

Žilinská univerzita v Žiline
Spoločnosť pre otvorené informačné technológie
Geokomunita, združenie CSTUG
Česká statistická společnost, INO-HUB Energy

**OTVORENÝ SOFTVÉR VO VZDELÁVANÍ,
VÝSKUME A V IT RIEŠENIACH**



**Zborník príspevkov medzinárodnej konferencie
OSSConf 2025**

1. júla – 3. júla 2025
Žilina, Slovensko

Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach

1. júla – 3. júla 2025, Žilina, Slovensko

Vedecký garant konferencie:

doc. Ing. Michal Koháni, PhD., prorektor pre informačné systémy, Žilinská univerzita, Žilina

Vedecký a programový výbor:

doc. Ing. Miroslav Kvaššay, PhD., UNIZA Žilina – **predseda**

doc. Ing. Peter Márton, PhD., UNIZA Žilina

doc. Ing. Jiří Rybička, Dr., MENDELU, Brno (CZ)

doc. Ing. Zdena Dobešová, Ph.D., Univerzita Palackého, Olomouc (CZ)

doc. Mgr. Ondřej Vencálek, Ph.D., PŘF UP Olomouc, Česká statistická společnost (CZ)

RNDr. Aleš Kozubík, PhD., UNIZA SOIT, Žilina

RNDr. Rudolf Blaško, PhD., UNIZA SOIT, Žilina

Ing. Tomáš Majer, PhD., UNIZA SOIT, Žilina

Mgr. Martin Šechný, SOIT

Dr inż. Radosław Maciaszczyk, Zachodniopomorski Uniwersytet Technologiczny w Szczecinie (PL)

Dr inż. Mirosław Łazoryszczak, Zachodniopomorski Uniwersytet Technologiczny w Szczecinie (PL)

Ing. Jan Růžička, Ph.D., GISMentors, Ostrava (CZ)

Ing. Pavel Stříž, Ph.D., Nakladatelství Martin Stříž, Bučovice (CZ)

Organizačný výbor:

Miroslav Kvaššay, UNIZA Žilina – **predseda**

Aleš Kozubík, UNIZA SOIT, Žilina

Rudolf Blaško, UNIZA SOIT, Žilina

Tomáš Majer, UNIZA SOIT, Žilina

Pavel Stříž, Nakladatelství Martin Stříž, Bučovice

Peter Márton, UNIZA Žilina

Vydavateľ: Žilinská univerzita v Žiline

ISBN 978-80-554-2209-1

Copyright © 2025 autori príspevkov

Ktokoľvek má dovoľenie vyhotoviť alebo distribuovať doslovný opis tohoto dokumentu alebo jeho časti akýmkoľvek médium za predpokladu, že bude zachované oznámenie o copyrighte a o tom, že distribútor príjemcovi poskytuje povolenie na ďalšie šírenie, a to v rovnakej podobe, akú má toto oznámenie.

Žilinská univerzita v Žiline
Spoločnosť pre otvorené informačné technológie
Geokomunita, združenie CSTUG
Česká statistická společnost, INO-HUB Energy

**OTVORENÝ SOFTVÉR VO VZDELÁVANÍ,
VÝSKUME A V IT RIEŠENIACH**



**Zborník príspevkov medzinárodnej konferencie
OSSConf 2025**

1. júla – 3. júla 2025
Žilina, Slovensko

Recenzenti:

Blaško, Rudolf, RNDr., PhD., Borčinová, Zuzana, RNDr., PhD., Kaukič, Michal, Mgr., CSc., Kozubík, Aleš, RNDr., PhD., Kvaššay Miroslav, doc. Ing., PhD., Márton Peter, doc. Ing., PhD., Koháni Michal, doc. Ing., PhD., Peško, Štefan, doc., RNDr., PhD., Rybička Jiří, doc., Ing., Dr., Stříž, Pavel, Ing., Ph.D., Radosław Maciaszczyk, Dr inż., Mirosław Łazoryszczak, Dr inż.

Editori:

Rudolf Blaško, Pavel Stříž

Všetky práce, uverejnené v zborníku, boli posúdené dvomi nezávislými recenzentmi.

Za jazykovú úroveň zodpovedajú autori príspevkov.

Obsah

Úvod	5
Zuzana Tkáčová	
Chatgpt ako postrach učiteľov programovania (!) (vyžiadaná prednáška) ...	7
Richard Fabo	
Pod kapotou „OpenPLC“ (vyžiadaná prednáška)	8
Eliška Cézová	
Výuka matematiky pomocí umělé inteligence (vyžiadaná prednáška)	10
Michal Páleník	
Tvorba máp pomocou R (vyžiadaná prednáška)	11
Spóťová Monika, Klimo Marek	
The BBC micro:bit in Digi Camp: how an Open Source tool forms technical thinking among Slovak pupils (vyžiadaná prednáška)	12
Michal Hoftich	
Publikace L ^A T _E Xových dokumentů na web pomocí T _E X4ht a Github Actions (vyžiadaná prednáška)	14
Vašek Chvátal	
Životní pouť v optimalizaci	15
Miroslav Bendík	
Smart internetové rádio na platforme ESP32	19
Rudolf Blaško	
Far Manager – správca súborov a archívov	25
Rudolf Blaško	
Moodle a matematika	35
Rudolf Blaško	
TikZ a zopár nápadov	41
Eliška Cézová	
Výuka matematiky pomocí umělé inteligence	53
Marek Ďurana, Miroslav Kvaššay	
HTCondor Computational Cluster in academic environment	61
Richard Fabo	
Eshell: skutočne prívetivý emulátor terminálu	69

Richard Fabo	
Pod kapotou „OpenPLC“	79
Aleš Kozubík	
Je Python priateľom L ^A T _E X-u?	87
Ondřej Ledvinka	
R balíčky určené k práci s geodaty a jejich využití v projektu Perun	95
Štefan Peško, Zuzana Borčinová	
Pokus o optimálne riešenie úlohy obchodného cestujúceho s dronom	103
Jiří Rybička, Pavel Haluza, Lucie Pacáková	
Výzkum dokumentové gramotnosti: stav a perspektivy	113
Pavel Stříž	
O jedné mapě na zakázku, část 1: Teorém 4 barev prakticky	123
Pavel Stříž	
Míchání rastrové a vektorové grafiky v typografii	131
Pavel Stříž	
Optimalizujeme s programem Concorde	139
Pavel Stříž	
Geocaching.com slaví 25 let: GCB78KB, GCB78MA, GC8RXDE	147
Pavel Stříž, Štefan Peško	
Maximum bloků v klasickém sudoku	153
Pavel Stříž	
Historie T _E Xu a klíčových nástrojů na grafiku	159
Ondřej Vencálek, Pavel Stříž	
Mapa míst konferencí ROBUST v R	163
Pavel Stříž	
Recenze knihy Opt Art od Roberta Bosche	171
Abstrakty nerecenzovaných ukážok, prezentácií a workshopov	175

Vážení kolegovia, priatelia a priaznivci otvorených technológií,

som rád, že si k vám našiel cestu tento zborník, ktorý je výstupom odbornej časti konferencie OSSConf 2025. Po úspešnom obnovení konferencie v roku 2024, ktorej prvý ročník prebehol v roku 2009, pokračujeme tentokrát v 13. ročníku tohto podujatia, ktorého cieľom je vytvoriť platformu na zdieľanie poznatkov z oblastí otvoreného softvéru, hardvéru a údajov. Pre niektorých môže číslo 13 symbolizovať nešťastie a smolu, no v kontexte OSSConf obvyčajne indikuje príchod novej krvi.



V 13. roku druhého tisícročia prebral po zakladateľovi konferencie, Michalovi Kaukičovi, štafetu v jej organizovaní Aleš Kozubík, ktorý ju dlhodobo viedol na vysokej a priateľskej úrovni. V 13. ročníku dochádza opäť k podobnej zmene. V porovnaní s inými zmenami vo svete je však táto zmena výsledkom postupnej evolúcie a nie revolúcie. Michalovi a Alešovi by som chcel veľmi pekne poďakovať za založenie a udržiavanie tejto konferencie ako aj za ich pomoc s organizáciou 13. ročníka.

Tohtoročný program je zaujímavý nielen svojou rozsiahlosťou, ale aj výnimočnosťou a vyváženosťou vo viacerých ohľadoch. Z pohľadu výnimočnosti by som z množstva zaujímavých príspevkov upozornil na rozhovor s prof. Vaškom Chvátalom, Ph.D., z Matematicko-fyzikálnej fakulty Univerzity Karlovy v Prahe, objaviteľom tzv. Chvátalovho grafu. Pán profesor je známy množstvom vedeckých a popularizačných publikácií z oblasti kombinatoriky, teórie grafov a matematického programovania. Na príprave niekoľkých článkov spolupracoval s jedným z najvýraznejších matematikov 20. storočia, ktorým bol Paul Erdős.

Z pohľadu vyváženosti máme 6 odborných sekcií a 6 vyžiadaných prednášok. Zo 6 vyžiadaných prednášok sú 3 od dámskeho osadenstva a 3 od pánskeho. Z množstva zaujímavých prednášateľov by som rád spomenul aspoň jedno meno, ktorým je Zuzana Tkáčová. Zuzka je učiteľkou informatiky a programovania na strednej škole a ako prvá na Slovensku získala ocenenie Učiteľ Slovenska. V súčasnosti vedie tím pre AI vo vzdelávaní v Národnom centre pre digitálnu transformáciu vo vzdelávaní.

V rámci 6 odborných sekcií je prvou už tradičná sekcia „Open GIS a Open Data“ pod vedením Mila Ofúkaného a Petra Mártona. Aj keď sa Milo tento rok nemôže zúčastniť prezenčne, chcel by som jemu a Peťovi veľmi pekne poďakovať za prípravu tejto sekcie a Milovi aj za jeho aktívnu účasť na celkovom organizovaní konferencie a napĺňaní webu konferencie pravidelnými novinkami.

Ďalšou tradičnou sekciou je sekcia „L^AT_EX a jeho priatelia“. Dlhoročnými garantmi tejto sekcie sú Aleš Kozubík, Rudolf Blaško a Pavel Stríž, vďaka ktorých vytrvalosti táto sekcia každý rok rastie nielen kvantitatívne, ale aj kvalitatívne. Ich vytrvalý prístup sa minulý rok premietol do rozšírenia názvu sekcie aj o písmeno „R“. Alešovi a Pavlovi by som sa tiež veľmi rád poďakoval za prípravu programu konferencie a akčný prístup pri propagácii konferencie prostredníctvom webu a viacerých komunikačných kanálov, ktoré majú svoje ovocie v podobe pozvaných prednášok a účastníkov z radov učiteľov pôsobiach na základných a stredných školách.

Rudolfovi Blaškovi a Mirekovi Biňasovi ďakujem za prípravu sekcií „OSS vo vzdelávaní“ a „Open Hardware“, v rámci ktorých máme viaceré zaujímavé príspevky na tému vzdelávania s využitím otvorených technológií. Tieto sekcie sú doplnené sekciou zameranou na „Vývoj OSS“. V súlade s obsahovou náplňou týchto troch sekcií sú pripravené aj 3 zaujímavé workshopy. V prvom sa pod vedením Zuzany Tkáčovej účastníci zoznámia s programovaním v p5.js s využitím umelej inteligencie. Druhý workshop, nazvaný „Pico na citrón“, predstaví praktické využitie mikrokontroléra Raspberry Pi Pico a dátových vizualizácií pomocou MicroPythonu a MQTT. Tretí workshop, zameraný na vývoj jednoduchého Pokédexu pomocou technológií FastAPI a SQLite, ukáže, že aj vzdelávacie projekty môžu byť zábavné a technicky podnetné.

Poslednou sekciou je „Open AI“. Vo svete sa umelá inteligencia čoraz viac uplatňuje vo vzdelávaní ako nástroj na personalizáciu výučby a prispôbovanie obsahu individuálnym potrebám študentov. Využitie jazykových modelov, akými sú ChatGPT, Claude alebo DeepSeek, otvára nové možnosti pre tvorbu vzdelávacích materiálov, spätnú väzbu a rozvoj analytického myslenia. Tieto skutočnosti sú reflektované aj na Slovensku, kde prebieha aktualizácia Štátneho vzdelávacieho programu aj v oblasti informatiky s ohľadom na nové trendy.

Popri odbornej časti nebude chýbať na konferencii ani umelecký duch. Tentokrát budú v priestoroch konferencie ukážky mrakov slov („wordcloud“), ktoré slúžia na vizuálnu reprezentáciu textu, v ktorej veľkosť slov zodpovedá ich frekvencii výskytu. Tieto sa používajú na rýchlu analýzu obsahu, keď chceme zistiť, aké pojmy alebo témy dominujú v texte. Ukážky pripravené pre výstavu sú pripravené s využitím TikZ balíka pre L^AT_EX. Na oficiálnej stránke konferencie je tiež možné nájsť kalendáre vytvorené Karlom Horákom, ktoré sú príkladom elegantného spojenia geometrie, matematiky a krásnej sadzby.

V neposlednom rade nemôžeme opomenúť našich dobrodincov a partnerov, vďaka ktorým môžeme zorganizovať 13. ročník konferencie OSSConf. Medzi týchto patrí vedenie Fakulty radenia a informatiky Žilinskej univerzity v Žiline, zastúpené osobou pána dekana prof. Ing. Emila Kršáka, PhD., ktoré dlhoročne podporuje našu konferenciu bezodplatným poskytnutím konferenčných miestností a laboratórií v priestoroch fakulty. Taktiež by som sa chcel poďakovať spoločnosti INO-HUB Energy j.s.a. a občianskemu združeniu ALUMNI FRI za ich podporu s finančným a organizačným zabezpečením konferencie. K organizácii konferencie ako aj k príprave a vydaniu tohto zborníka pomohol taktiež grant KEGA 019ŽU-4/2023 Inovatívne učenie matematiky s podporou Open Source podporený Slovenskou kultúrno-edukačnou grantovou agentúrou.

Na záver chcem popriať všetkým účastníkom príjemnú konferenciu, inšpiratívne diskusie a veľa dobrých nápadov.

Za organizačný výbor OSSConf 2025

Miroslav Kvaššay
predseda

ChatGPT AKO POSTRACH UČITELOV PROGRAMOVANIA (?!)

Zuzana TKÁČOVÁ (SK)

VYŽIADANÁ PREDNÁŠKA

Abstrakt: V ére, keď študenti dokážu získať funkčný kód jedným promptom, sa mnohí učitelia programovania pýtajú: „Má ešte zmysel učiť syntax a algoritmy?“ Táto otázka dnes neznie ako provokácia, ale ako reálna výzva, pred ktorou stojí každý, kto učí informatiku. Táto prednáška ponúka pohľad na to, ako sa ChatGPT stal nielen postrachom a nočnou morou, ale aj výzvou a príležitosťou pre moderné vyučovanie programovania. Učiteľ sa zrazu ocitá v úplne novej role: už nie je len nositeľom vedomostí, ale sprievodcom v prostredí, kde odpoveď často prichádza skôr, než padne otázka.

Diskutovať budeme o konkrétnych stratégiách, ako využiť ChatGPT na zlepšenie výučby programovania a pripraviť študentov na svet, kde AI je bežnou súčasťou ich pracovného prostredia. Zároveň predstavíme rôzne prístupy a nové trendy v oblasti novovznikajúcej AI didaktiky a o tom, ako v tom môže pomôcť ChatGPT.

Prednáška je určená všetkým (nadšencom aj nešťastníkom), ktorí chcú lepšie porozumieť tomu, ako sa mení výučba (nielen) programovania pod vplyvom AI – a ako si v tom celom nájsť miesto pre kvalitné a zmysluplné učenie.

Príspevok vznikol v rámci riešenia grantu KEGA 010UPJŠ-4/2024 „Využitie umelej inteligencie vo vyučovaní školskej informatiky na stredných školách“.



Zuzana Tkáčová. Som (zatiaľ ešte stále) nadšená učiteľka informatiky a programovania na strednej škole. Vyštudovala som AI a v súčasnosti pôsobím v tejto oblasti ako lektorka, metodička, školiteľka učiteľov, publikujem a vediem tím pre AI vo vzdelávaní v Národnom centre pre digitálnu transformáciu vo vzdelávaní.

Moje superschopnosti: Creative computing & Physical computing.

POD KAPOTOU „OpenPLC“

Richard FABO (SK)

VYŽIADANÁ PREDNÁŠKA

Abstrakt: Programovateľné logické kontroléry sú v priemyselných aplikáciách používané takmer polstoročie, s výraznou dominanciou komerčných výrobcov. Projekt Arduino vznikol v roku 2005 ako nástroj na uľahčenie interaktívneho dizajnu a prototypovania. Sprístupnil mikroelektroniku a jej programovanie aj ľuďom, ktorí by inak nemali záujem prekonávať pomerne strmú krivku učenia potrebnú na zvládnutie oboch zručností (elektronika, programovanie). Okamžite začali nadšenci uvažovať o nahradení komerčných PLC mikrokontrolérmi, ktoré by sa dali programovať práve pomocou, ale nielen Arduino IDE, s využitím veľkého množstva knižníc a podpory silnejúcej komunity. Fascinovaní filozofiou slobodného a otvoreného hardvérového dizajnu v kombinácii s analogickým prístupom k programovacím nástrojom, knižniciam či dokumentácii. A to je hlavnou motiváciou, prečo dizajnovať, vyrábať a používať slobodné PLC zariadenia. A to sa aj podarilo, v súčasnosti existuje množstvo zavedených výrobcov, ktorých výrobky sú vhodné do neprívětivého zarušeného prostredia priemyselných strojov a liniek. A nie je to už len o známych procesoroch od Atmelu, ktoré preslávili Arduino najviac, časom pribudli procesory a vývojové dosky iných výrobcov.

