Outlook, pak vypadají podobně jako na obr. 11.1. Toto rozhraní je ale potřeba brát pouze jako orientační. Outlook je možno poměrně výrazně upravit jednak z pohledu šablony vzhledu, jednak z pohledu zapnutých nástrojů. Uživatelé tak mají možnost do jisté míry nástroj přizpůsobit svému workflow.

Základní funkcionalitou e-mailového klienta se nemá smysl zabývat, Outlook v tomto ohledu se chová jako kterýkoliv jiný e-mailový klient. Zkusme proto jít do další funkcionality. A začněme možnosti „vlaječek“.



Obrázek 11.1: Základní rozhraní MS Outlook (se zapnutým přehledem událostí)



Obrázek 11.2: ToDo seznamy - propsání ovlaječkovaných mailů

11.2 Funkcionalita ToDo

Systém vlaječek je s námi v Outlooku, i jiných e-mailových klientech poměrně dlouhou dobu. Kliknutím na vlaječku e-mail označíte k pozdějšímu zpracování. E-mail je tímto způsobem zvýrazněn jednak vizuálně, což je samo o sobě účinné, navíc ale moderní verze Outlook integrují tuto funkcionalitu do úkolovníku. Ovlaječkovaný mail je tedy vnímán jako nesplněný úkol.

Tento úkol může mít přidělenu jinou osobu na slpnění, je možno k němu přidávat textovou poznámku, další souboru a je viditelný také v úkolovníku.

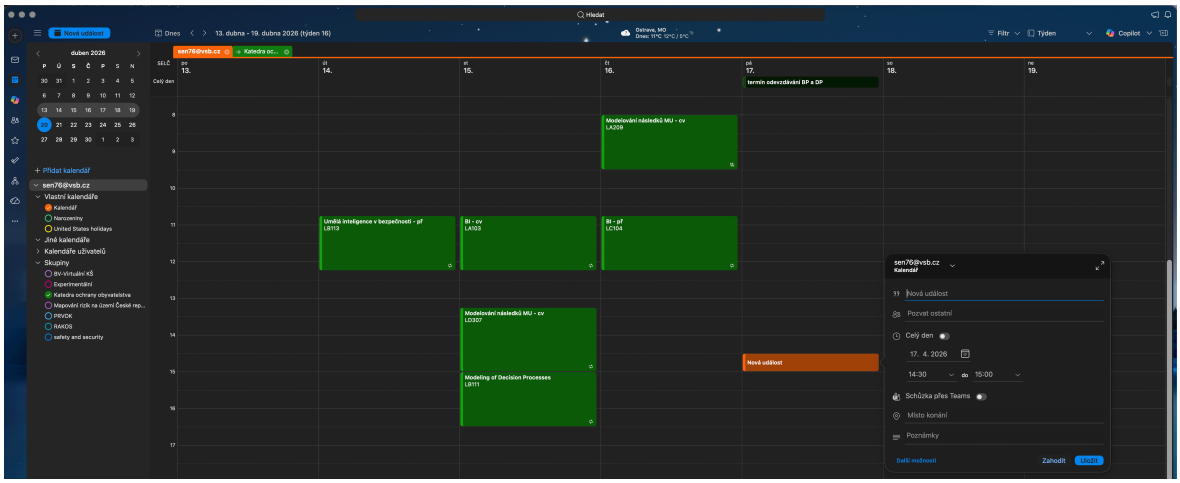
Úlohu úkolovníku vede aplikace MS ToDo. Tuto aplikaci je možno provozovat samostatně na všech populárních operačních systémech nebo můžete použít Outlook samotný pro zpřístupnění této funkcionality. Naleznete jej vlevo v panelu nástrojů - ikona fajfky. 4 e-maily z obr. 11.1 v MS ToDo vypadají jako na obr. 11.2.

Technicky je ToDo zcela samostatná služba. Outlook je ale schopen do ní publikovat s použitím jejího API úkoly a je schopen přebírat také zpět informaci o jejich plnění. Úkol lze ale také přesunout do zcela jiného seznamu. Ty ovlaječkované se automaticky směřují do dynamického seznamu *E-mail označený příznakem*. Pro vlastní účely si ale můžete přímo v ToDo vytvořit libovolný počet seznamů a úkoly definovat přímo do nich.