Prednáška zmieni fungovanie programu OpenPLC. Je to integrované vývojové prostredie na programovanie PLC kontrolérov vo viacerých grafických i textových jazykoch v súčinnosti so zaužívanou normou. Jedná sa o slobodné a otvorené riešenie, ktoré je dostupné pre každého, od študentov po profesionálov. Má vzdelávací potenciál, vďaka otvorenej koncepcii je flexibilný a rozšíriteľný. Spolu s inými projektmi prispieva k otvorenej a nezávislej priemyselnej automatizácii.

Väčší dôraz, ako na používanie programu, budeme klásť na spôsob jeho fungovania: Čo všetko sa musí udiť, aby sa z (nielen) grafickej implementácie programu pre kontrolér stal výsledný binárny súbor pre ten-ktorý procesor. A to všetku v duchu „unixovej filozofie“.



Richard Fabo. „Sloboda je to, na čom skutočne záleží“, povedal som si a cca v roku 2005 sme prešli v našej malej spoločnosti (dnes Triton Famme s.r.o.) na slobodné operačné systémy. Postupne sme prekonávali prekážky typické pre toto rané GNU/Linuxové obdobie: od chýbajúcich ovládačov pre modemy pre mobilnú sieť, cez nefungujúci internetbanking, nedostatok dokumentácie či skúseností okolia, až po migráciu dokumentov. Nakoniec to stálo (a stále stojí) za to! Nebola to „hra s nulovým súčtom“ – strávený čas sa nám vrátil mnohonásobne – získané vedomosti, spoz-

nanie nových ľudí, prispôsobený softvér našim požiadavkám a predstavám, prenikanie do tajov a prístupu k slobodnému softvéru.

A fascinácia slobodným a otvoreným softvérom pretrváva. Sám sa najlepšie cítim v hrejivom objatí softvérovej implementácii lisp-ového stroja s názvom GNU Emacs, považujúc ho za najvyšší stupeň digitálnej existencie :-).

Naša spoločnosť sa orientuje na návrhy, realizácie, prípadne rekonštrukcie technologických liniek na pozberové a potravinárske spracovanie zemiakov a zeleniny, spolu so skladovými technológiami na Slovensku i v zahraničí. Od dizajnovania celých liniek, jej súčastí, spôsobu ich riadenia. Niektoré stroje a ich ovládanie si navrhujeme sami, s inými nám pomáhajú spriatelené firmy, ďalšie kupujeme od iných výrobcov, pričom realizujeme spôsob ich technologického a technického (včítane komunikačného) prepojenia. Riadenie zverujeme otvoreným kontrolérom – dospelým, testovaným, spoľahlivým zariadeniam spĺňajúcich požiadavky nasadenia do priemyselného prostredia.

Hobby: Ak nerátame záľuby viac-menej prepojené s IT, tak občasná horská cyklistika a sledovanie motoršportu (rôzne pretekárske i menej známe, série).

VÝUKA MATEMATIKY POMOCÍ UMĚLÉ INTELIGENCE

Eliška CÉZOVÁ (CZ)

VYŽIADANÁ PREDNÁŠKA

Abstrakt: Rozvoj umělé inteligence (AI) přináší nové příležitosti i výzvy v oblasti vzdělávání, zejména ve výuce matematiky. Tento příspěvek analyzuje současné možnosti využití AI nástrojů k podpoře výukového procesu, a to jak z pohledu učitele, tak žáka. Zaměřuje se na adaptivní výukové systémy, inteligentní doučovací platformy, automatizované hodnocení a personalizované vzdělávací přístupy. Cílem je nabídnout vyvážený pohled na potenciál umělé inteligence jako nástroje pro modernizaci a zkvalitnění výuky matematiky.



Eliška Cézová. Působím na Masarykově ústavu vyšších studií a Fakultě strojní ČVUT v Praze, na Ústavu konstruování a části strojů. Především vyučuji matematickou analýzu. Jsem zapálená kantorka a hledám cesty ke zkvalitnění výuky. V roce 2022 mi vyšla Sbírka příkladů z neurčitého integrálu, kterou před pár dny rozšířil druhý díl Sbírka příkladů z určitého integrálu. Mám z toho neskutečnou radost!

Pravidelně navštěvuji statistické konference jako jsou Statistické dny, STAKAN a ROBUST.

Jsem členkou České statistické společnosti. Ráda $\TeX$ uji a zkoumám limity ChatGPT na všechny způsoby. Miluji zvířátka, přírodu a výlety. Na OSSConf budu historicky poprvé, těším se za Vámi, přátelé!

TVORBA MÁP POMOCO U R

Michal PÁLENÍK (SK)

VYŽIADANÁ PREDNÁŠKA

Abstrakt: Mapy sú významný prostriedok vizualizácie geografických údajov, dokážu zobrazit komplexné sady dát. Avšak aj tvorba jednoduchých máp zobrazujúcich jednu hodnotu v okrese či obci má viacero úskalí. V prednáške si ukážeme, ako v prostredí Open Source štatistického programu R vytvoriť základnú mapu s výstupom vo forme obrázku (png, svg, pdf).

Ako dáta použijeme LAU1 dataset (DOI:10.5281/zenodo.6165135) o nezamestnanosti v okresoch krajín Vyšehradskej štvorky. Na pôdoryse okresov budeme zobrazovať mieru nezamestnanosti či index demografickej závislosti, teda kontinuálnu numerickú premennú.

Príspevok je súčasťou grantu KEGA 076UK-4/2025 „CORE Econ z perspektívy strednej Európy“.



Michal Páleník. Posledné desiatky rokov používam otvorený softvér na všetko čo sa dá, vďaka čomu som sa dostal do rôznych tajov a zákutí: od správy PostgreSQL servera a optimalizáciu jeho behu, cez štatistickú analýzu dát až po vizualizáciu mapových dát.

Robím na stránke www.oma.sk na Univerzite Komenského či v Inštitúte zamestnanosti. Najneužitečnejšia vec čo som urobil je interaktívna mapa Slovenska v klingončine.

THE BBC MICRO:BIT IN DIGI CAMP: HOW AN OPEN SOURCE TOOL FORMS TECHNICAL THINKING AMONG SLOVAK PUPILS

SPÓTOVÁ Monika (SK), KLIMO Marek (SK)

VYŽIADANÁ PREDNÁŠKA

Abstrakt: Digi Camp (Innovation Camp of Digital Literacy) was a nationwide competition aimed at pupils of elementary and grammar schools. The objective was to introduce pupils to modern technologies via practical activities and technical challenges using BBC micro:bit as an Open Source tool. This project was organized by Junior Achievement Slovakia (<https://www.jaslovensko.sk>).

The event was divided into two days, with the first day being practical and study-oriented, and the second day being a competition between the teams. The event occurred on 11th and 12th of March 2025. A total of 20 teams participated, with each team having four pupils. Every team consists of a pupil from differing school, so that no one team has two pupils from the same place to promote teamwork and socializing. A pair of teams were given an instructor to advise on problem-solving and evaluating. Instructors were university and PhD students from the University of Žilina.

BBC micro:bit (<https://microbit.org/>) is an Open Source tool developed by the BBC. It gained popularity among elementary and secondary Slovak schools teaching Informatics and also being used in extracurricular activities. First introduced in 2014, gaining popularity in the United Kingdom, eventually made its impact worldwide, reaching Slovak education.

The first day of the event was oriented around 9 challenges, with the first being remote control of a BBC micro:bit. This challenge serves as a foundation for later challenges. Another challenge was creating a remotely controlled motorcycle. Each team was given required materials, step-by-step instructions to build and code the functionality.

The second day was competitive, having teams compete against each other. The first part of the competition were 11 challenges, each being evaluated on construction, functionality and controllability. Teams were given a set amount of time to complete as many challenges as possible within the time limit. Some of these challenges were: automatic doors, reading range guitar, spider walker, helicopter, locomotive, ambient lighting, and other. The second part of the competition was to create a functional car, which could traverse racecourse with

obstacles. Here teams were evaluated based on reaching the obstacles, avoiding the obstacles, not touching the obstacles, and finally reaching the finish line. The average time of reaching the finish line was 20.6 seconds.

Feedback was positive, both from pupils and teachers. Teachers viewed the use of instructors and organization as very impactful on pupils learning the basics of programming. Teachers also mentioned mixed teams as a great basis for team-building of pupils, impacting their creativity and enjoyment of the event. Pupils dignified teamwork and learning basics of programming in a short time paired with fun challenges. Teachers and pupils alike shared their view to have similar events in the future.

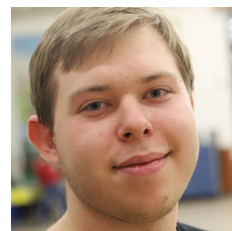
The main benefit of this competition was the development of technical thinking and mutual cooperation between pupils, using the Open Source BBC micro:bit as a microcontroller – a compact, beginner-friendly platform that empowers pupils to dive into the world of programming. This openness paves the way for endless possibilities, from crafting simple interactive challenges to building intelligent systems later, nurturing skills that could evolve into advanced technological endeavors in the future.



Ing. Monika Spóťová. Programme Manager at the non-profit organisation Junior Achievement Slovakia, where she has been working since 2019. Since 2021, she has been leading the Digital Competencies for Teachers educational programme, which supports the innovative use of technology in schools (<https://www.digikoordinatori.jaslovensko.sk/>). She graduated from the Faculty of Business at the University of Economics in Bratislava, majoring in International Business. Her work combines managerial thinking with a practical view of the needs of Slovak education. She is interested in developing the potential of

teachers and schools, which she also does as a lecturer, coach and mentor of strengths according to the Gallup CliftonStrengths methodology. She has co-created educational tools and webinars that connect digital skills with critical thinking. Her work focuses on promoting sustainable change and sharing good practice between schools.

Ing. Marek Klimo. Studied information systems security at the Slovak University of Technology and is currently a PhD student at the University of Žilina. His current research is focused on the object retrieval and 3D modelling of anatomy. He teaches Introduction to Studies, Informatics for Managers 2, Fundamentals of Theoretical Medicine. He loves Open Source tools, such as Blender and Linux. In the free time he plays guitar, practices archery, and loves cooking.



PUBLIKACE $\text{\LaTeX}$ OVÝCH DOKUMENTŮ NA WEB POMOCÍ $\text{\TeX}$ 4HT A GITHUB ACTIONS

Michal HOFTICH (CZ)

VYŽIADANÁ PREDNÁŠKA

Abstrakt: Přednáška představí sadu šablon pro nástroj $\text{\TeX}$ 4ht, který slouží k převodu $\text{\LaTeX}$ ových dokumentů do HTML. Tyto šablony výrazně usnadňují publikaci různých typů dokumentů na webu a přinášejí moderní možnosti zpracování a automatizace.

První šablona je určena pro převod knižních dokumentů do webové podoby. Umožňuje rozdělení textu do jednotlivých kapitol s automaticky generovanou navigací a podporou responzivního designu, takže je výsledek dobře čitelný i na mobilních zařízeních.

Druhá šablona slouží k tvorbě staticky generovaných blogů. Každý příspěvek je psán jako samostatný $\text{\LaTeX}$ ový dokument, který je pomocí $\text{\TeX}$ 4ht převeden do HTML. Následně jsou tyto články zpracovány statickým generátorem webů, jako je například Jekyll, který se postará o sestavení celého blogu, vytvoření rozcestníků, archivů a další navigace.

Třetí šablona je zaměřena na převod prezentací vytvořených v prostředí Beamer do formy tzv. handoutů – přehledových materiálů pro posluchače. Výsledkem je čitelný a dobře strukturovaný webový dokument vhodný pro sdílení po přednášce.



Všechny šablony jsou navrženy tak, aby fungovaly v rámci GitHub Actions. To znamená, že dokumenty mohou být automaticky zkompilovány a publikovány online pokaždé, když dojde ke změně v repozitáři. Tento přístup zajišťuje, že je webová verze dokumentu vždy aktuální.

Michal Hoftich. Michal je knihovník z Prahy. Je autorem řady balíčků pro $\text{\LaTeX}$ a vývojářem konverzního nástroje $\text{\TeX}$ 4ht. Kromě toho rád chodí do přírody.

ŽIVOTNÍ POUŤ V OPTIMALIZACI

VAŠEK CHVÁTAL (CZ)

Abstrakt. Příspěvek je rozhovor s Panem profesorem Vaškem Chvátalem, matematikem, vysokoškolským pedagogem a uznávaným vědcem, který pan profesor laskavě poskytl spoluorganizátorovi konference OSSConf, známému to nadšenci, Pavlovi Střížovi.

Vážené čtenářky, vážení čtenáři,

v Informačním bulletinku České statistické společnosti jsme představili tři knihy *Combinatorics: Ancient & Modern*, *Competitive Programming 4* či *Opt Art* od Roberta Bosche. Všechny mají něco společného, a to, že zmiňují našince, pana profesora Vaška Chvátala, <https://users.encs.concordia.ca/~chvatal>. Shodou okolností se pan profesor vrátil do Čech, do Prahy, na rodnou Alma Mater <https://kam.mff.cuni.cz/~chvatal/>. Ale nebudu předbíhat!



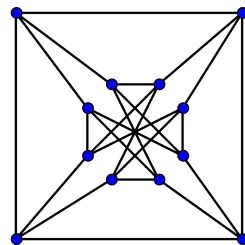
Vítejte, pane profesore, v naší skromné zpovědnici. Prozradte nám, prosím, jak ta Vaše životní pouť v matematice začala.

Svým způsobem mou účastí v matematických olympiádách, i když tou dobou jsem měl podobných koníčků víc. Na Matfyz jsem se přihlásil s úmyslem studovat teoretickou fyziku, protože mým tehdejším snem bylo, že vysvětlím závislost atomové váhy různých prvků na jejich atomovém čísle. Tam se mě ujal Zdeněk Hedrlín a ten mě přetáhl k matematice.

Řada lidí pozná Chvátalův graf, který zmiňuje i profesor D. E. Knuth u programu GraphBase, <https://www-cs-faculty.stanford.edu/~knuth/sgb.html>. Zde je ukázka. Publikován byl v roce 1970, ale to nebyla Vaše první publikační činnost, že?

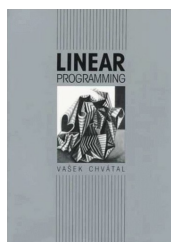
Jako dospívající jsem publikoval miniaturní povídky a kreslené vtipy v humoristicko-satirickém časopise Dikobraz, ale o to Vám nejspíš nejde. Můj první matematický článek vzniknul pod Hedrlínovým vedením. Předložil mi problém z teorie

grafů a když jsem ho začátkem roku 1965 vyřešil, řekl mi, ať to řešení sepíšu. Poznamenan atmosférou rakousko-uhersko-komunistické hierarchie jsem čekal, že se pod výsledek podepíšeme oba, ne-li on sám, ale vyrazil mi dech prohlášením, že si odmítá přisvojit uznání za mou práci, takže to mám podepsat sám. Pak ten rukopis odnesl do redakce časopisu *Commentationes Mathematicae Universitatis Carolinae*, kde vyšel ještě téhož roku: <https://dml.cz/handle/10338.dmlcz/105034>. Hedrlín byl nevšední pedagog a já mu vděčím za mnoho.



Pokud se díváme na Vaši publikační činnost, je obsáhlá, <https://users.ensc.concordia.ca/~chvatal/publ.html>, ale zaujme ještě jedna věc: „joint with P. Erdős“. Řekněte nám prosím víc o této jedinečné spolupráci.

Začalo to zase Hedrlínem. Vyjednal pro mne, že se Erdősovi představím během jeho návštěvy Slovenské akademie vied. Díky tomu pak nade mnou Erdős držel ochrannou ruku, když jsem tři dny po ruské invazi odešel do Vídně. Poslal mi tam peníze s tím, že mu je vrátím, až je nebudu potřebovat, a kromě toho nechal pro mne na několika kanadských univerzitách doporučení ke stipendiu. Náš první společný článek vznikl na jaře 1971. Řekl jsem mu o jedné své hypotéze, on ji během naší dvouhodinové společné jízdy v autě dokázal, a já jsem pak ten důkaz sepsal. Náš druhý společný článek vznikl o půl roku později podobným způsobem. Řekl jsem mu o jiné své hypotéze, on během naší společné večere dokázal její slabší verzi, a já jsem pak ten důkaz sepsal. Náš třetí společný článek vznikl na jaře 1976 tradičnějším způsobem, když jsme pinkali nápady jeden druhému jako pingpongový míček. Já jsem to pak sepsal.



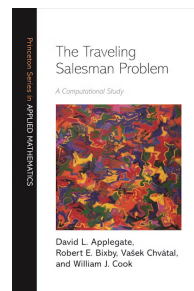
Studenti si pochvalují Vaši učebnici *Linear Programming* o lineárním programování z roku 1983.

Takovým studentům blahopřeju k jejich vytříbenému vkusu. Tu knihu jsem se rozhodl napsat, když jsem v sedmdesátých letech na Stanfordu dostal za úkol učit lineární programování a žádná učebnice nevyhovovala mému pohledu na ten předmět. To se vlastně zdá být motivací spousty autorů.

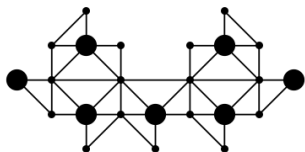
Jeden z nejvýznamnějších počínů je program TSP Concorde. Nově se dá vyzkoušet i přes NEOS server, <https://neos-server.org/neos/solver/s/co:concorde/TSP.html>. V roce 2007 vychází kniha *The Traveling Salesman Problem: A Computational Study*.

To začalo před Vánoci roku 1987 v baru restaurace Café Bruxelles v Greenwich Village. Můj kamarád Bill Cook a já jsme už měli společný matematický článek

<https://kam.mff.cuni.cz/~chvatal/discipline.pdf> s odkazem na markýze de Sade a rozhodli jsme se podobnou hravost uplatnit v oblasti softwaru pro problém obchodního cestujícího, kde vládla tvrdá soutěživost. Bill mi tenkrát říkal, že by bylo zajímavé v té aréně vyprovokovat trochu rozruchu a případně tam vyhrát zlatou medaili. Nakonec mě do toho ukecal a ke spolupráci na tom projektu jsme přizvali fenomenálního programátora Davida Applegata, který se během následujících let taky stal jedním z mých nejlepších kamarádů. Později se k nám přidal Bob Bixby se svým softwarem CPLEX pro lineární programování. Tu zlatou medaili jsme vyhráli v dubnu 1992 řešením relativně těžkého případu pcb3038 s 3038 městy a pak se to skutečně rozjelo. Více o tom je na <https://users.encs.concordia.ca/~chvatal/talks/KAM1.pdf> a v přednášce <https://slideslive.com/38895310>.



Pokud někdo z čtenářů sleduje americký seriál *Numb3rs*, tak v druhé řadě třetí epizodě je zmínka o Art Gallery, <https://mathworld.wolfram.com/ArtGalleryTheorem.html>.



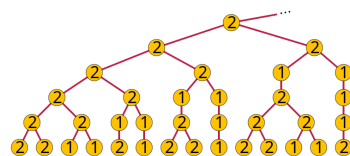
Jednou jsem si všimnul, že v každé místnosti s pětiúhelníkovým půdorysem je bod, ze kterého je vidět celá místnost, ale pro šestiúhelníkový půdorys to už platit nemusí. Na konferenci v srpnu 1973 jsem se o tom zmínil Victoru Klee a ten se mě hned zeptal, jaké je nejmenší $f(n)$, pro které v každé místnosti s n -úhelníkovým půdorysem existuje $f(n)$ bodů takových, že každý bod v té místnosti je vidět z nějakého z nich. Vic Klee byl jedním z mých vzorů, vážil jsem si ho a měl jsem ho moc rád, a tak jsem na něj chtěl udělat dojem. Proto jsem se do jeho otázky okamžitě zakousnul a než konference skončila, měl jsem pro něj odpověď: $f(n)$ je $n/3$ zaokrouhleno dolů. Netušil jsem tenkrát, jakou lavinu publikací tenhle můj výsledek vyvolá.

Neodpustím si zmínku ještě na jednu zajímavost: *The Kolakoski Sequence* (<https://oeis.org/A000002>), ke které máte publikován článek.

Kolakoskiho posloupnost je 1221121221221121122

1211212211211212212211212212112... a tak dál:

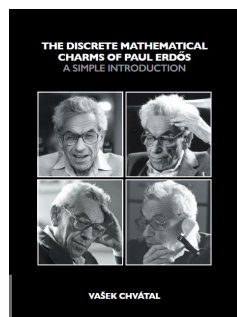
její k -tá číslice je délka jejího k -tého bloku. Jedna z hypotéz o téhle fascinující posloupnosti říká, že frekvence jejích jedniček (a frekvence jejích dvojek) je 0,5. Já jsem popsal metodu dokazování mezí (horních a dolních) této frekvence pomocí výpočtů <https://users.encs.concordia.ca/~chvatal/93-84.pdf>. Ty meze se dostanou čím blíže k 0,5, čím více paměti vám poskytne váš počítač. Tak mi v roce 1993 umožnil můj počítač dokázat, že frekvence jedniček nepřesahuje 0,500838 a o dvacet let později umožnil Johanu Nilssonovi počítač



s větší pamětí snížit tu mez na 0,500080, <http://przyrbwn.icm.edu.pl/APP/PDF/126/a126z2p31.pdf>.

Váš poslední knižní počín není nic menšího než *The Discrete Mathematical Charms of Paul Erdős: A Simple Introduction*, 2021. Ta se již dočkala čínského a japonského překladu.

V té knížce jsem se pokusil skloubit vlídné vysvětlování matematiky s vlastními vzpomínkami na Erdőse. Víc o tom je na <https://kam.mff.cuni.cz/~chvatal/DMCPE.pdf>. Mám z ní velkou radost a tak Vám děkuju, že ji čtenářům představujete.



Poslední otázka, pane profesore, mohu-li. Dorazíte do Žiliny na konferenci OSSConf, kde bude mimo jiné představen Concorde TSP Solver, zajímavé užití TSP v keškách a budou zmíněny některé varianty problému obchodního cestujícího?

Děkuji za pozvání! Nikoliv, nemohu dorazit, ale přeji účastníkům konference hodně zdaru a srdečně Vás všechny pozdravuji!

Pane profesore, děkujeme velice za rozhovor!

Pro zaujaté čtenáře můžeme doporučit:

- Článek v časopisu *Graphs and Combinatorics*, https://kam.mff.cuni.cz/~chvatal/vasek_gc.pdf.
- Rozhovor *Dobrodružství mého života* pro časopis Matfyz, <https://www.matfyz.cz/clanky/dobrodruzstvi-meho-zivota>.
- Den pro Zdeňka Hedrlína, <https://slideslive.com/38908216/den-pro-zdenka-hedrlina>.
- Povídání pro *Subject to*, www.youtube.com/watch?v=WzEVi32jaR0, kde se člověk skutečně zasmál!

Kontaktní adresa

prof. Vašek Chvátal, Ph.D., KAM MFF UK, Malá Strana, Malostranské nám. 2/25, Praha 1, CZ-11800, Česká republika,
E-mailová adresa: chvatal@kam.mff.cuni.cz

SMART INTERNETOVÉ RÁDIO NA PLATFORME ESP32

MIROSLAV BENDÍK (SK)

Abstrakt. Článok opisuje návrh a implementáciu inteligentného internetového rádia založeného na platforme ESP32. Riešenie využíva open source prostredie ESP-IDF a operačný systém FreeRTOS. Digitálny zvuk sa prenáša cez rozhranie I²S do prevodníka MAX98357. MP3 stream sa prijíma pomocou protokolu HTTP a dekoduje knižnicou MAD. Konfigurácia DMA zaručuje plynulé prehrávanie. Vývoj urýchľuje POSIX simulátor FreeRTOS pre Linux, ktorý umožňuje detailné trasovanie plánovania úloh. Dialkové ovládanie hlasitosti a prepínanie staníc zabezpečuje webové rozhranie. Projekt demonštruje, ako možno vďaka otvorenému softvéru a hardvéru vytvoriť cenovo dostupné, rozšíriteľné a reprodukovateľné IoT zariadenie.

Kľúčové slová. ESP32, ESP-IDF, FreeRTOS, WiFi, HTTP, MP3, Internetové rádio.

SMART INTERNET RADIO ON THE ESP32 PLATFORM

Abstract. This article presents the design and implementation of a smart Internet radio based on the ESP32. The solution relies entirely on the open source ESP-IDF SDK and the FreeRTOS operating system. Audio is delivered over the digital I²S bus to a MAX98357 DAC/amplifier. Incoming MP3 streams are fetched via HTTP and decoded by the MAD library. The DMA configuration guarantees glitch free playback. A POSIX-based FreeRTOS simulator for Linux accelerates development and enables detailed task-switch tracing. A web interface allows remote station selection and volume control from any mobile or desktop browser. The project demonstrates how open source hardware and software can be leveraged to create affordable, reproducible and easily extensible devices.

Keywords. ESP32, ESP-IDF, FreeRTOS, WiFi, HTTP, MP3, Internet radio.

Úvod

Mikrokontroléry ESP32 sú pokračovaním úspešnej série ESP8266. Pôvodne mali tieto mikrokontroléry poskytovať pripojenie k Wi-Fi cez jednoduché rozhranie USART. Výrobcom dodávaný firmvér mal implementovanú podporu AT príkazov. Firmvér sa dal prepísať a tak zariadenie mohlo plniť aj iné funkcie než len jednoduchý „modem“ pre Wi-Fi.

Pri návrhu novej generácie mikrokontrolérov — ESP32 — už výrobca bral do úvahy, že čip nemusí slúžiť len ako modem, ale môže pracovať ako plnohodnotný mikrokontrolér. Nový čip mal vyšší výkon a viac vstupno-výstupných rozhraní. Prvá generácia mala dvojjadrový CPU taktovaný na 240 MHz. Medzi rozhrania pribudol napríklad zvukový výstup I²S (Inter-IC Sound).

Kombinácia Wi-Fi rozhrania, digitálneho zvukového výstupu I²S a nízkej ceny z tohto mikrokontroléra robí vhodného kandidáta na prehrávač internetových rádií.

1. Hardvér

Ako mikrokontrolér som zvolil ESP32-WROVER namontovaný na jednej z dostupných vývojových dosiek. Napriek možnosti používať vstavaný analógový výstup som sa rozhodol využiť plne digitálne rozhranie I²S. MCU (Mikrokontrolér) výrazne rozširuje možnosti zvukového rozhrania, ktoré by sa skôr mohlo označovať ako **multimedia interface**, pretože umožňuje obojsmernú komunikáciu v sériovom aj paralelnom režime, čo sa využíva napríklad pri práci s kamerou.

Keďže je výstup digitálny a naše uši sú analógové, musíme do zariadenia zapojiť aj digitálno-analógový prevodník. Rozhodol som sa pre MAX98357, ktorý integruje prevodník a 3 W zosilňovač. Ide o veľmi jednoduché zariadenie, ktoré sa dá pomocou rezistorov nakonfigurovať na výstup ľavého, pravého alebo zmiešaného kanála. Pri použití dvoch kusov je možné zostaviť stereofónny výstup. Oba prevodníky prijímajú ten istý stereo signál, ale vďaka konfigurácii dekoduje každý samostatný kanál. Jeden kus však vie fungovať aj v mono režime, keď kombinuje oba kanály.

Čip MAX98357 je veľmi jednoduchý, pretože nemá žiadny ovládací protokol; dokonca ani ovládanie hlasitosti. Možno nastaviť iba niekoľko úrovní základného zosilnenia pomocou konfigurácie rezistorov. Ovládanie hlasitosti preto bude softvérové. Vstupné dáta sú 32-bitové, čo by v prípade 32-bitového DAC prevodníka poskytovalo dostatočný priestor na softvérovú zmenu hlasitosti bez straty kvality. Či je tomu skutočne tak, môžeme len hádať, pretože výrobca túto informáciu nezverejňuje.

2. Vývojové prostredie

Výrobca zverejňuje k hardvéru vývojové prostriedky ESP-IDF pod Open Source licenciou Apache 2.0 pre operačné systémy Linux, macOS a Windows. Ide o balík zahŕňajúci build systém (CMake), kompilátor (GCC), knižnice HAL (Hardware Abstraction Layer), systémové knižnice, FreeRTOS, ovládače a nástroje na flashovanie a ladenie. Takmer všetky časti sú plne Open Source okrem malej časti ovládačov Wi-Fi, Bluetooth a secure boot.

Programy je možné vyvíjať aj s Arduino knižnicami, ktoré však v konečnom dôsledku len zapuzdrujú ESP-IDF. V tomto projekte sa snažím z hardvéru vyťažiť maximum a preto používam len minimálne potrebnú abstrakciu. V jednom prípade dokonca obchádzam HAL a manipulujem s registrami MCU priamo.

3. FreeRTOS

Vývojové prostredie ESP-IDF v sebe zahŕňa jednoduchý operačný systém určený pre embedded zariadenia — FreeRTOS. V pravom slova zmysle to nie je operačný systém, ale skôr zbierka funkcií a algoritmov pre správu a časovanie úloh. Na takto obmedzenom hardvéri, akým je ESP32, sa typicky nespúšťa operačný systém s podporou izolovaných procesov; celý systém a jednotlivé úlohy/obsluhy ovládačov fungujú v rámci jedného procesu.

Navonok API FreeRTOS pripomína skutočný operačný systém, aj keď s kooperatívnym multitaskingom. Riadenie sa z úlohy musí explicitne vracat' operačnému systému volaním `taskYIELD()`. V tom momente preberie kontrolu FreeRTOS, prehľadá zoznam úloh a podľa typu plánovača vyberie ďalšiu úlohu alebo `idle` úlohu s najnižšou prioritou.

Okrem explicitného prepínania úloh vie FreeRTOS obsluhovať aj prerušenia a, v prípade ESP32, dokonca aj paralelný beh na viacerých CPU, keďže tento MCU obsahuje dve jadrá.

3.1. Simulátor

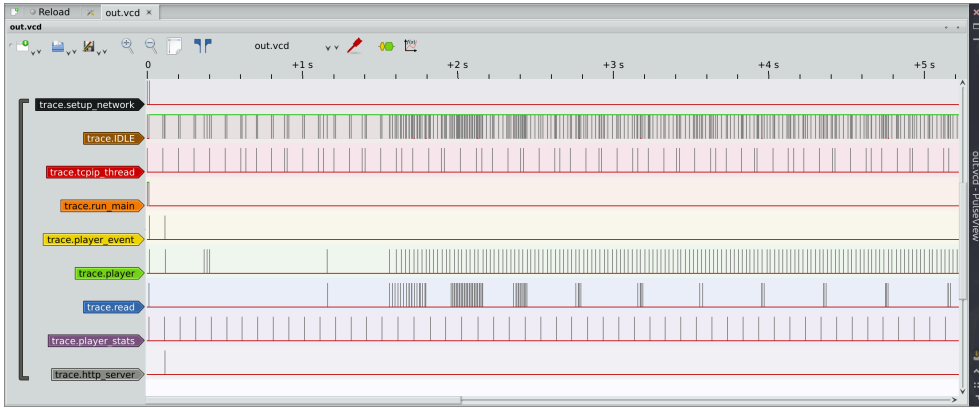
Pri písaní rozsiahlejších aplikácií či algoritmov môže byť neustále nahrávanie a kontrola na zariadení časovo náročné. Aby som sa vyhol najpomalšiemu kroku, pripravil som simulátor FreeRTOS pre Linux. Ide o aktualizovanú verziu FreeRTOS portu pre POSIX/Linux, ktorú mám zverejnenú na svojom GitHubu (https://github.com/mireq/Posix_GCC_Simulator). Na pozadí používa skutočné linuxové vlákna, ktoré sú medzi sebou synchronizované tak, aby beh čo najviac pripomínal skutočné zariadenie.

3.2. TCP/IP

Aby sa sieťové požiadavky správali čo najbližšie k skutočnému zariadeniu, je použitá rovnaká user-space implementácia TCP/IP prostredníctvom knižnice lwIP (A Lightweight TCP/IP stack). Simulátor využíva virtuálne sieťové rozhranie TUN/TAP.

3.3. Ladenie plánovania úloh

Správne nastavenie priorít a plánovania úloh môže byť na systéme s kooperatívnym multitaskingom tvrdým orieškom. Pre analýzu som pripravil trasovací nástroj na zaznamenávanie aktívnych úloh do súboru a skript na konverziu súboru do formátu VCD (Value Change Dump) zobraziteľného napríklad v nástroji PulseView. Ukážka záznamu je na Obrázku 1.



Obr. 1. Zobrazenie prepínania bežiacich úloh

4. Implementácia

Kód rádia sa skladá z niekoľkých menších komponentov. V nasledujúcej sekcii predstavím niektoré z nich.

4.1. Inicializácia

Nastavenia siete a predvolenej stanice sú v konfiguračnom súbore, z ktorého sa generujú konštanty v hlavičkovom súbore. Po skompilovaní má priamo binárny súbor v sebe informácie o pripojení k Wi-Fi a statickej či dynamickej konfigurácii siete.

4.2. Čítanie dát zo siete

Na čítanie dát slúži jednoduchý HTTP klient. Dáta sú čítané do vstupno-výstupného bufferu. Pri čítaní sa dekodujú aj metadáta `Icy-MetaData` (súčasť špecifikácie protokolu Icecast). Nepovinné metadáta sa vypisujú len do konzoly. V budúcnosti je priestor na rozšírenie rádia o stavový displej so zobrazením aktuálnej skladby.

Veľkosť bufferu je nastavená na 1 MB, čo predstavuje približne 64 s pomerne kvalitného 128 kb/s MP3 streamu. Oblasť je alokovaná v pomalšej sériovej pamäti PSRAM. Jej celková kapacita je 4 MB. Medzi PSRAM a CPU je malá vyrovnávacia pamäť, vďaka čomu je externá pamäť dostatočne rýchla na odkladanie sieťových bufferov. Pre alokáciu v PSRAM sa používa príznak `MALLOC_CAP_SPIRAM`:

```
heap_caps_malloc(
    STREAM_BUFFER_SIZE,
    MALLOC_CAP_DEFAULT | MALLOC_CAP_SPIRAM
)
```

4.3. Dekódovanie

Na dekodovanie streamu som použil knižnicu MAD (MPEG Audio Decoder). Ide o knižnicu, ktorá používa aritmetiku bez plávajúcej desatinnej čiarky. Výstupom volania dekodéra na časť streamu je typicky 576 čísel reprezentujúcich hodnotu signálu. Tie sú následne upravené podľa požadovanej hlasitosti a odoslané do zariadenia I²S.

4.4. Zvukový výstup

Pri zvukovom výstupe sa prejavil veľmi šikovný návrh DMA (Direct Memory Access). Platforma ESP32 umožňuje zretazovanie DMA požiadaviek, vďaka čomu je možné posilať neprerušovaný tok dát cez viacero zretazených bufferov. Celá komplexnosť DMA systému je však skrytá pod abstraktnou vrstvou a stačí volať funkciu `i2s_write()`. O rozkúskovanie požiadavky a prepínanie úloh počas prenosu sa postarajú knižnice ESP-IDF. Zatiaľ čo DMA automaticky posiela dáta cez I²S rozhranie, môže CPU dekodovať ďalší MP3 rámec.