Kromě manuálně vytvořených seznamů jsou v ToDo také dva seznamy dynamické. První z nich *Můj den* se použije u termínovaných úkolů s termínem splnění dnes. Tento seznam tak poskytuje určitý vhled do možnosti organizování pracovního dne.

Pokud u úkolu zaškrtnete hvězdičku, označíte úkol jako důležitý a v takovém případě bude přístupný také ze seznamu *Důležité*.

Již jsme říkali, že úkolům lze přiřadit také jinou osobu ... bohužel ale ne pomocí aplikace Outlook nebo ToDo, k tomuto účelu potřebujeme použít MS Planner. K němu se trochu podrobněji dostaneme v následující kapitole věnované MS Teams.



Obrázek 11.3: Outlook kalendáře

11.3 Kalendáře

Jednou ze základních schopností dobrého komunikačního klienta je schopnost plánování času. K tomuto účelu je nutná funkcionalita kalendáře a schopnost vnášet do něj časově orientované informace. Původní implementace v Outlook umožňovala uživatelů v kalendáři plánovat vlastní čas. Moderní verze jsou ale v tomto ohledu pokročilejší a tak obsahují funkcionalitu k zvaní účastníků na schůzky. Schůzku přitom lze naplánovat jako prezenční nebo s možností dálkového přístupu pomocí MS Teams.

Na obr. 11.3 je vytvoření nové schůzky v existujícím kalendáři. Na obr. jsou dva mé kalendáře - jeden osobní spojený s účtem Microsoft 365 a druhý je týmový pro mou domácí katedru.

Každý tým tedy může mít vlastní kalendář, který můžeme prohlížet samostatné nebo společně integrované do jednoho kalendáře. V mém případě je kalendář nastaven tak, aby jako primární zdroj dat používal můj kalendář a do něj integroval data z kalendáře katedry. To, že tomu tak skutečně je uvidíte pomocí šipky u názvu kalendáře. Tímto způsobem se označují kalendáře, ze kterých se data integrují do hlavního formuláře.

Z tohoto pohledu není vlastně bezpodmínečně nutné se mezi kalendář přepínat, v některých případech to ale může dávat smysl. právě pro nějaké hustěji naplánované kolektivní kalendáře, kde hlavní část aktivit možná nutně nechceme vidět ve vlastním kalendáři, protože se budou týkat jiných osob, ale zároveň chceme mít přístup ke kalendáři jako takovému.

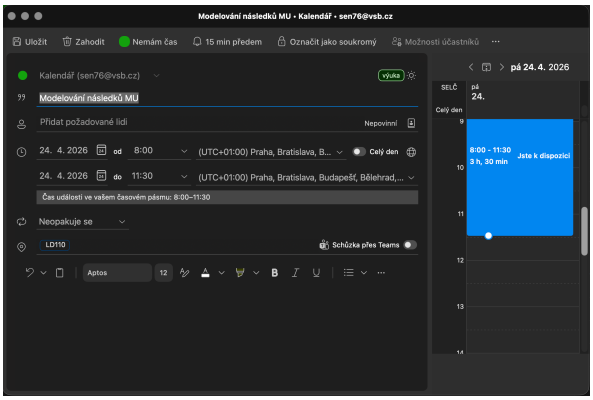
Na obr. 11.3 je znázorněn také prvotní krok k vytvoření události v kalendáři. Vlevo nahoře je vybraný kalendář, do kterého se bude událost zapisovat. Při vytváření událostí se nabízí předvolený hlavní kalendář. Tuto volbu ale můžeme změnit. Outlook pro každou další událost bude nabízet naposledy zvolený kalendář. Pokud se tedy při vytváření událostí často přepínáte, dbejte zvýšené pozornosti směrem k tomu, do kterého kalendáře událost vytváříte.

Volbu kalendáře lze sice kdykoliv změnit, zároveň se ale jedná o záležitost, která není na první pohled viditelná, pokud je kalendář používán v režimu s integrovanými údaji (vše skončí v jednom kalendáři). Událost ale může být viditelná i jiným osobám, než u kterých to očekáváte. K události lze také připojovat soubory a poznámky, což otevírá prostor pro „únik“ citlivých informací.