Jedna časť ESP-IDF však nebola navrhnutá najlepšie — konkrétne výpočet deličky vzorkovacej frekvencie audio výstupu. Preto som využil priamy prístup k registrom a ručne vypočítal parametre deličky tak, aby sa lepšie približovali k požadovanej frekvencii. Drobným bonusom je možnosť jemne meniť frekvenciu bez vypnutia audio výstupu. Jemné spomalenie sa dá využiť napríklad na predchádzanie podtečeniu bufferu pri live streamoch, keď je frekvencia oscilátora ESP32 o kúsok vyššia, než by mala byť.

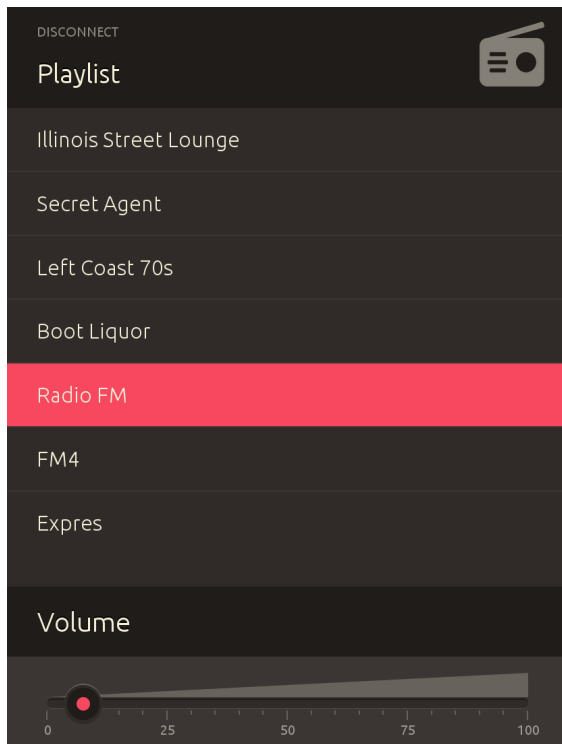
4.5. Ovládanie

Zariadenie je možné ovládať na diaľku cez webové rozhranie. Systém obsahuje jednoduchý HTTP server, ktorý implementuje presmerovanie klienta na statické úložisko (aby nebol celý web uložený na zariadení) a API cez WebSocket rozhranie. Ovládanie umožňuje zmenu hlasitosti a prepínanie stanice.

Webová aplikácia zároveň funguje ako PWA (Progressive Web App), vďaka čomu sa dá nainštalovať do telefónu ako natívna aplikácia. Vzhľad rozhrania je zachytený na Obrázku 2.

5. Záver

Platforma ESP32 síce nie je z hľadiska Open Source plne otvorená, pretože malá časť ovládačov zostáva uzavretá, zvyšok vývojového prostredia je však otvorený. Aplikácia s uzavretou časťou vôbec nekomunikuje, všetko je zabalené v štandardnom POSIX rozhraní, prípadne v nízkoúrovňovom rozhraní lwIP. Kód je preto prenositeľný aj na inú platformu, čo demonštruje použitie simulátora FreeRTOS. Vďaka tomu, že výrobca zvolil Open Source knižnice, je možné skompilovať tú istú aplikáciu pre MCU aj pre Linux bez prepisovania kľúčových častí.



Obr. 2. Ovládanie rádia cez webové rozhranie

Budúca práca sa zameria na podporu ďalších audio formátov (napríklad AAC či FLAC), integráciu Bluetooth Audio. Plánujeme tiež doplniť displej zobrazujúci názov prehrávanej skladby.

Kontaktná adresa

Ing. Miroslav Bendík, Kukučínova 480/1, 059 18 Spišské Bystré, Slovensko,

E-mailová adresa: miroslav.bendik@gmail.com, <https://linuxos.sk/profil/424/>

FAR MANAGER – SPRÁVCA SÚBOROV A ARCHÍVOV

RUDOLF BLÁŠKO (SK)

Abstrakt. File and ARchive Manager (skrátene Far manager) je správca súborov prioritne vytvorený pre Microsoft Windows vytvorený podľa vzoru klasického ikonického správcu Norton Commander. FAR Manager využíva konzolu Win32 a je orientovaný na ovládanie z klávesnice, ale plne funkčne ho možno ovládať aj pomocou myšky. Má výborný vstavaný editor súborov a mnoho funkcionalít, ktoré je možné veľmi jednoducho rozšíriť použitím tzv. pluginov. Väčšina týchto pluginov je napísaná v skriptovacom jazyku Lua.

Kľúčové slová. Správca súborov, príkazový riadok, editor, Open Source.

FAR MANAGER – FILE AND ARCHIVE MANAGER

Abstract. File and ARchive Manager (Far manager for short) is a file manager primarily created for Microsoft Windows, modeled after the classic iconic Norton Commander manager. FAR Manager uses the Win32 console and is oriented towards keyboard control, but it can also be fully functionally controlled with a mouse. It has an excellent built-in file editor and many functionalities that can be easily extended using so-called plugins. Most of these plugins are written in the Lua scripting language.

Keywords. File manager, command line, editor, Open Source.

Úvod

Existuje veľmi mnoho rôznorodých správcov súborov. V tomto príspevku sa budem venovať jednému z najlepších z nich a navyše Open Source a to Far Manager-u. Tento správca súborov je považovaný za klon ikonického Norton Commander-a (NC). NC poznajú všetci používatelia počítačov osemdesiatych a deväťdesiatych rokov pod operačným systémom DOS, keď ešte myš nebola ani štandardným ani lacným zariadením počítačích strojov. Myš je síce záležitosť šesťdesiatych a sedemdesiatych rokov, prvý osobný počítač s myšou Xerox Star sa predával od roku 1981, ale do polovice osemdesiatych rokov bola myš prakticky neznáma. V tých časoch sa všetko riešilo cez príkazový riadok, dokonca aj e-mail.

NC je založený na geniálnom princípe dvoch nezávislých okien vedľa seba, jednoduchým prepínaním a komunikáciou medzi nimi. Toto väčšina novodobých súborových správcov postráda. Je pravda, že sa ovládajú jednoducho a pohodlne myškou, ale toto môže stačiť iba bežnému užívateľovi, ktorý nepotrebuje spravovať a rôzne kopírovať, porovnávať, premenovávať ap. viac súborov. Toto samozrejme dokážu aj títo správcovia, ale už to nie je intuitívna záležitosť ako v prípade správcov ovplyvnených NC. Ale proti Gustovi žiaden dišputát.

Far Manager bol prvý krát predstavený svetu 10. septembra 1996. Vytvoril ho a až do roku 2000 ho vyvíjal Eugene Roshal a považuje sa za bezplatnú alternatívu ku komerčnému Total Commander-u. Od 18. júna uvedeného roku ho vyvíja skupina ruských programátorov pod názvom Far Group. Far manager je jeden z mála, ak nie jediný Open Source program, ktorý dlho neexistoval pod Linux-om ale iba pod Windows. Ak sa vrátim k histórii programu, pôvodne bol Far Manager dostupný iba ako 40-dňový shareware. Ako freeware ho mohli používať iba občania postsovietskych republík ale iba na nekomerčné účely. Dňa 26. októbra 2007 bol zdrojový kód vývojovej verzie Unicode 1.80 vydaný pod licenciou BSD-3-Clause. Neskôr 13. decembra 2008 túto verziu kvôli lepšiemu rozlíšeniu medzi komerčnou a Open Source verziou preznačili na celkom novú sériu 2.00. To bol začiatok Far Manager-a ako slobodného softvéru. 17. mája 2010 oznámili autori, že aj komerčná verzie Far Manager-a 1.75 je uvoľnená pod slobodnou licenciou BSD-3-Clause, hoci bez zdrojového kódu. Tým sa stal Far Manager úplne Open Source, ale v tom čase stále fungoval iba pod OS Windows.

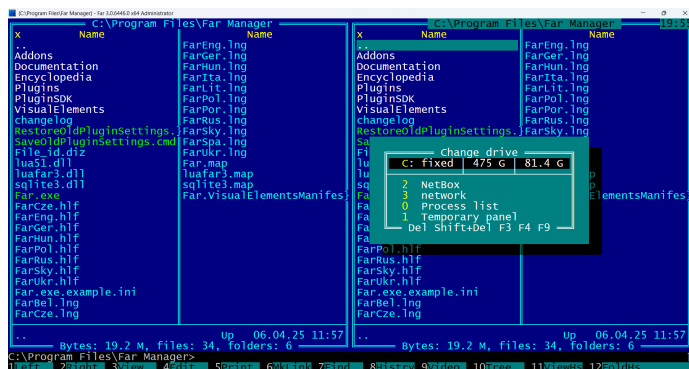
Pod Linuxom existuje jeho obdoba Midnight Commander (<https://midnight-commander.org/>), ktorý je implementovaný prakticky v každej jeho distribúcii a každý Linuxák ho určite pozná. Midnight Commander (MC) u nás na fakulte známy pod názvom „Modrý čert“, je vizuálne aj funkcionálne podobný Far Manager-u, ale užívatelia, ktorí poznajú oboch správcov mi potvrdia, že nedosahuje kvality Far Manager-a. Far Manager je od 20. septembra 2016 dostupný ako **far21** aj pre OS Linux, Mac OS a BSD. V tomto čase funguje základná funkcionálna Far Manageru ako správca súborov a archívov. Boli portované a fungujú aj pluginy FarFTP, Colorer, MultiArc a TmpPanel. V novembri roku 2020 boli pridané plugin-y NetRocks (s protokolmi NetBox, FTP, FISH, SCP, SFTP, SMB, NFS a WebDAV), podpora práce v grafickom a terminálovom režime, bolo portovaných najmenej 13 pluginov pôvodného Far Manager-a. V januári roku 2022 vstúpila táto časť projektu do beta fázy a bola pridaná podpora pre skripty Python-u a Lua.

Far Manager môžeme stiahnuť z jeho domovskej stránky <https://www.farmanager.com/opensource.php?l=en>, kde nájdeme aj aktuálnu verziu FAR3.0 v stabilnej aj „nightly“ verzii, v 32 a aj 64 bitovej verzii. Na tejto stránke sú ale ponúkané iba verzie pre Windows, pre ostatné OS nájdeme link na GitHub a viac informácií pre Linux nájde vnímavý čitateľ na stránke <https://github.com/elfmz/far21>. Pre Linux sú verzie FAR2L, napr. pre Ubuntu distribúciu môžeme Far Manager nainštalovať pomocou príkazu `sudo apt install far21`.

1. Základná charakteristika

Program je napísaný v jazyku C++ a je podporovaný pod operačnými systémami Microsoft Windows (prioritne), Linux, Mac OS a BSD Unix, je dostupný v rôznych jazykových mutáciách (angličtina, ruština, slovenčina, čeština,

Far Manager (skrátka pre File and ARchive Manager) je považovaný za ortodoxného správcu súborov na princípe funkcionality NC, používa konzolu Win32 a má používateľské rozhranie orientované na klávesnicu (hoci je možné obmedzené ovládanie myšou vrátane funkcie drag-and-drop, t. j. uchopiť objekt myšou a premiestniť ho na nové miesto).



Obr. 1. Predvolené rozhranie Far Manager

Predvolené rozhranie Far Manageru kombinuje dva panely (ľavý a pravý) súborov a príkazový riadok (obr. 1). Medzi panelmi sa prepína klávesom TAB. V rámci okna sa dá pohybovať šípkami, resp. pomocou PgUp a PgDn. Pokiaľ nie je nič označené, aktívny je súbor/adresár, na ktorom je kurzor. Súbor a adresár sa označujú klávesom INS – zmení sa farba jeho zobrazenia. Ale pozor ak je označený nejaký súbor a kurzor je na inom mieste, prioritu má označený súbor a na tento označený súbor sa budú vzťahovať všetky prípadné aktivity. Hromadne sa súbory môžu označiť stlačením SHIFT a šípiek. Samozrejme sa môže použiť aj

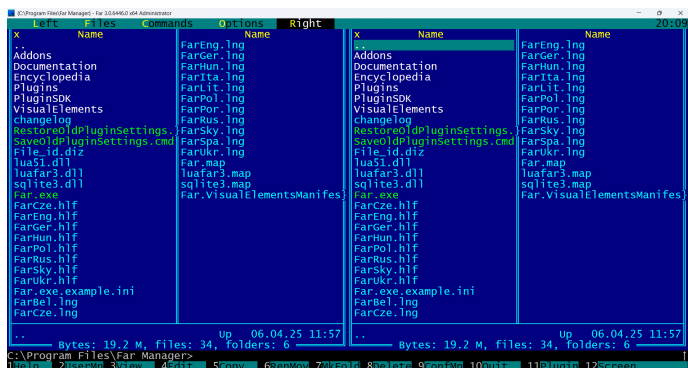
myška, ale toto nie je Microsoft Office, takže to je nepohodlné. Napríklad kopírovať a presúvať viaceré súbory môžeme jednotlivo, alebo do zjednotiť do jedného súboru.

Panely je možné úplne prispôsobiť (zobrazenie stĺpcov, ich počet, poradie zobrazenia súborov a adresárov, atribúty a ich vlastnosti, ako sú čas vzniku, čas modifikácie, veľkosť, meno a prípona ap.). Operácie je možné vykonávať na jednom paneli (napr. triedenie, mazanie, vytváranie (aj hromadných) súborov a adresárov, ich editovanie a prezeranie ap.) a taktiež z jedného panela na druhý a späť navzájom (kopírovanie, porovnávanie ap.). Štandardnú funkcionality Far je možné výrazne rozšíriť pomocou makier (napísaných v skriptovacom jazyku Lua, ktoré sa primárne používajú na zaznamenávanie sekvencií stlačení klávesov) a doplnkov, tzv. klávesové makrá. Medzi doplnky nainštalované štandardne patria FTP, sieť Windows, podpora rozširiteľných archívnych súborov a dočasné panely (sandbox) virtuálne súborové systémy, zoznam procesov, prevodník veľkých a malých písmen v názvoch súborov a niekoľko doplnkov editora na formátovanie, zalamovanie a inú zmenu textu.

V jednotlivých paneloch si môžeme zobraziť ľubovoľný adresár na disku vrátane skrytých súborov, ale aj z prístupnej siete Windows (voľba Network), z ľubovoľného servera na sieť pomocou protokola FTP, resp. SSH, SCP (voľba plugin-u NetBox), prípadne iné nainštalované plugin-y. Ďalej to sú zoznam aktuálne bežiacich procesov v počítači (Process list) alebo dočasný panel (Temporary panel), ktorý môžeme použiť a dočasne tam nahrať rôzne súbory a adresáre z rôznych častí disku PC a siete a s nimi pohodlne pracovať. Panely súborov navyše podporujú výber zástupných znakov, pokročilé filtrovanie, triedenie a zvýrazňovanie. Panely súborov a príkazový riadok sú aktívne súčasne (interagujú sa pomocou rôznych klávesov) a väčšina funkcií je prístupná pomocou klávesových skratiek.

V dolnej časti je možné zobraziť ale nezobraziť (CTRL B) panel akcie funkčných klávesov. Týchto panelov je prakticky viac, základný je pre klávesy F1 až F12, ostatné sa modifikujú pri stlačení klávesov CTRL, ALT, SHIFT, ALT SHIFT a CTRL SHIFT plus klávesy F1 až F12. Pod týmito klávesovými skratkami sú predefinované rôzne operácie, napr. F4 je editovanie označeného súboru, F5 je kopírovanie a F6 je presun označených súborov a adresárov do druhého okna, F7 je tvorba adresárov (vrátane možnosti hromadnej tvorby naraz viacerých adresárov). Samozrejme sú k dispozícii aj notoricky známe klávesové skratky ako sú CTRL A, CTRL X, CTRL C, CTRL V ap., pričom tieto skratky je možné predefinovať podľa vlastných predstáv. Far je oveľa mocnejší ako mnoho moderných grafických súborových manažérov, umožňuje výber za pomoci wildcards (konvencie pre hromadné označovanie hviezdikou a otáznikom), má jednoducho tvoriteľné makrá postupnosť klávesov, rozšírené filtrovanie, zvýrazňovanie ap. Má veľmi účinný príkazový riadok, ktorý je na rozdiel od NC dobre integrovaný a neobmedzuje sa len na jednoriadkové príkazy. Ďalej obsahuje interný zobrazovač a editor textu, prispôsobiteľné menu pre užívateľov, zobrazenie adresárov v stromovej štruktúre,

vyhľadávanie súborov, porovnávanie, integrovaného pomocníka a prepínač úloh pre svoje nástroje.



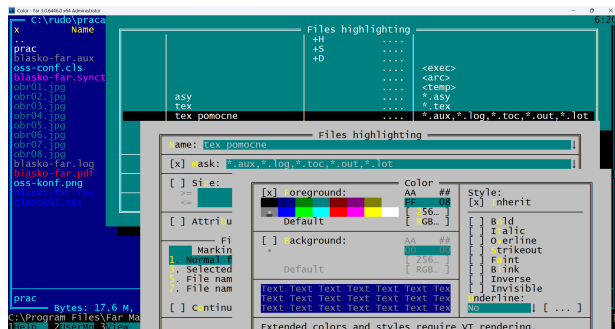
Obr. 2. Horné menu Far Manager

Po stlačení F9 sa zobrazí horné menu (obr. 2), kde môžeme nastaviť Far Manager podľa svojich predstáv. Menu obsahuje päť položiek: **Left** a **Right** (nastavujú sa parametre pre jednotlivé okná), **Files** (nastavujú s atribúty, vyberajú, kopírujú, presúvajú ap. jednotlivé súbory a adresáre vrátane vzdialených súborov na Linuxových počítačoch), **Commands** (história príkazov, história otvorených súborov, porovnávanie adresárov ap.), **Options** (systémové nastavenia, nastavenia panelov, interfejsu, jazyka zobrazenia a nápomoci Help F1, menu, farieb, editora, masky skupín, kódovania atď.). Niektoré nastavenia sa uložia natrvalo, ale niektoré iba pre aktuálne zobrazenie programu, ak chceme zachovať zmeny natrvalo, musíme ich uložiť v menu **Commands** alebo klávesovou skratkou **SHIFT F9**.

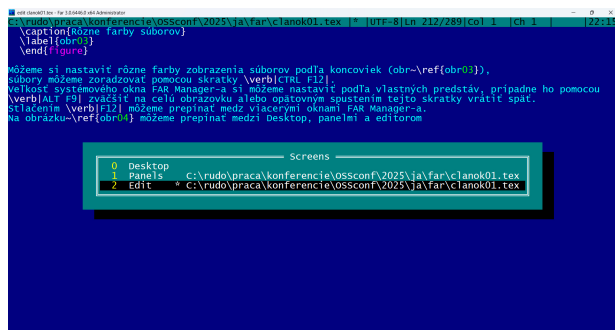
Veľmi silnou pomocou pre používateľov je tvorba vlastných klávesových makriér, t.j. zapamätanie si určitej sekvencie klávesov do vlastnej skratky. Toto makro sa inicializuje stlačením klávesov **CTRL .** (ctrl+bodka), potom sa vykoná sekvencia postupných stlačení (môže obsahovať aj prechod medzi jednotlivými oknami, otvorenie a editovanie súboru, kopírovanie časti súboru, prechod na zvolené miesto súboru ap.). Makro sa ukončí opätovným stlačením **CTRL .**, potom nás Far Manager vyzve na uloženie makra, pričom môžeme zvoliť stlačenie **CTRL**, **SHIFT**, **ALT** alebo ich kombinácie spolu s ľubovoľnou klávesou, napr. **CTRL B**. Makro môžeme potom bez problémov používať. Ak ho chceme používať natrvalo, musíme si ho uložiť pomocou **SHIFT F9**. Je samozrejmé, že takto vytvorené vlastné makrá môžeme hocikedy prepísať.

Práca s príkazovým riadkom je tiež veľmi pohodlná, je k dispozícii história príkazov **CTRL E**, ktorou môžeme prechádzať a spúšťať zvolené príkazy, prípadne ich kopírovať. Okno s históriou otvoríme klávesovou skratkou **ALT F11**. Príkazovou skratkou **CTRL O** sa skryjú oba panely a potom môžeme históriou prechádzať pomocou šípiek hore a dole. Systém automaticky dopĺňa (ponúka voľbu vhodných

alternatív v okne) jednotlivé príkazy a názvy vhodných súborov. Ak príkaz alebo spustiteľný súbor potvrdíme klávesom ENTER spustí sa v prostredí Far Manager-a, ale ak ho potvrdíme SHIFT ENTER spustí sa v novom okne na pozadí. Otváranie rôznych súborov podľa koncovky môžeme nastaviť nezávislé na OS, môžeme nastaviť reláciu, ktorá sa vykoná po nastavení kurzora na daný súbor a stlačení ENTER, resp. inú akciu po stlačení SHIFT ENTER, resp. CTRL ENTER.



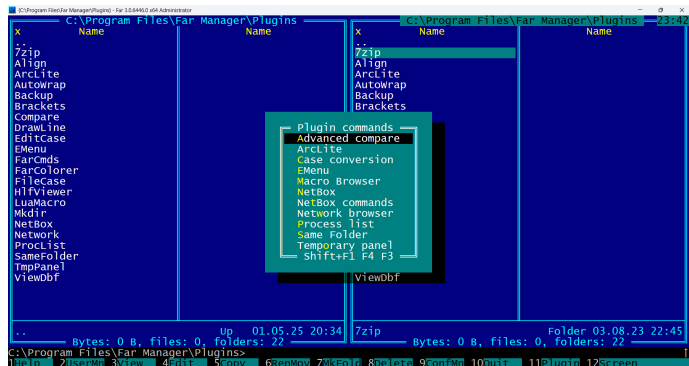
Obr. 3. Rôzne zafarbenia názvov súborov



Obr. 4. Prepínanie medzi viacerými oknami Far Manager-a

Pomocou voľby F9 Option a podvoľby Colors si môžeme nastaviť farby panelov a príkazového riadku podľa vlastných predstáv. Pre rôzne farby zobrazenia súborov podľa koncoviek použijeme Files highlighting and sort groups (obr. 3), súbory môžeme zoradzovať pomocou skratky CTRL F12. Veľkosť systémového okna Far Manager-a si môžeme nastaviť podľa vlastných predstáv, prípadne ho pomocou ALT F9 zväčšiť na celú obrazovku alebo opätovným spustením tejto skratky vrátiť späť. Stlačením F12 môžeme prepínať medzi viacerými oknami Far Manager-a. Na obrázku 4 môžeme prepínať medzi Desktop, panelmi a editorom.

Veľmi účinnou zbraňou Far Manager-a sú plugin-y, ktoré dopĺňajú a rozširujú jeho funkčnosť. Úplná knižnica plugin-ov je k dispozícii na stránke <https://plugin.farmanager.com/index.php?l=en>, kde si môžeme nájsť vhodný plugin



Obr. 5. Tabuľka makier viditeľná po stlačení F11

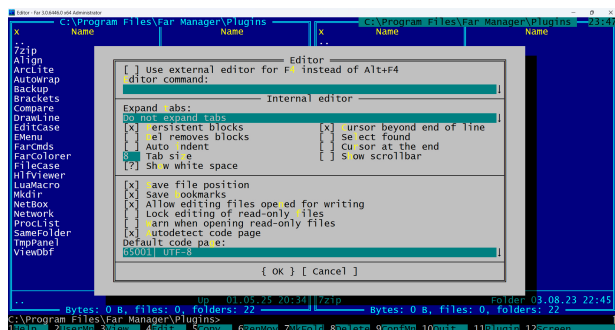
podľa požadovanej funkčnosti, podľa systému, podľa verzie, resp. podľa mena ak ho poznáme. Inštalácia pluginu je jednoduchá, nahráme ho do systémového adresára Far Manager\Plugins a ukončíme Far Manager-a. Po opätovnom spustení je plugin funkčný. Dostupné plugin-y sa zobrazia po stlačení F11 (obr. 5). Zoznam plugin-ov pri spustení F11 z editora a z panela môže byť rôzny. Napr. v paneli môžu byť k dispozícii Advanced Compare, Case conversion (zmena písmen v názvoch na veľké alebo malé), Macro browser, Netbox (prístup na počítače s inými OS), Netbox commands, Network browser (prístup na počítače s OS Windows), Process list, Same folder, Temporary panel a v editore to môžu byť Align block, Draw lines, FarColorer (farebné zvýrazňovanie textu podľa druhu súboru na základe jeho koncovky), Change Case (zmena veľkosti písma v označenom bloku – napr. veľké, malé písmena, iniciálky, cyklická zmena), Macro Browser (zoznam dostupných vlastných makier) ap.

Farebné zvýrazňovanie textu v závislosti od koncovky je možné zmeniť, preddefinovaných je množstvo jazykov a programov (R, Lua, Python, Java, HTML, TeX atď). Vlastné nastavenie je možné uložiť do súboru *.hrc. Napríklad farbenie v L^AT_EX-u mi kontroluje párovanie `begin` a `end` prostredím a párovanie matematického prostredia pomocou dolárov \$, taktiež kontroluje párovanie zložených zátvoriek {}, ktoré sú najčastejšou súčasťou L^AT_EX-ovského zdrojového súboru. Príkazy sa zobrazujú bielou farbou, samostatný text svetlo modrou a matematické prostredie čiernou farbou.

Veľmi užitočným plugin-om je Netbox, pomocou ktorého sa môžeme dostať na internet. Ponúka protokoly SFTP, FTP, SCP, musíme zvoliť názov počítača, na ktorý sa dobíjame a vhodný port, môžeme zvoliť užívateľské meno a heslo, prípadne súbor s privátnym kľúčom. Ak užívateľské meno a heslo nezvolíme systém si ho vypýta, pričom všetky tieto atribúty si môžeme uložiť pod vlastným názvom a spúšťať priamo. Far Manager priamo rozbaľuje komprimované súbory a je možné z nich súbory kopírovať, prípadne mazať. Ja ma prácu s komprimovanými súborami používam navyše plugin 7zip.

2. Editor

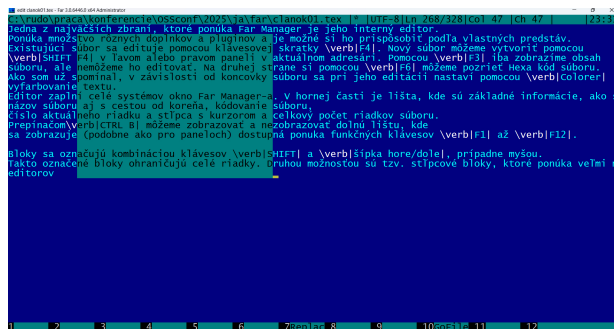
Jedna z najväčších zbraní, ktoré ponúka Far Manager je jeho interný editor. Ponúka množstvo rôznych doplnkov a pluginov a je možné si ho prispôbiť podľa vlastných predstáv. Existujúci súbor sa edituje pomocou klávesovej skratky **F4**. Nový súbor môžeme vytvoriť pomocou **SHIFT F4** v ľavom alebo pravom paneli v aktuálnom adresári. Pomocou **F3** iba zobrazíme obsah súboru, ale nemôžeme ho editovať. Na druhej strane si pomocou **F6** môžeme pozrieť Hexa kód súboru. Ako som už spomínal, v závislosti od koncovky súboru sa pri jeho editácii nastaví pomocou **Colorer** vyfarbovanie textu. Editor zaplní celé systémové okno Far Manager-a. v hornej časti je lišta, kde sú základné informácie, ako sú názov súboru aj s cestou od koreňa, kódovanie súboru, číslo aktuálneho riadku a stĺpca s kurzorom a celkový počet riadkov súboru. Prepínačom **CTRL B** môžeme zobrazovať a nezobrazovať dolnú lištu, kde sa zobrazuje (podobne ako pri paneloch) dostupná ponuka funkčných klávesov **F1** až **F12**.



Obr. 6. Nastavenie editora

Editor si môžeme nastaviť v menu **Options** a voľbe **Editors settings**, sú tam rôzne nastavenia. Namiesto zabudovaného editora si môžeme zvoliť a nastaviť ľubovoľný externý editor, voľba **Show white space** ponúka tri možnosti. Prvá možnosť je, že sa nezobrazujú skryté znaky, pri druhej sa zobrazujú iba prázdne medzery, konce riadkov a konce súborov (voľba s otáznikom na obr. 6), resp. pri voľbe **x** sa zobrazujú všetky skryté znaky a symboly. Medzi týmito voľbami sa prepína klávesom **SPACE**. Medzi ďalšie voľby patrí nastavenie veľkosti tabulátora (na obrázku je nastavených 8 medzier) a potvrdená voľba **Save file position**, ktorá nám zaručuje, že po uzavretí súboru ho editor opätovne otvorí na identickom mieste. Voľba **Autodetect code page** sa snaží automaticky detekovať kódovanie súboru, pričom implicitne **Default code page** je v tomto prípade nastavené kódovanie **UTF8**.

Bloky sa označujú kombináciou klávesov **SHIFT** a šípka hore/dole, prípadne myšou. Takto označené bloky ohraničujú celé riadky. Druhou možnosťou sú tzv. stĺpcové bloky, ktoré ponúka veľmi málo editorov (obr. 7). Tieto bloky sa označujú

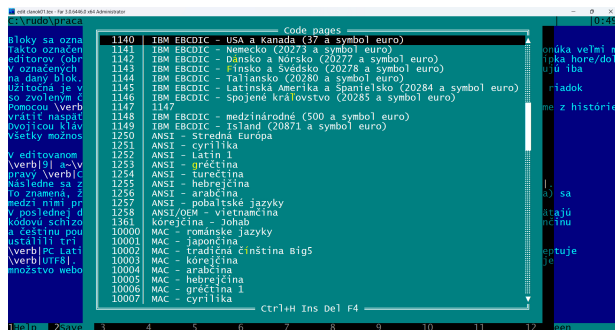


Obr. 7. Interný editor so zvýrazneným stĺpcovým blokom

kombináciou klávesov ALT a šípka hore/dole. V označených blokoch v riadkových aj stĺpcových môžeme spúšťať rôzne plugin-y, ktoré sa aplikujú iba na daný blok. Užitočná je voľba Goto, ktorá sa inicializuje pomocou ALT F8 a ponúka odskok na riadok so zvoleným číslom. Pomocou CTRL Z môžeme vrátiť históriu editovania a pomocou CTRL SHIFT Z sa môžeme z histórie vrátiť naspäť. Dvojicou klávesov SHIFT ENTER vložíme do editovaného súboru jeho názov aj s koncovkou. Všetky možnosti klávesových skratiek dostaneme po spustení nápomoci F1.

V editovanom súbore si môžeme pomocou kombinácie pravého CTRL a čísla od 1 po 9 a 0, napr. pravý CTRL 3 označiť danú pozíciu. Označí sa aktuálna pozícia kurzora. Následne sa z ľubovoľného miesta súboru dostaneme na túto pozíciu stlačením ľavý CTRL 3. To znamená, že si môžeme nezávisle na sebe označiť desať miest v súbore a rýchlo (bez hľadania) sa medzi nimi presúvať. V poslednej dobe sa najčastejšie používa kódovanie UTF8. Starší užívatelia si ešte pamätajú kódovú schizofréniu, kde každý program, systém, užívateľ používal iné kódovanie, aj pre slovenčinu a češtinu používali rôzni užívatelia viacero kódovaní. Na začiatku tisícročia sa v podstate ustálili tri kódovania. Užívatelia Linux-u používalo ISO Latin2 a užívatelia DOS PC Latin2 (CP852) a samozrejme Windows mal a má svoju CP1250, aj keď akceptuje UTF8.

V súčasnosti už nie je problém s kódovaním taký akútny ako kedysi, aj keď stále existuje množstvo webových stránok s exotickým kódovaním. Po voľbe SHIFT F8 sa nám ukáže so všetkými možnými kódovaniami, ktoré sú k dispozícii. Na obrázku 8. Ak otvorený súbor nemáme v správnom kódovaní, potom pomocou tejto tabuľky môžeme zvoliť správne. Veľkou výhodou interného editora je automatické prekódovanie označeného textu. Ak máme napr. súbor v kódovaní CP1250 a chceme ho prekódovať do UTF8. Môžeme použiť nejaký vhodný script, ktorých je mnoho a pomocou neho súbor prekódovať. Druhá možnosť je použiť interný editor Far Manager-a. Otvoríme si daný súbor v kódovaní CP1250 (súbor by sa mal automaticky otvoriť v správnom kódovaní). Skopírujeme celý súbor do schránky pomocou CTRL A a CTRL X, pomocou tabuľky kódovania SHIFT F8 si zvolíme kódovanie UTF8 a vložíme obsah schránky pomocou CTRL V naspäť. Pri



Obr. 8. Tabuľka možných rôznych kódovaní editora

tomto postupe sa text v schránke automaticky prekóduje a vloží sa v správnom kódovaní.

3. Bez záveru

Far Manager dokáže toho samozrejme oveľa viac a na predstavenie všetkého, čo dokáže a čo z toho využívam nie je čas. Ja ho používam už viac ako 25 rokov a stále som s ním spokojný. Niekedy sa človek nevyhne použitiu iných správco, ale tam našinec nemá súbory pod takou kontrolou ako by si predstavoval. Aj keď mladá generácia odchovaná na myškách je s nimi spokojná. Iba som ukázal, že existuje aj iná, podľa môjho názoru, lepšia bezplatná alternatíva.

Poďakovanie. Príspevok vznikol s príspevom grantu KEGA 019ŽU-4/2023 „Inovatívne učenie matematiky s podporou Open Source“ podporeného Slovenskou kultúrno-edukačnou grantovou agentúrou.

Literatúra

- [1] Far Manager. <https://www.farmanager.com/>.
- [2] Far Manager. <https://github.com/FarGroup/FarManager/>.
- [3] Far Manager. https://en.wikipedia.org/wiki/Far_Manager, https://sk.wikipedia.org/wiki/FAR_Manager, https://cs.wikipedia.org/wiki/Far_Manager.

Kontaktná adresa

RNDr. Rudolf Blaško, PhD., Katedra matematických metód, Fakulta riadenia a informatiky, Žilinská univerzita, Univerzitná 8215/1, 010 26 Žilina, Slovensko, *Aktuálna adresa:* SOIT, 010 01 Žilina, Slovensko,
E-mailová adresa: beerb@frcatel.fri.uniza.sk, <http://frcatel.fri.uniza.sk/~beerb/>

MOODLE A MATEMATIKA

RUDOLF BLAŠKO (SK)

Abstrakt. Moodle je asi najznámejší a v akademických kruhoch najpoužívanější výučbový systém a to aj napriek tomu, že je Open Source a teda zadarmo. Umožňuje rôzne organizovať výučbu, umiestňovať študijné materiály, vytvárať slovníky a iné učebné pomôcky, vytvárať ankety, dotazníky a testy z rôznych typov úloh a následne ich automaticky vyhodnocovať. Tento príspevok sa venuje matematickej podpore Moodle a modulom, ktoré umožňujú nielen jednoduchý zápis matematických vzťahov pomocou systému $\text{\LaTeX}$, ale aj testovanie matematických výpočtov pomocou klasických testov a tiež pomocou symbolických výpočtov.

Kľúčové slová. Moodle, MathJax, Stack, $\text{\LaTeX}$, Maxima, test, matematika.