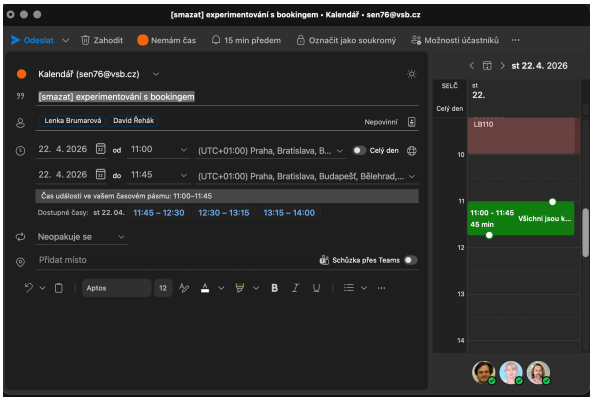
V podrobnostech události lze přiřazovat také účastníky, specifikovat, kde se událost koná prezenčně a zda je možno se připojit dálkově pomocí MS Teams, viz obr. 11.4. Pro zvýšení přehlednosti, lze přiřazovat události do jednotlivých kategorií a těmto kategoriím pak přiřazovat barvu. Tímto způsobem se zlepšuje vizuální přehlednost kalendáře.

Ne vždy je ale možno provést plánování události takto - tedy zvolením termínu. Často se stává, že termín je nejprve nutno dohodnout. Tradičním způsobem je rozeslat mailem několik termínů a doufat, že účastníci odpoví a shodnou na některém z termínů. Z praktických zkušeností Vám ale mohu říci, že příliš dobrou šanci v tomto případě nemáte.

Trochu lepší je použití služby jako je Doodle (https://doodle.com/en/) - zde si uživatelé prostě naklikají své časové preference a Vy jako organizátor výběr finalizujete. Tento způsob je lepší, protože vidíte výsledek v přehledném formuláři. Ani tento způsob ale není ideální, protože závisí na Vašem výběru času a Vy se můžete, ale také nemusíte trefit.



Obrázek 11.4: Editace události



Obrázek 11.5: Plánování času schůzky - kontrola volného času účastníků

Jaký další možnosti tedy máme? Máme kalendář, ve kterém jsou naše plánované schůzky a také ostatní si svůj čas plánují pomocí kalendářů. Kalendáře se ukládají online. Prakticky to pro nás znamená, že existuje zdroj informací, který lze použít pro efektivní plánování schůzek.

V okamžiku kdy přidáte k události účastníky, použije Outlook informaci z kalendářů účastníků k tomu, aby navrhl časy, ve kterých nemají účastníci nic naplánováno, viz obr. 11.5. Tímto způsobem se ale zpracovává pouze jeden den - plánovaný den schůzky.

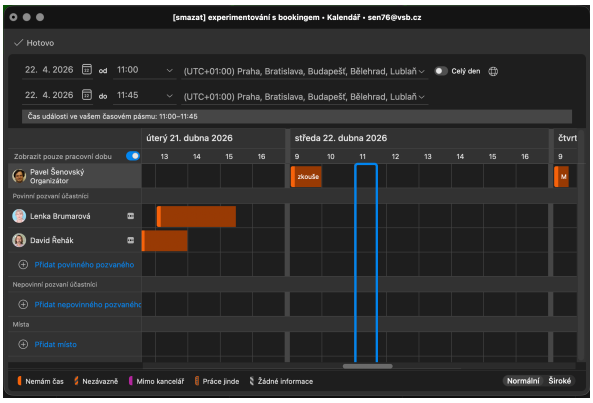
Co ale když jsme se neujednotili na konkrétním dni? Outlook poskytuje průvodce plánováním, který zobrazí kalendáře všech účastníků v přehledné časové ose s jasně patrným rozlišením volno/rezervovaný čas, viz obr. 11.6.

Zde lze vybrat jednodušeji čas schůzky.