MOODLE AND MATHEMATICS

Abstract. Moodle is probably the best-known and most widely used learning system in academic circles, despite the fact that it is Open Source and therefore free. It allows you to organize teaching in various ways, place study materials, create dictionaries and other teaching aids, create polls, questionnaires and tests from various types of tasks and then automatically evaluate them. This article deals with Moodle's mathematical support and modules that allow not only simple writing of mathematical relations using the $\text{\LaTeX}$ system, but also testing mathematical calculations using classical tests and also using symbolic calculations.

Keywords. Moodle, MathJax, Stack, $\text{\LaTeX}$, Maxima, test, mathematics.

Úvod

Moodle (angl. Modular Object-Oriented Dynamic Learning Environment) je bezplatný Open Source systém pre tvorbu a správu výučbových systémov a elektronických kurzov na webe. Systém Moodle je distribuovaný pod licenciou GNU General Public License. Moodle je najrozšírenejšou vzdelávacou platformou na svete. Moodle sa používa na vytváranie vlastných webových stránok s online kurzami a umožňuje používanie rôznych doplnkov.

Moodle pôvodne vyvinul Martin Dougiamas, aby pomohol pedagógom a akademickým pracovníkom vytvárať online kurzy a zamerať sa na interakciu a spoluprácu. Prvá verzia Moodlu bola vydaná 20. augusta 2002. Odvtedy sa tento systém sústavne vyvíja a vylepšuje o nové prvky a nástroje. Cieľom Moodle je poskytovať učiteľom, správcom a študentom bezpečný a integrovaný systém vytvárajúci personalizované vzdelávacie prostredie.

Projekt Moodle vedie a koordinuje austrálska spoločnosť Moodle HQ, ktorú finančne podporuje sieť partnerských servisných spoločností Moodle po celom svete. Na vývoji sa tiež nemalou mierou podieľa Open Source komunita. K dispozícii sú plugíny, vlastné grafické témy, responzívny webdizajn pre mobilné zariadenia a mobilná aplikácia Moodle, ktoré umožňujú prispôbiť používateľské prostredie každej osoby na platforme. Moodle funguje bez problémov a bez úprav prakticky na všetkých OS, ktoré podporujú PHP a databázy, vrátane poskytovateľov webhostingu (Linux, BSD, Windows, Mac OS, Android, iPadOS...).

Výučba v prostredí Moodle prebieha v tzv. kurzoch, ktoré vytvárajú učitelia pomocou modulov. Moodle ponúka štandardne niekoľko modulov, pomocou ktorých je možné bez problémov budovať takéto kurzy. U nás na univerzite sa jednotlivé kurzy volajú ako vyučované predmety a pred semestrom ich prepravia Centrum informačných a komunikačných technológií (CIKT) pre celú školu. Samozrejme ich nenapĺňa študijnými materiálmi, ale priradí učiteľov a študentov k daným kurzom (zapísaným predmetu). Všetko ostatné si musí učiteľ zorganizovať a nakrmiť materiálmi sám. V štandardne dodávaných moduloch môžu učitelia v rámci vytvorených online kurzov pridávať študijné materiály vo forme HTML, pridávať súbory a texty na stiahnutie. Problém je s hromadným nahrávaním súborov, pretože sa nahrávajú samostatne. Ak chcete nahrávať viac súborov, napr. každému zo 150 študentov vlastné zadanie, je to nadľudský výkon. To je hlavný dôvod, prečo nezverejňujem študentom študijné materiály v Moodle, ale na svojej domovskej stránke (ich podstránkach), kde mám v Open Source File Manager pomocou SFTP bezproblémový prístup. Ďalej môže učiteľ tvoriť a hodnotiť rôzne testy, zadania a úlohy, vytvárať ankety a dotazníky, zakladať diskusné fóra a odpovedať na otázky účastníkov ap.

Moodle je vcelku užívateľsky príjemný systém, teda pre študentov. Pre učiteľov je to s tým komfortom trochu slabšie. Veľká väčšina úloh a situácií je bezproblémových, ale existuje zopár zvláštnosti a pre mňa nelogických prepojení a nastavení. Na našej univerzite sa Moodle používa už od mojej nepamäti (15–20 rokov), aktuálne je nainštalovaná verzia 4.5.4. Ja osobne Moodle používam iba na pravidelné rýchle testovanie študentov na predmete Matematická analýza. Študentov na tejto platforme sme testovali už dva roky po sebe. Pred viac ako dvomi rokmi nebol problém s testovaním, ale bol problém s matematickým zápisom úloh. Myslí, že to bola verzia 4.1.?, ale to je v tomto čase irelevantné, pretože už nie sú podporované ani verzie 4.4.? V danom čase by sa museli do Moodle nainštalovať moduly MathJax a STACK, čo nebol v spolupráci s našim školským CIKT triviálny problém. Našťastie nasledujúce verzie 4.4.? už mali tento modul v sebe automaticky implementovaný. V budúcnosti plánujeme testovať aj postupy riešení daných matematických príkladov a problémov. Toto je taktiež v posledných verziách Moodle možné a rieši sa to pomocou modulu STACK, ktorý dovoľuje symbolické a numerické výpočty, pričom na pozadí využíva ďalší Open Source algebraický výpočtový systém Maxima.

1. MathJax

MathJax je JavaScript knižnica pre zobrazovanie matematických vzťahov a výrazov vo webových prehliadačoch pomocou $\text{\LaTeX}$ -u a MathML. MathJax je Open Source distribuovaný pod licenciou Apache license. MathJax sa začal vyvíjať v roku 2009 ako nástupca starej knižnice JavaScript pre zobrazovanie matematických vzorcov jsMath. Projekt MathJax založili AMS (American Mathematical Society), Design Science a Society for Industrial and Applied Mathematics. Projekt spravuje AMS a podporujú ho mnohí sponzori ako sú napríklad American Institute of Physics a Stack Exchange. MathJax používajú rôzne webové portály (MathSciNet, GitHub, n-category cafe, MathOverflow, Scholarpedia...) a podporujú ho prakticky všetky webové prehliadače. Niektoré staršie verzie prehliadačov síce fungujú, ale nepodporujú webové písma a potom MathJax nahradí matematický text obrázkami. MathJax dokáže spracovať príkazy v matematickom prostredí $\text{\LaTeX}$ -u, navyše sú podporované matematické príkazy z balíčkov AMS (amsmath, amscd, amssymb, amsfonts). MathJax je možné bez problémov doinštalovať do rôznych publikačných systémov (napr. MediaWiki, Drupal, WordPress a Joomla).

Autor príspevku používa MathJax na svojich webových stránkach už skoro 15 rokov (napr. <https://frcatel.fri.uniza.sk/users/beerb/ma1/prednasky/basic-integrals.htm>). Ja svoje webové stránky nezverejňujem cez žiadny publikačný systém, takže si môžem ich disajn bez problémov nastaviť podľa vlastných predstáv. Aby sa mohol MathJax na stránke aplikovať, musí sa na prenos stránky použiť protokol `https`, pričom `http` už niekoľko rokov nefunguje. Navyše musí mať súbor v hlavičke konfiguračný odkaz na nejaký server MathJax, napr.:

```
<HEAD>
<script type="text/javascript"
  src="https://cdn.mathjax.org/mathjax/latest/MathJax.js?config=TeX-AMS-MML_HTMLorMML">
  MathJax.Hub.Config({tex2jax: {inlineMath: [ ['$','$'], ["\\(", "\\)"] ],
    displayMath: [ ['$$','$$'], ['\[','\]'] ] }});
</script>
</HEAD>
```

Matematický text sa sádza klasicky medzi dva doláre \$ \$, napr. [1]:

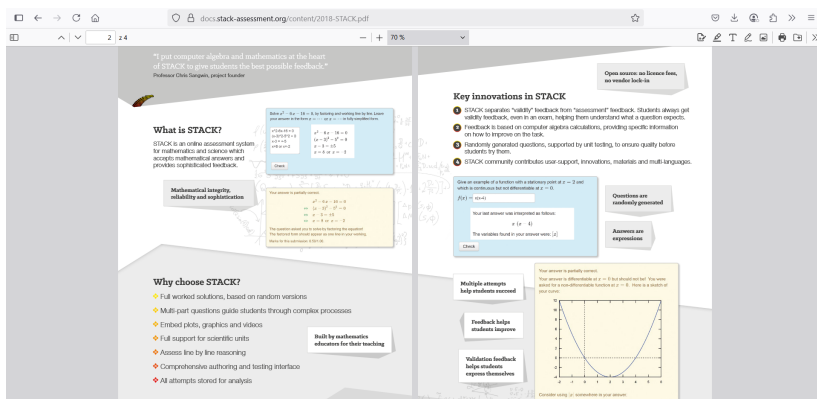
```
{\displaystyle\int\frac{\mathrm{d}x}{\sqrt{a^2-x^2}}=\arcsin{\frac{x}{|a|}}+c_1
=-\arccos{\frac{x}{|a|}}+c_2$}
```

a výsledok má tvar $\int \frac{dx}{\sqrt{a^2-x^2}} = \arcsin \frac{x}{|a|} + c_1 = -\arccos \frac{x}{|a|} + c_2$.

Pri použití v Moodle sa musí matematický text dávať medzi dvojicu dvoch dolárov \$\$ \$\$ (ako samostatné rovnice v $\text{\LaTeX}$ -u), resp. medzi ich ekvivalent $\left(\right)$. Matematický text sa vysádza klasicky do riadku ako v prostredí \$ \$.

Pri našom testovaní študentov sme vystačili s modulom Mathjax, pretože sme ich netestovali pomocou výpočtov ale iba pomocou výberu správnej odpovede (ale zo šiestich až desiatich možností). Symbolické a numerické výpočtové skúšanie

umožňuje STACK Online assessment, ktorý je taktiež implementovaný v posledných verziách Moodle. Modul STACK využíva na svoje výpočty Open Sorce program Maxima (Computer Algebra System Maxima). Symbolické online testovanie študentov pomocou STACK je v pláne a s tým zatiaľ nemám praktické skúsenosti. Praktické skúsenosti mám s online testovaním pomocou formulárov vytvorených v AcroTeX-u, ale to je iná kapitola a nepatrí do tohto príspevku. Na záver tejto časti uvádzam náhľad stránky <https://docs.stack-assessment.org/content/2018-STACK.pdf>, kde je zobrazený formulár profesora Chrisa Sangwina z University of Edinburgh, UK (obr. 1).

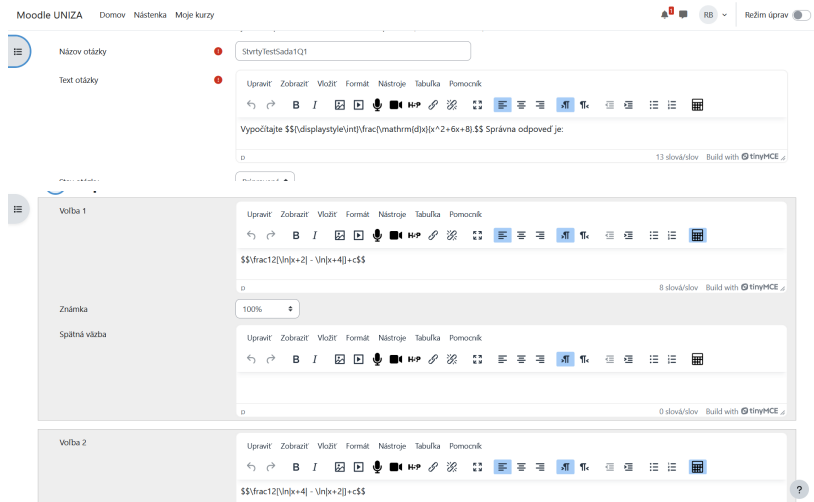


Obr. 1. STACK formulár, zdroj <https://docs.stack-assessment.org/content/2018-STACK.pdf>.

2. Testovanie študentov z matematiky

Na predmete Matematická analýza sa vyučuje základná vyššia matematika vrátane diferenciálneho a integrálneho kalkulu. z uvedeného je zrejmé, že zadávať testové otázky opisne bez možnosti matematického zápisu je nielen nereálne ale aj úplne nezmyselné. Existuje možnosť generovať jednotlivé matematické vzorce do nejakého vektorového obrázka (svg, png ap.) a zobrazovať tieto obrázky. Ale takýto postup je nesystematický a hlavne zdĺhavý a prácny. Problém vyriešil Moodle implikáciou modulov MathJax a STACK do svojho prostredia. My sme študentov hodnotili pomocou Moodle 4-krát počas semestra a 2-krát sme ich hodnotili klasickou písomnou previerkou na papier, kde mali vypočítavať v stanovenom čase 45–50 minút niekoľko úloh.

Krátky test v prostredí Moodle bol nastavený na 10 minút pre každého študenta, pričom test bol otvorený 20 minút, aby mal študent čas stihnúť svoj desťminutový limit. Výsledky sa študent dozvedel po uzavretí test, dozvedel sa iba celkový výsledok, ale nie body za jednotlivé príklady, ktoré boli 4. Takto to bolo nastavené preto, lebo študenti celého kurzu (prakticky celý ročník) absolvovali



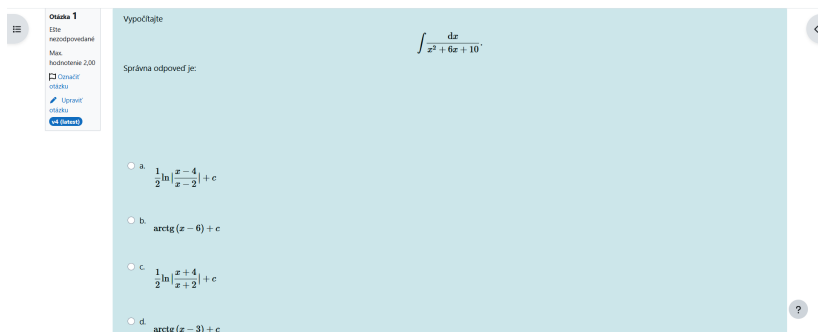
Obr. 2. Ukážka tvorenia otázky z krátkeho online testu v Moodle

rovnaký test počas stanoveného týždňa na cvičeniach. Termíny otvorenia a uzavretia testu nastavoval podľa potreby cvičiaci učiteľ. Termíny všetkých testov sa študenti dozvedeli na začiatku semestra, tieto termíny sa počas semestra z objektívnych dôvodov mohli posunúť o plus mínus týždeň.

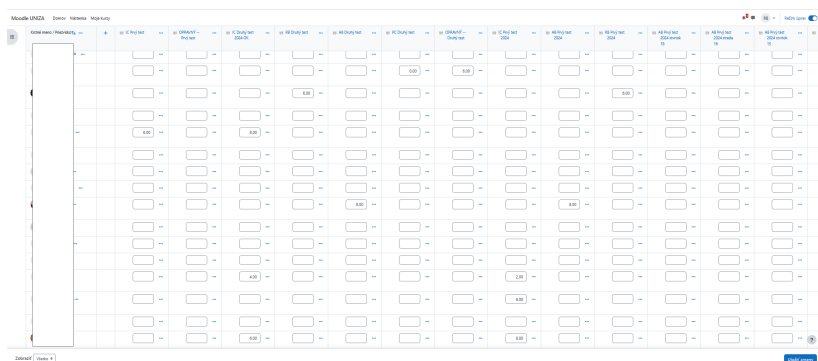
Podmienka absolvovania testu bola fyzická prítomnosť študenta na cvičení a vlastný počítač, notebook, mobil, na ktorom mohol tento test realizovať. Test sa skladal zo 4 príkladov z vopred dohodnutých okruhov po dvoch bodoch. Počítali sa všetky body z týchto štyroch online testov a aj oboch papierových testov a na postup ku skúške musel študent získať aspoň 60 % všetkých bodov. Ku každému testu boli pripravené 4 sady po 6 až 8 príkladov. Ku každému z týchto príkladov bolo priradených aspoň 6 odpovedí, pričom práve jedna bola správna (obr. 2 a náhľad uvedeného príkladu obr. 3). Moodle vytvoril každému študentovi unikátny test tak, že z každej sady vybral náhodne jeden príklad a jeho odpovede zamiešal. Taktiež Moodle zamiešal v tomto teste poradie príkladov. Vzhľadom na krátkosť času testu a spôsob jeho vygenerovania, nebolo potrebné študentom dávať dodatočné obmedzujúce podmienky na test.

Pri vyplňaní testu môže študent riešiť otázky v ľubovoľnom poradí a môže test hocikedy uzavrieť. Pokiaľ študent nestihne v časovom limite vyplniť všetky otázky, test sa po uplynutí limitu automaticky uzavrie sám. Časť výsledného známkového výkazu za semester je znázornená na obrázku 4. Moodle spočítal výsledky študentov.

Poďakovanie. Príspevok vznikol s príspevom grantu KEGA 019ŽU-4/2023 „Inovatívne učenie matematiky s podporou Open Source“ podporeného Slovenskou kultúrno-edukačnou grantovou agentúrou.



Obr. 3. Náhľad vygenerovanej otázky z obr. 2 v hotovom teste



Obr. 4. Náhľad známkového výkazu absolvovaných testov počas semestra

Literatúra

- [1] Table of basic integrals. <https://frcatel.fri.uniza.sk/users/beerb/ma1/prednasky/basic-integrals.htm>.
- [2] STACK. <https://stack-assessment.org/>, <https://stack-assessment.org/About/>.
- [3] Moodle community. <https://moodle.org/?lang=sk>.
- [4] Moodle. <https://moodledev.io/general/releases/4.4/4.4.7>.
- [5] Moodle. <https://en.wikipedia.org/wiki/Moodle>.
Moodle MathJax filter. https://docs.moodle.org/500/en/MathJax_filter.
- [6] MathJax. <https://www.mathjax.org/>.
- [7] MathJax. <https://en.wikipedia.org/wiki/MathJax>.