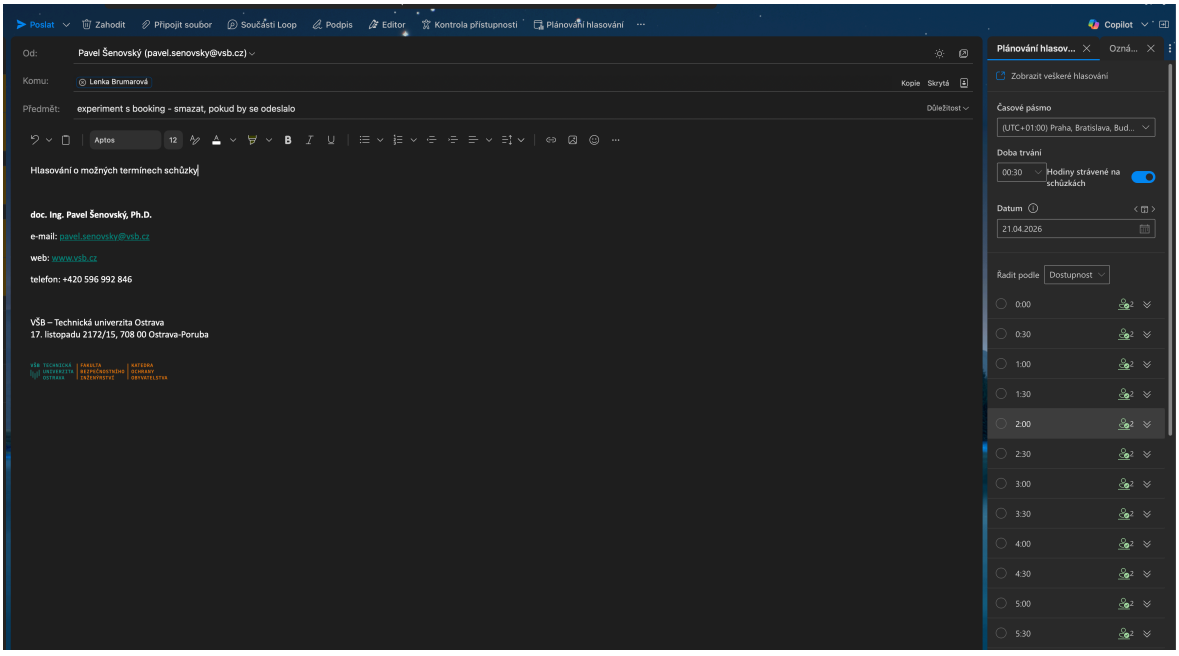
Co ale v případě, že bychom, že chceme nechat hlasovat o termínu? Pro tyto účely Outlook poskytuje tzv. *hlasování o plánu*. To lze vyvolat tak, že vytvoříte nový mail, do něj přidáte účastníky. Obvykle se doporučuje specifikovat nějaký průvodní text mailu. Před odesláním klikněte v panelu nástrojů na *Plánování hlasování*.

Zobrazí se rozhraní jako na obr. XX. Vpravo vybíráte několik termínů. Rovnou přitom vidíte, kolik z účastníků má v uvedené době čas (zelené postavičky) a kolik ne (červené postavičky). Předvolené termíny nemusí přitom nutně být v jednom dni. Po výběru všech možných termínů, o kterých se bude hlasovat, klikněte vpravo dole na další kroky průvodce, v rámci kterých lze nastavit místa a to zda schůzka je on-line a nechat vytvořit hlasovací formulář.

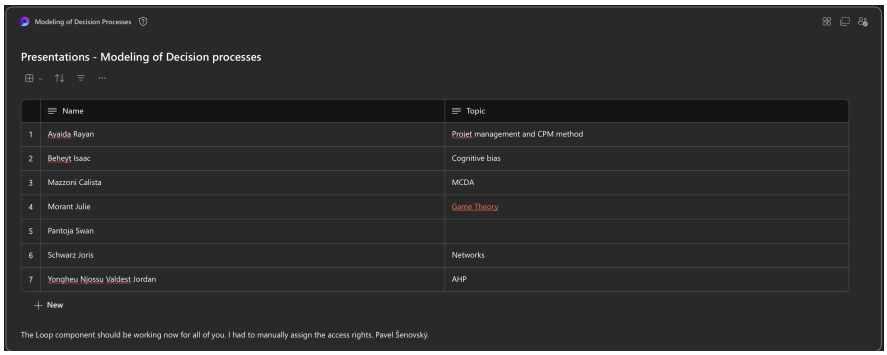
Odkaz na hlasovací formulář pak bude rozeslán účastníků, kteří do něj budou moci vložit své časové preference. Celý proces dokonce lze nastavit tak, že v okamžiku, kdy se účastníci na nějakém termínu shodnou, rovnou se ta událost na tento termín naplánuje. Lze ale ponechat také na ruční volbě organizátora schůzky.



Obrázek 11.6: Průvodce plánováním schůzky



Obrázek 11.7: Hlasování o termínu schůzky



Obrázek 11.8: Loop komponenta - tabulka

11.4 MS Loop

Posledním nástrojem, ale svým způsobem nejzajímavějším je MS Loop. Tato služba pracuje s tzv. Loop komponentami. Myšlenka pro použití je jednoduchá - pořád rozesíláme maily, abychom získali zpět nějakou informaci. To znamená obvykle, že napíšeme mail rozešleme příjemcům a doufáme, že nám odpoví. Až tak učiní, budeme integrovat jejich odpovědi do jednoho materiálu.

V minulé kapitole jsme probírali OneDrive. Lepším řešením by bylo nasdílet dokument a shromažďovat informace do něj. Toto je docela dobré řešení, zejména pokud se jedná o nějaký sofistikovanější dokument a vyžadujeme extenzivní diskuzi o různých problémech uvnitř tohoto dokumentu. Pokud ale chceme pouze rychle vyplnit nějakou tabulku nebo zapracovat na pár odstavcích textu nepůsobí řešení pomocí nasdíleného dokumentu elegantně ... je těžkopádné.

Právě pro tyto účely Microsoft navrhl Loop. Vytvoříte komponentu - nastavíte práva a rozešlete mailem, MS Teams, apod. Příjemci pak do komponenty rovnou vepisují. Není potřeba tedy odpovídat. Všichni příjemci vidí aktuální stav komponenty včetně změn realizovaných dalšími příjemci.

Např. níže (viz obr. 11.8) je má komponenta s volbou témat prezentací ERASMUS+ studentů v tomto semestru.

Vytvořte Loop komponentu:
V mailu v Outlook vytvořte a rozešlete Loop komponentu. Může se jednat o tabulku, odstavec nebo cokoliv jiného a rozešlete ji. Sledujte funkci Loop, jak postupně je komponenta vyplňována.

Můžete také použít webové rozhraní služby pro práci a manipulaci s komponentami na https://loop.microsoft.com.

Kapitola 12

Nástroje on-line komunikace

Náhled kapitoly
V poslední kapitole se ponoříme do komunikačních platforem jako je MS Teams. Tato kapitola bude ve skutečnosti poměrně krátká, protože v posledních dvou kapitolách jsme probrali řadu nástrojů, které se využívají také v MS Teams.

Vlastně budeme postupovat trochu jinak - pracovat budete Vy!

Vytvořte tým
Vytvořte nový tým v MS Teams a nastavte tam několik svých přátel.

- nastavte některé stránky teamu
- nechte si vygenerovat týmový SharePoint (a zkuste si jej nastavit)
- doplňte kalendář
- a úkoly

uspořádejte schůzku
Nastavte pozadí tak, aby bylo anonymní. Otestujte způsob, jakým se připojuje kamera a mikrofon. Pokud to Váš systém podporuje (např. Mac) použijte při připojení do jednání kameru a mikrofon ze svého telefonu, mají výrazně lepší kvalitu záznamu než Váš notebook (obvykle).

Pozvěte do schůzky některé spolužáky a testujte vlastnosti Teamsů - umíte se ztlumit? (A co je horší, umíte si opětovně zapnout obraz a zvuk?)

Sdílení obrazovky
Ve schůzce si nasdílejte obrazovku:

- vyzkoušejte sdílení celé obrazovky - považujete to za bezpečné?
- porovnejte s možností nasdílení jediného okna
- nechte někoho jiného prezentovat sdílení ve Vaší schůzce - udělte práva k prezentování

Chat
zapněte textový chat a vyzkoušejte jej. Zkuste nasdílet soubor, popř. odkaz,

Integrace
prozkoumejte, jak se MS Teams integruje s ostatními službami Microsoft 365. (OneDrive, SharePoint, Kalendáře, Loop). Jak byste je využili Vy?

Literatura

[1] Microsoft project. URL: http://office.microsoft.com/cs-cz/project/.

[2] Open workbench. URL: http://sourceforge.net/projects/openworkbench/.

[3] Project libre. URL: http://www.projectlibre.org/.

[4] Analytic hierarchy process – car example, 2010. URL: https://en.wikipedia.org/wiki/Analytic_hierarchy_process_%E2%80%93_car_example.

[5] PRISMA: TRANSPAREN REPORTING of SYSTEMATIC REVIEWS and META-ANALYSES, 2020. URL: https://www.prisma-statement.org.