Kontaktná adresa

RNDr. Rudolf Blaško, PhD., Katedra matematických metód, Fakulta riadenia a informatiky, Žilinská univerzita, Univerzitná 8215/1, 010 26 Žilina, Slovensko, *Aktuálna adresa*: SOIT, 010 01 Žilina, Slovensko,

E-mailová adresa: beerb@frcatel.fri.uniza.sk, <http://frcatel.fri.uniza.sk/~beerb/>

TikZ A ZOPÁR NÁPADOV

RUDOLF BLAŠKO (SK)

Abstrakt. Príspevok nemá za cieľ učiť čitateľa základné funkcionality TikZ-u, predpokladajú sa minimálne znalosti práce s TikZ-om. Cieľom je ukázať niektoré nápady na kreslenie obrázkov a ich automatizovanie pomocou jednoduchých makier, cyklov a zovšeobecnení. V príspevku je ukázaná tvorba jednoduchých schém, ktoré s obľubou kreslia tvorcovia rôznych modelov a algoritmov. Pre milovníkov teórie grafov je v príspevku uvedený jeden z možných návodov ako vykresliť úplný graf. Príspevok končí názornou ukážkou použitia knižnice TikZ-u, pomocou ktorej sa dajú nájsť priesečníky rôznych kriviek.

Kľúčové slová. $\text{\LaTeX}$, TikZ, diagram, graph, intersection of curves.

TIKZ AND SOME IDEAS

Abstract. The article does not aim to teach the reader the basic functionalities of TikZ, minimal knowledge of working with TikZ is assumed. The aim is to show some ideas for drawing images and automating them using simple macros, cycles and generalizations. The article shows the creation of simple diagrams that are popularly drawn by creators of various models and algorithms. For lovers of graph theory, the article provides one of the possible instructions on how to draw a complete graph. The article ends with a demonstration of the use of the TikZ library, which can be used to find intersections of various curves.

Úvod

Na kreslenie obrázkov, schém, grafov matematických funkcií existuje mnoho viacmenej dobrých programov. Niektoré dokážu zobrazovať 2D aj 3D objekty, niektoré nie, niektoré dokážu vizualizovať pripravené zadané dáta, niektoré ich dokážu aj spracovať a následne zobrazovať, niektoré dokážu vykresľovať manažérske a programátorské schémy a vývojové diagramy. Niektoré používajú formát bitmapa a niektoré sú orientované na vektorovú grafiku. Takto by som mohol pokračovať ešte dlho, ale dôležité je, že so všetkými spomenutými činnosťami si poradí vektorový grafik TikZ. Spolu s balíčkami `pgf*` a tlačiarňou `pgfplot` predstavuje veľmi silnú podporu pre štatistické spracovanie a vizualizáciu údajov.

TikZ sa síce načítava ako štandardné balíčky pomocou príkazu `\usepackage`, ale skôr by sa dal charakterizovať ako účinná grafická nadstavba nad $\text{\LaTeX}$ -om. Ak používame $\text{\LaTeX}$ na vlastnom počítači (štandardne sa nainštaluje aj celá dokumentácia ku všetkým súčastiam $\text{\TeX}$ -u), manuál si zobrazíme na príkazovom riadku spustením `texdoc tikz` (vo všetkých OS). Ak dokumentáciu

nemáme k dispozícii, môžeme sa obrátiť na domovský T_EX-ovský portál `ctan`. Uvedený dokument nájdeme na adrese <https://www.ctan.org/pkg/pgf> alebo samozrejme aj inde. Manuál TikZ-u stále s novými verziami priberá strany a pri aktuálnej verzii 3.1.10 má 1321 strán, čo tiež o niečom svedčí. Napríklad manuál verzie 1.18 z roku 2007 mal 405 strán (viď <https://www.bu.edu/math/files/2013/08/tikzpgfmanual.pdf>),

Veľmi dobrý pomocník pre prácu s TikZ-om, ktorý vytvoril Jean Pierre Castelyn v roku 2018 (TikZ a aj T_EX a L^AT_EX sú nadčasové záležitosti a aj staršie príručky a návody, aj tridsaťročné, majú čo povedať aj dnes) a nazýva sa `Visual TikZ`. Nájdeme ho na už spomínanom portáli `ctan` na adrese <https://mirrors.nic.cz/tex-archive/info/visualtikz/VisualTikZ-fr.pdf> alebo tiež na adrese <https://invimath.fri.uniza.sk/images/slides/LaTeX/VisualTikZ.pdf>. Tento dokument je písaný po francúzsky, ale to neprekáža ani našincovi neznalému reči, pretože je tam množstvo praktických ukážok, návodov a hlavne rôznych nastavení parametrov ku príkazom TikZ-u.

1. Krátky príklad z matematickej teórie grafov

Projekt TikZ, presnejšie povedané PGF/TikZ sa vyvja od roku 2005 a stále sa zdokonaľuje. Už dospel do štádia, že TikZ môžeme dať ako parameter do preambuly T_EX-ovského príkazu `\documentclass`, ale s triedou `standalone`, napr.

```
\documentclass[tikz,border=3mm]{standalone}
```

Je zrejmé, že parameter `border` definuje okraje dokumentu, ktorý bude vždy tvoriť iba jedna strana i keď divných rozmerov. Keď takto vytvoríme niekoľko obrázkov, zobrazia sa vedľa seba. Ak ich chceme zobrazit pod sebou, musíme situáciu riešiť pomocou tabuľky alebo inou násilnou formou.

Nasledujúci zdrojový kód ukazuje ako prakticky na dvoch riadkov vygenerovať úplný graf s n vrcholmi. Graf vygeneruje makro `\CompletGraph` a výsledok je zobrazený na obr. 1. Základom sú dva do seba vložené cykly `\foreach`, v prvom s premennou `\x` sa generuje n vrcholov a v druhom s premennou `\y` sa vygenerujú všetky hrany grafu.

```
\documentclass[tikz,border=3mm]{standalone}
\usetikzlibrary{arrows.meta}
\begin{document}
\newcommand\twodigits[1]{\ifnum#1<10 0#1\else #1\fi}
\colorlet{FaI}{black}    %% FaI nastavenie farby zobrazenia

%% definicia makra \UplnyGraf .. #1 pocet vrcholov %%
\def\UplnyGraf#1{\begin{tikzpicture}[rbA/.style={thick,FaI,fill=FaI!20,
draw,circle,minimum size=1.8em,inner sep=.1em,scale=.6}
,rbH/.style={FaI!80,line width=.5pt,{Stealth[scale=.6]}--{Stealth[scale=.6]}}]
%% generovanie vlastneho uplneho grafu %%
\foreach \x in {1,...,#1}
{ \node[rbA] (g\x) at (90-\x*360/#1:2cm) {\bf\twodigits{\x}};
  \foreach \y in {1,...,\x} {\ifx\x\y \else\path (g\x) edge[rbH] (g\y);\fi}}
```

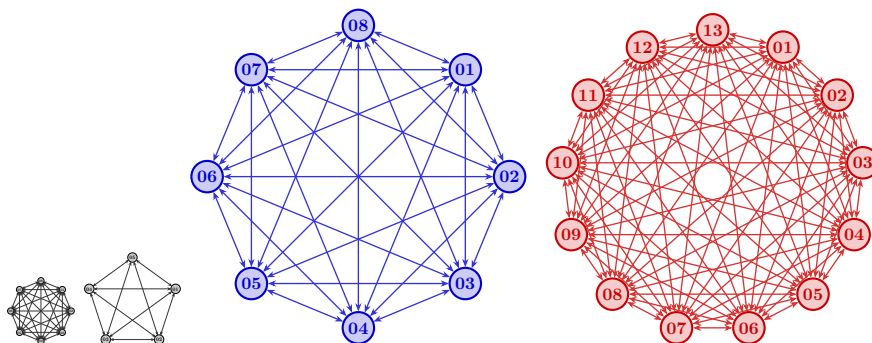
```

\end{tikzpicture}}

%% spustanie makra \UplnyGraf s roznymi parametrami %%
\scalebox{.2}{\UplnyGraf{8}}      %% L cierny graf n= 8 skala=20%
\scalebox{.3}{\UplnyGraf{5}}      %% LS cierny graf n= 5 skala=30%
\colorlet{FaI}{blue!80!black}\UplnyGraf{8}    %% PS modry graf n= 8
\colorlet{FaI}{red!80!black} \UplnyGraf{13}    %% P cerveny graf n=13
\end{document}

```

Keďže sa v makre kreslia medzi vrchmi grafu krajšie šípky **Stealth** ako sú štandardné, je v preambule zdrojového súboru nevyhnutné načítať knižnicu `\usetikzlibrary{arrows.meta}`. Tieto šípky sú definované v TikZ-ovskom štýle `rbH` a sú nastavené na orientovaný graf, ak chceme graf neorientovaný, stačí tento štýl prepísať na tvar `rbH/.style={FaI!80,line width=.5pt,-}`.



Obr. 1. Generovanie úplného grafu s rôznym počtom vrcholov

Zväčšenie a zmenšenie grafu je riešené pomocou príkazu `\scalebox{}{}`, ktorý má dva parametre. Prvý parameter určuje násobok zmeny veľkosti a druhý parameter obsahuje objekt tejto zmeny (napr. `.25`, resp. `0.25` predstavuje zmenšenie na 25 % pôvodnej veľkosti a parameter `4` znamená štvornásobné zväčšenie).

Vrcholy grafu sú označené vždy dvomi ciframi, čo rieši makro `\twodigits`, ktoré je definované príkazom:

```
\newcommand\twodigits[1]{\ifnum#1<10 0#1\else #1\fi}
```

Pred spustením makra `\UplnyGraf` je potrebné nejakým farbiacim príkazom definovať farbu zobrazenia `FaI`, napr. príkazom `\colorlet`, ktorý je definovaný v balíčku `xcolor`. Balíček `xcolor` nie je potrebné načítavať zvlášť, TikZ si ho aktivuje sám. Uvedené makro má iba jeden parameter, ktorý určuje počet vrcholov a musí byť prirodzené číslo. Jeho hodnota sa nikde netestuje a môže byť aj `#1=1`. Vtedy sa zobrazí iba jeden vrchol $\textcircled{01}$. Ak chceme zmeniť číslovanie na kladný smer, stačí prepísať v riadku

```
{\node[rbA] (g\x) at (90-\x*360/#2:3cm) {\bf\twodigits{\x}$};
```

zátvorku `(90-\x*360/#2:3cm)` na tvar `(90+\x*360/#2:3cm)`. Pre nastavenie `(0+\x*360/#2:3cm)` sa číslovanie začne od 3. hodiny.

Pred vykresľovaním hrán sa testuje príkazom `\ifx \else \fi`, či sú vrcholy rôzne alebo nie. Hrana sa vykreslí iba v zápornom prípade pomocou príkazu:

```
{\ifx\x\y \else\path (g\x) edge[rbH] (g\y);\fi}
```

Namiesto predchádzajúceho príkazu môžeme s rovnakým efektom použiť príkaz `{\ifx\x\y \else\draw[rbH] (g\x)--(g\y);\fi}`, pričom $(g\backslash x)$ a $(g\backslash y)$ sa postupne nahrádzajú vrcholmi $g_1, g_2, \dots$.

2. Podpivníky a schémy

Pred niekoľkými rokmi som kreslil modely, ktoré pozostávali z niekoľkých poprepletaných kruhov, ktoré obsahujú jednotlivé pojmy. Pokúsil som sa tieto schémy zautomatizovať na ľubovoľný počet vrcholov s možnosťou jednoduchých zmien vnútorných textov a výsledkom je makro `\Podpivnik`.

Už dlhšie majú slovenčina a čeština niekedy problém s expandovaním pomlčky - v makrách. Používateľ $\text{\LaTeX}$ -u sa s tým najčastejšie stretne pri počiarkovaní v tabuľkách `tabular` pomocou `\cline`. Riešením je zadanie príkazu

```
\shorthandoff{-}
```

hneď za začiatok dokumentu, t. j. za príkaz `\begin{document}`. Toto je aj náš prípad. `TikZ` nevyžaduje žiadnu dodatočnú knižnicu.

Na písanie textov vo vnútri pivných tácoch som použil sanserifové písmo, ktoré je vhodnejšie na incidenčné materiály. Ak budeme takéto písmo používať v celom dokumente, môžeme ho nastaviť v preambule zdrojového dokumentu príkazom

```
\renewcommand*\familydefault{\sfdefault}
```

V tomto prípade sa nám príkazom `{\bf text}`, zobrazí **text**. Ak takéto písmo nepoužívame v celom dokumente a uvedený príkaz v preambule nie je, t. j. je nastavené klasické pätkové písmo, uvedeným príkazom dostaneme **text**. Ak chceme použiť tučné sanserifové písmo, môžeme použiť dvojicu prepínačov `\sf\bfseries`. Týmto spôsobom je riešené písmo v nasledujúcej schéme `\Podpivnik` pomocou makra `\def\bfs{\sf\bfseries}`. V nasledujúcich obrázkoch je použitá iná možnosť pomocou makra `\def\bfs{\bf\sfseries}`.

`\Podpivnik` je konštruovaný tak, aby sa dal ľubovoľne zmenšovať a zväčšovať. Zmena veľkosti sa zadáva ako prvý parameter škála **#1**. Rozmery v prostredí `tikzpicture` sú implicitne nastavené na **cm**, ale hrúbky čiar na body **pt**. Zmenou parametra `scala` sa zmení veľkosť obrázka, ale nezmení sa hrúbka čiar. To znamená, že pri veľkom zmenšení je čiara prihrubá a pri veľkom zväčšení naopak pritenká. Tento parameter **#1** sa hneď na začiatku makra `\Podpivnik` nastaví pomocou príkazu `\def\Psc{#1}` do parametra `\Psc`, ktorým sa násobia všetky nastavenia čiar. Je tu ešte jeden problém s textom, kde musí byť škála tiež násobená týmto parametrom, pretože škála nastavená na začiatku prostredia `tikzpicture` sa ignoruje v texte definovanom za `node`. To znamená, že týmto parametrom sú

vynásobené aj všetky škály v celom prostredí `tikzpicture` a veľkosť obrázku sa reguluje iba týmto parametrom.

Pred spustením vlastného makra `Podpivnik` musia byť definované všetky texty, ktoré sa použijú. Definujú sa v parametroch `\PP*`. Podobne, ako pri predchádzajúcom príklade, musíme pred spustením definovať farbu zobrazenia `FaI`.

```
%% text do stredneho kruhu a po okrajoch velkeho kruhu %%
\def\PPc{\begin{tabular}{@{}c@{}}OSS Conf\ 2025\ Žilina\end{tabular}}
\def\PPe{} \def\PPn{UTOROK}
\def\PPw{} \def\PPs{1.\,7.\,2025}
%% text do jednotlivych vnutornych kruhov ... pre n>10 treba dalsie \PPxi atd %%
\def\PPi{čapuje Rudo} \def\PPii{čapuje Aleš\ čapuje Milan\ čapuje Karol}
\def\PPiii{čapuje Pavel} \def\PPiv{čapuje Peter}
\def\PPv{čapuje Miro} \def\PPvi{čapuje Tomáš}
\def\PPvii{čapuje Milo} \def\PPviii{čapuje Michal}
\def\PPix{dnes sa nečapuje} \def\PPx{dnes sa nečapuje}
%% definicia pisma \bfs %% farby zobrazenia TaI %% pocneho textu \Poo %%
\def\bfs{\sf\bseries} \colorlet{FaI}{blue!60!black} \def\Poo{Pult~}
```

Druhý parameter `#2` nastavuje počet vnútorných kruhov, ak zvolíme počet väčší ako je 10, musíme dodefinovať aj textové zásobníky `\PP*` do požadovaného počtu. Makro funguje tak, že sa v cykle `\foreach` načítavajú postupne čísla od 1 po `#2`, arabské čísla sa premenia na malé rímske čísla, pridá sa prefix `PP` a celé sa do zloží do príslušného názvu príkazu `\PP*`, ktorý sa načíta do parametra `\PQs` a ten sa následne vytlačí pomocou `node`. Celý text sa tvorí konštrukciou

```
\def\PQs{\phantom.\smash{\begin{tabular}[#3]{@{}c@{}}
\scalebox{1.5}{\color{FaI}\Poo\MakeUppercase{\romannumeral\xi}}\
\csname PP\romannumeral\xi\endcsname \end{tabular}}\phantom.}
```

príčom `\csname PP\romannumeral\xi\endcsname` postupne vyrobí `\PPi`, `\PPii`, atď., a `\Poo\MakeUppercase{\romannumeral\xi}` vytvorí texty **Pult I**, **Pult II**...

Parametre `#3` a `#4` definujú zarovnanie textov v jednotlivých kruhoch navzájom (t zarovnané na vzdialený okraj od stredu, b zarovnanie na najbližší okraj od stredu, resp. c zarovnané na stred výšky textu) a vzdialenosť textu od stredu. Overené nastavenia týchto parametrov `[#3|#4]` sú `[b|.23]`, `[c|.31]`, resp. `[t|.38]`. Cyklus `\foreach` je použitý dvakrát z praktických dôvodov. Najskôr sa vykreslia jednotlivé kruhy aj s výplňou, aby potom netienili vlastné texty.