[6] Hugging Face – The AI community building the future., March 2026. URL: https://huggingface.co/.

[7] Jsme připraveni? | 72hodin, 2026. URL: https://www.72h.gov.cz/public/styles/meta_img/azblob/meta/images/hp-cs.png?itok=HxLUJjWg.

[8] LM Studio - Local AI on your computer, 2026. URL: https://lmstudio.ai.

[9] R: The R Project for Statistical Computing, 2026. URL: https://www.r-project.org/.

[10] Antropic. Agent Skills, 2026. URL: https://agentskills.io/home.

[11] Artificial Analysis. AI Model & API Providers Analysis, 2026. URL: https://artificialanalysis.ai.

[12] Niclas Aunin. A comprehensive list of Large Language Model knowledge cut off dates - ALLMO: Boost Your Brand's Visibility in AI Search, 2026. URL: https://www.allmo.ai/articles/list-of-large-language-model-cut-off-dates.

[13] CDF. SuperDecisions. URL: http://sdbeta.superdecisions.com/.

[14] Aditya Challapally, Chris Pease, Ramesh Raskar, and Pradyumna Chari. *The GenAI Divide STATE OF AI IN BUSINESS 2025*. MIT, 2025. URL: https://mlq.ai/media/quarterly_decks/v0.1_State_of_AI_in_Business_2025_Report.pdf.

[15] Amy Chang and Vineeth Sai Narajala. Personal AI Agents like OpenClaw Are a Security Nightmare, January 2026. URL: https://blogs.cisco.com/ai/personal-ai-agents-like-openclaw-are-a-security-nightmare.

[16] Clarivate. Web of Science, 2026. URL: http://webofknowledge.com.

[17] D. Diakoulaki, G. Mavrotas, and L. Papayannakis. Determining objective weights in multiple criteria problems: The critic method. *Computers & Operations Research*, 22(7):763–770, August 1995. URL: https://www.sciencedirect.com/science/article/pii/030505489400059H, doi:10.1016/0305-0548(94)00059-H.

[18] Elseiver. Scopus, 2026. URL: https://www.scopus.com.

[19] Miloš Gligorić, Zoran Gligorić, Suzana Lutovac, Milanka Negovanović, and Zlatko Langović. Novel Hybrid MPSI–MARA Decision-Making Model for Support System Selection in an Underground Mine. *Systems*, 10(6), December 2022. URL: https://www.mdpi.com/2079-8954/10/6/248, doi:10.3390/systems10060248.

[20] Google. Gemini, 2026. Online; accessed 10-March-2026. URL: https://gemini.google.com/.

[21] Google. Google NotebookLM: AI nástroj pro výzkum a partner pro přemýšlení, 2026. URL: https://notebooklm.google/.

[22] Google. Google Scholar, 2026. URL: http://scholar.google.com/.

[23] Gostev, Peter. BullshitBench: V2, 2026. URL: https://github.com/petergpt/bullshit-benchmark.

[24] Don Ho. Notepad++, 2026. URL: https://notepad-plus-plus.org/.

[25] KDE. Kate - Dostaňte se na hranu při editování, 2026. URL: https://kate-editor.org/cs/.

[26] Mehdi Keshavarz-Ghorabaee, Maghsoud Amiri, Edmundas Kazimieras Zavadskas, Zenonas Turskis, and Jurgita Antucheviciene. Determination of Objective Weights Using a New Method Based on the Removal Effects of Criteria (MEREC). *Symmetry*, 13(4), March 2021. URL: https://www.mdpi.com/2073-8994/13/4/525, doi:10.3390/sym13040525.

[27] Knihovna ČVUT. Co je rešerše, 2026. URL: https://knihovna.cvut.cz/seminare-a-vyuka/jak-psat/co-je-reserse.

[28] Antonín Krömer, Petr Musial, and Libor Folwarczny. *Mapování rizik*. SPBI, Ostrava, 2010.

[29] Raman Kumar, Paramjit Singh Bilga, and Sehijpal Singh. Multi objective optimization using different methods of assigning weights to energy consumption responses, surface roughness and material removal rate during rough turning operation. *Journal of Cleaner Production*, 164:45–57, October 2017. URL: https://www.sciencedirect.com/science/article/pii/S0959652617312477, doi:10.1016/j.jclepro.2017.06.077.

[30] Microsoft. Copilot, 2026. Online; accessed 10-March-2026. URL: https://copilot.microsoft.com/.

[31] OECD. *Handbook on constructing composite indicators: methodology and user guide*. OECD, Paříž, 2008.

[32] Open AI. ChatGPT, 2026. Online; accessed 10-March-2026. URL: https://chatgpt.com/cs-CZ/.

[33] OpenAI. Představujeme ChatGPT Atlas, March 2026. URL: https://openai.com/cs-CZ/index/introducing-chatgpt-atlas/.

[34] Matthew J Page, Joanne E McKenzie, Patrick M Bossuyt, Isabelle Boutron, Tammy C Hoffmann, Cynthia D Mulrow, Larissa Shamseer, Jennifer M Tetzlaff, Elie A Akl, Sue E Brennan, Roger Chou, Julie Glanville, Jeremy M Grimshaw, Asbjørn Hróbjartsson, Manoj M Lalu, Tianjing Li, Elizabeth W Loder, Evan Mayo-Wilson, Steve McDonald, Luke A McGuinness, Lesley A Stewart, James Thomas, Andrea C Tricco, Vivian A Welch, Penny Whiting, and David Moher. The PRISMA 2020 statement: an updated guideline for reporting systematic reviews. *BMJ*, (372), 2021. URL: https://www.bmj.com/lookup/doi/10.1136/bmj.n71, doi:10.1136/bmj.n71.

[35] Posit. RStudio, 2026. URL: https://posit.co/download/rstudio-desktop/.

[36] David Rehak, Pavel Senovsky, Martin Hromada, and Tomas Lovecek. Complex approach to assessing resilience of critical infrastructure elements. *International Journal of Critical Infrastructure Protection*, 25:125–138, June 2019. URL: http://www.sciencedirect.com/science/article/pii/S1874548218301744, doi:10.1016/j.ijcip.2019.03.003.

[37] Thomas L. Saaty. Decision making with the analytic hierarchy process. *International Journal of Services Sciences*, 1(1):83–98, 2008.

[38] Scite. AI for Research | Scite, 2026. URL: https://scite.ai.

[39] Scite. Scite Badge Documentation, 2026. URL: https://scite.ai.

[40] Scite. Scite browser extension, 2026. URL: https://scite.ai.

[41] Peter Steinberger. OpenClaw — Personal AI Assistant, 2026. URL: https://openclaw.ai/.

[42] Edmundas Kazimieras Zavadskas and Valentinas Podvezko. Integrated Determination of Objective Criteria Weights in MCDM. *International Journal of Information Technology & Decision Making*, 15(2):267–283, March 2016. URL: https://www.worldscientific.com/doi/abs/10.1142/S0219622016500036, doi:10.1142/S0219622016500036.

[43] Pavel Šenovský. AHP YAML Editor, 2017. URL: https://fbiweb.vsb.cz/~sen76/data/uploads/programy/AHPEditor%20v0.1.7z.

[44] Pavel Šenovský. MCDASupport Package 0.36, 2026. URL: https://github.com/psenovsky/MCDASupport.

[45] Pavel Šenovský. *MCDASupport Package Documentation version 0.36.* Ostrava, 2026. URL: https://github.com/psenovsky/MCDASupport/releases/download/v0.36/MCDASupport_0.36.pdf.

Slovník

AHP Analytic Hierarchy Process.

ANP Analytic Network Process.

CILAS Criterion Impact LOSs.

CMS Content Management System.

CPM Critical Path Method.

CRITIC Criterion Importance Through Intercriteria Correlation.

EWM Entropy Weight Method.

GCW Gini Coefficient Weighting.

GUI Graphical User Interface.

IDOCRIW Integrated Determination of Objective CRIteria Weights.

JDK Java Developlent Kit.

LLM Large Language Model (velký jazykový model.

MCP Model Context Protocol.

MEREC Method based on Removal Effects of Criteria.

MPSI M Preference Selection Index.

MW Mean weighting.

NER Named Object Recognition.

RAG Retrieval Augmented Generation.

SDW Standard Deviation Weighting.

SSO Single Sigh On.

SVD Singular Variable Decomposition.

SVW Statistical Variance Weighting.

VPN Virtual Private Network.

WoS Web of Science.

WSM Weighted Sum Method.

YAML YAML Ain't Markup Language.

[title=Seznam zkratek]