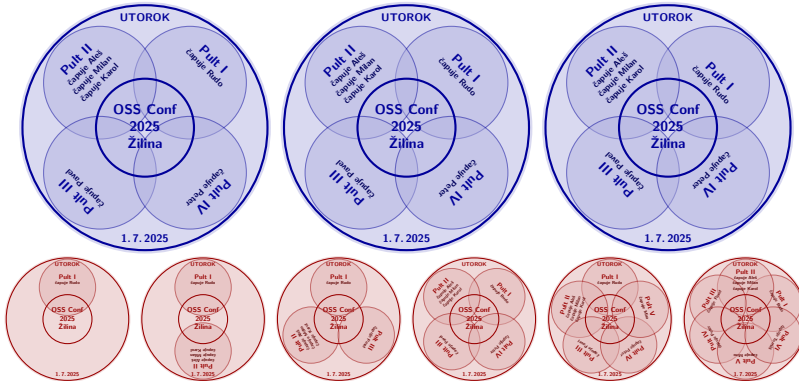
```
%% #1..skala -- zmensenie/zvacsenie #2..pocet vnutornych kruhov
%% #3..btc..pozicia textu od stredu #4..vzdialenost textu od stredu
\def\Podpivnik#1#2[#3|#4]{\def\Psc{#1}\colorlet{FaI}{black!40!FaI}%
\def\PPo{\ifodd#2 0\else 1\fi}\def\PPoo{\ifodd#2 90\else 0\fi}%
\begin{tikzpicture}[FaI,scale=.6*\Psc,rbS/.style={scale=.75*\Psc}
,rbDm/.style={line width=2*\Psc pt},rbDv/.style={line width=4*\Psc pt}
,rbFi/.style={FaI!15,fill},rbFiii/.style={FaI,fill=FaI!30,line width=1*\Psc pt}]
\draw[rbFi,rbDv] (0,0) circle (.5\textwidth+2pt);
\draw[rbDm] (0,0) circle (.5\textwidth);
\foreach \xi in {1,...,#2}
{ \draw[rbFiii,opacity=.5]
({\PPoo+\PPo+180/#2+360*(\xi-1)/#2}:.26\textwidth) circle (.23\textwidth);}
\foreach \xi in {1,...,#2}
```

```

\def\PQs{\phantom.\smash{\begin{tabular}[#3]{@{}c@{}}
\scalebox{1.5}{\color{FaI}\Poo\MakeUppercase{\romannumeral\xi}} \\\
\cename PP\romannumeral\xi\endcsname \end{tabular}}\phantom.}
\draw ({\PPoo+\Ppo*180/#2+360*(\xi-1)/#2}:#4\textwidth)
node[FaII,rbS,rotate={\PPoo+\Ppo*180/#2+360*(\xi-1)/#2-90}] {\bfs\PQs};}
\draw[rBm] (0,0) circle (.2\textwidth) node[scale=1.2*\Psc] {\bfs\PPc};
\draw (0:.45\textwidth) node[scale=.95*\Psc,rotate=-90] {\bfs\PPe};
\draw (90:.45\textwidth) node[scale=.95*\Psc] {\bfs\PPn};
\draw (180:.45\textwidth) node[scale=.95*\Psc,rotate=90] {\bfs\PPw};
\draw (270:.45\textwidth) node[scale=.95*\Psc] {\bfs\PPs};
\end{tikzpicture}}

```

Na obrázku 2 sú postupne zobrazené modré podpivníky vygenerované s parametrami $\{.4\}\{4\}\{t|.38\}$, $\{.4\}\{4\}\{c|.31\}$, $\{.4\}\{4\}\{b|.24\}$ a pod nimi v spodnom riadku po zmene farby `\colorlet{FaI}{red!60!black}` červené s parametrami $\{.2\}\{x\}\{c|.31\}$, kde x predstavuje počet vrcholov od 1 po 6.



Obr. 2. Konferenčný Podpivník v rôznych vyhotoveniach

Druhý podpivník by som skôr nazval podsudovník a nie je tvorený samostatným makrom. Skladá sa z piatich kruhov pravidelne umiestnených po obvodu kružnice, ktoré obsahujú po päť vnútorných kruhov s textom (obr. 3).

Najprv sú definované vlastné TikZ podmakrá, ktoré sa potom dajú spúšťať v prostredí `tikzpicture`. Majú názvy `\rbK` a `\rbM`. Vlastného textu je menej, ale rôzne rotuje vo vnútorných kruhoch. Názvy jednotlivých príkazov v makre `\rbM` sa opäť generujú v cykle pomocou `\cename TT\makealph\xi\endcsname`, skladajú z písmen TT a z písmen, na ktoré sa transformujú čísla pomocou príkazu `\makealph`, ktorý je nutné definovať v preambule príkazom

```

\makeatletter
\newcommand\makealph[1]{\@alph{#1}}
\makeatother

```

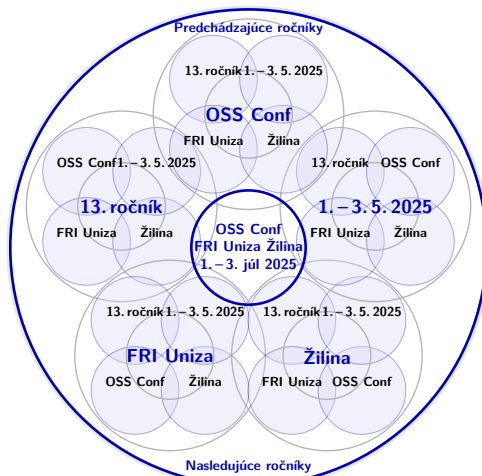
Pri kreslení samotného obrázku v príkaze cyklu `\foreach \x in {1,...,5} {\rbK at {90+\x*72:.7\DX} }` je nutná na konci aspoň jedná medzera alebo prechod na nový riadok medzi `\DX} }`, inak preklad havaruje.

Pred `\begin{tikzpicture}` musia byť definovaná farba kreslenia `FaI`, polomer obvodového kruhu `\DX` a samotné texty `\Ta` – `\Te`. Ešte je tam definované makro `\bfs` na prevod do tučného sanserif písma.

```
\colorlet{FaI}{blue!70!black}\def\FBI{\color{FaI}}
\newlength{\DX}\setlength{\DX}{14cm}
\def\Ta{OSS Conf} \def\Tb{1.\,--\,3.\,5.\,2025} \def\Tc{13.\,ročník}
\def\Td{FRI Uniza} \def\Te{Žilina} \def\bfs{\bf\sffamily}
```

Makrá `\rbK` a `\rbM` spúšťateľné v prostredí `tikzpicture` vyzerajú nasledovne:

```
%.....
\def\rbK at #1 {\begin{scope}[scale=.4]
  \foreach \xi in {1,...,4}
    {\draw[FaI,fill=FaI!15,opacity=.35] (#1)+(45+\xi*90:.26\DX) circle (.23\DX);}
  \draw[black!30,line width=1.pt] (#1) circle (.23\DX) (#1) circle (.5\DX);
  \end{scope}}
%.....
\def\rbM at #1#2#3#4#5#6 {\def\TTx{\bfs#2}\def\TTa{\bfs#3}
  \def\TTb{\bfs#4}\def\TTc{\bfs#5}\def\TTd{\bfs#6}
  \begin{scope}[scale=.4,black]
    \draw (#1) node[scale=1.3] {\FBI\TTx};
    \foreach \xi in {1,...,4} {\draw (#1)+({45+(\xi-1)*90}:.26\DX)
      node[above,scale=.9] {\csname TT\makealph\xi\endcsname};}
  \end{scope}}
```



Obr. 3. Podsudovník OSS Conf 2025

Samotný zdrojový kód prostredia `tikzpicture` je:

```
\begin{tikzpicture}[scale=.9]
\draw[black!10, line width=4pt] (0,0) circle (.5\DX+1pt);
\draw[FaI,line width=2pt] (0,0) circle (.5\DX);
\foreach \x in {1,...,5} {\rbK at {90+\x*72:.7\DX} }
  \rbM at {90+0*72:.7\DX}{\Ta}{\Tb}{\Tc}{\Td}{\Te}
  \rbM at {90+1*72:.7\DX}{\Tc}{\Tb}{\Ta}{\Td}{\Te}
  \rbM at {90+2*72:.7\DX}{\Td}{\Tb}{\Tc}{\Ta}{\Te}
```

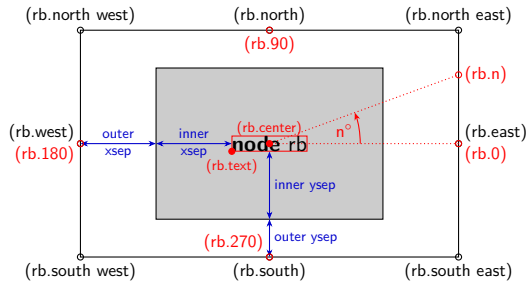
```

\rbM at {90+3*72:.7\DX}{\Te}{\Tb}{\Tc}{\Td}{\Ta}
\rbM at {90+4*72:.7\DX}{\Tb}{\Ta}{\Tc}{\Td}{\Te}
\draw[FaI,line width=2pt] (0,0) circle (.12\DX) node[FaI,scale=1.05]
{\bfs{\begin{tabular}{@{}c@{}}OSS Conf\\ FRI Uniza Žilina\\
1.\,--\,3.\ júl 2025\end{tabular}}};
\draw (90:.46\DX) node[FaI,scale=.95] {\bfs Predchádzajúce ročníky}
(-90:.46\DX) node[FaI,scale=.95] {\bfs Nasledujúce ročníky};
\end{tikzpicture}

```

3. Použitie knižnice positioning

TikZ obsahuje mnoho rôznych knižníc na riešenie rôznych problémov. Jedna z nich je **positioning** pomocou, ktorej môžeme umiestňovať objekty na rôzne pozície v závislosti od predchádzajúcich. Tieto objekty môžeme umiestňovať tak, že zadáme iba smer (svetové strany) alebo môžeme zadať aj vzdialenosť medzi objektami (viď obr. 4). Uhol n sa počíta v kladnom zmysle v stupňoch v rozsahu od 0° po 360° , pričom (rb.0) zodpovedá (rb.360).



Obr. 4. Okolité pozície okolo daného vrcholu **rb**

Pre vytvorenie modelu na obr. 5 potrebujeme načítať knižnice:

```

\usetikzlibrary{arrows.meta}
\usetikzlibrary{positioning}
\usetikzlibrary{shapes.multipart}

```

Základom uvedeného modelu sú rámčeky a poschodové rámčeky (vytvorené pomocou `.style`), ktoré sú poprepájané šípkami. Správne dorazenie šípiek na okraje rámčekov zabezpečuje TikZ, našinec iba nastavuje medzery a pozície medzi jednotlivými rámčekami. Aby rámčeky mali rovnakú výšku, niekedy je použitý príkaz `\smash{text}`, ktorý zabezpečí nulovú výšku textu v zátvorkách. Problémy robia dolné presahy písmen `pjqgz` a horné presahy niektorých písmen s diakritikou `íšž` ap. Podobne ako pri podpivníku je potrebné hneď za `\begin{document}` zadať príkaz `\shorthandoff{-}`. Na horizontálne posunutie šípiek je v niektorých prípadoch použitý príkaz na posunutie `xshift`, napr. konštrukcia

```

([xshift=-.5cm]rbL01.north)--([xshift=-.5cm]rbL01 |- rb01.south)

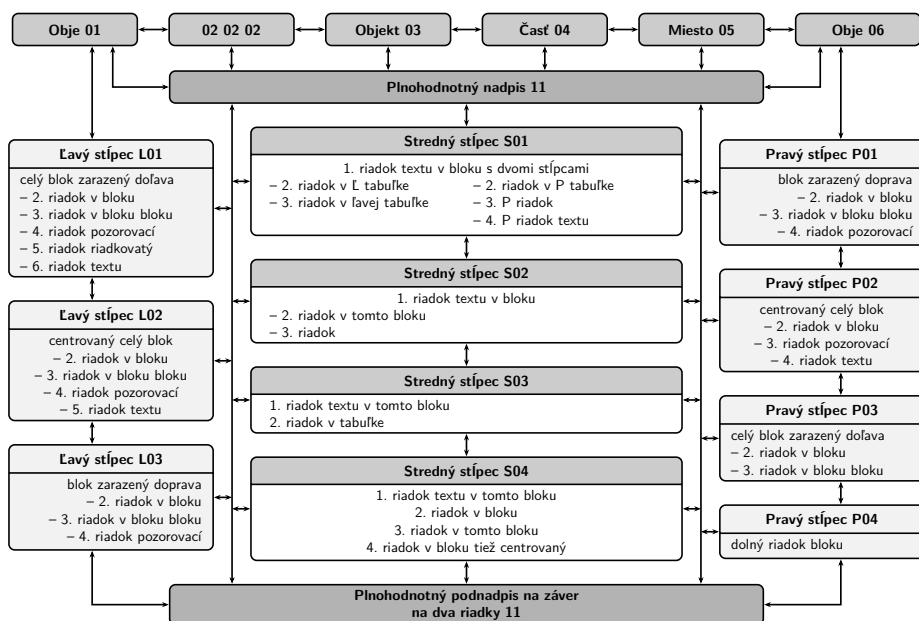
```


zabezpečí, že sa pozícia (`rbL01.north`) posunie o pol centimetra doľava. Druhá časť príkazu (konkrétne `| -`) spôsobí, že sa na horizontálnu úroveň (`rb01.south`) vytvorí kolmica z pozície (`[xshift=-.5cm]rbL01.north`). Opačné poradie

(`[xshift=-.5cm]rbL01.north`)--(`[xshift=-.5cm]rbL01 -| rb01.south`)

by vytvorilo kolmicu na vertikálnu úroveň (`rb01.south`).

Rámčeky sa konštruujú od pozície (0,0) smerom nadol v troch stĺpcoch, preto sú na začiatku definované tri fiktívne body pomocou príkazu `\coordinate`. Samotný text je zapisovaný priamo v rámčekoch do tabuľky `tabular`, ktorá je kvôli väčšej prehľadnosti preddefinovaná v makre `\rbT`.



Obr. 5. Poprepletaný dvojrozmerný model

Publikovanie celého zdrojového kódu presahuje rozmery príspevku, čitateľ ho nájde na adrese <https://invimath.fri.uniza.sk/images/slides/OSSConf/2025-obr05.tex> ako standalone súbor. Pre ilustráciu uvedieme iba niekoľko riadkov od začiatku zdrojového kódu, aby bolo zrejme použitie príkazov.

```
\def\rbT[#1]#2{\begin{tabular}[t]{@{}#1@{}}#2\end{tabular}}
\noindent\scalebox{.4}{\sf\def\bfs{\bf\sffamily}\begin{tikzpicture}[%
  rbto/.style={line width=1.25pt,{Stealth[scale=.6]}-Stealth[scale=.6]}},
  rbi/.style={rectangle,fill=black!20,minimum width=3.3cm,minimum height=2.2em
    ,very thick,draw=black,rounded corners=.4em},
  rbII/.style={rectangle,fill=black!30,minimum width=15.7cm,minimum height=2.2em
    ,text width=14.7cm,text centered,very thick,draw=black,rounded corners=.4em},
  rbS/.style={rectangle split,rectangle split parts=2
```



```

,rectangle split part fill={black!20,white}
,draw,minimum width=11.4cm,text width=10.4cm,very thick,rounded corners=.4em},
rbLR/.style={rectangle split,rectangle split parts=2,draw,minimum width=5.4cm
,fill=black!5,very thick,draw,text width=4.8cm,inner sep=4pt,rounded corners=.4em}]

%% koordinaty fiktivnych bodov na odrazenie
\coordinate (rb00) at (0,0);
\coordinate (rbL0) at (-9.4,0);\coordinate (rbP0) at (9.4,0);
%% horný riadok
\node[rbI,right] at (.4,0) (rb04) {\bfs \smash{\Časť} 04};
\node[rbI,right =.8 of rb04] (rb05) {\bfs Miesto 05};
\node[rbI,right =.8 of rb05] (rb06) {\bfs \smash{Objekt} 06};
\node[rbI,left] at (-.4,0) (rb03) {\bfs \smash{Objekt} 03};
\node[rbI,left =.8 of rb03] (rb02) {\bfs 02 02 02};
\node[rbI,left =.8 of rb02] (rb01) {\bfs \smash{Objekt} 01};
\foreach \xi/\xj in {1/2,2/3,3/4,4/5,5/6} {\draw[rbto] (rb0\xi.east)--(rb0\xj.west);}

%% nadpis pod horným riadkom
\node[rbII,below=2.8em of rb00] (rb11) {\bfs\smash{Plnohodnotný nadpis} 11};
\draw[rbto] ([xshift=1cm]rb01.south)|-(rb11.west);
\draw[rbto] ([xshift=-1cm]rb06.south)|-(rb11.east);
\foreach \xi in {2,...,5} {\draw[rbto] (rb0\xi.south)--(rb11.north -| rb0\xi);}

%% stredný stĺpec
\node[rbS,below =.6 of rb11] (rbS01) {\bfs\centerline{Stredný stĺpec S01}
\nodepart{second}\centering 1. riadok textu v bloku s dvomi stĺpcami \\
\rbT[l]{-- 2. riadok v ľ tabuľke \\ -- 3. riadok v ľavej tabuľke}\hfill
\rbT[l]{-- 2. riadok v P tabuľke \\ -- 3. P riadok\\
-- 4. P riadok textu}\hfill\phantom. };
\path (rb11) edge[rbto] (rbS01);

\node[rbS,below =.6 of rbS01] (rbS02) {\bfs\centerline{Stredný stĺpec S02}
\nodepart{second}\centerline{1. riadok textu v bloku} \\
-- 2. riadok v tomto bloku \\
-- 3. riadok \\
};
\path (rbS01) edge[rbto] (rbS02);

\node[rbS,below =.6 of rbS02] (rbS03) {\bfs\centerline{Stredný stĺpec S03}
\nodepart{second}1. riadok textu v tomto bloku \\
2. riadok v tabuľke \\
};
\path (rbS02) edge[rbto] (rbS03);

```

4. Použitie knižnice intersection

Použitie hľadanie priesečníkov kriviek pomocou balíčka ilustruje obr. 6. Jednotlivé krivky sa označia pomocou príkazu `\path` a následne pomocou konštrukcie `\draw[name intersections={of=...}` sa nájdu a vykreslia priesečníky. V prvom prípade sa vykreslí jeden priesečník (`intersection-1`), v druhom prípade sa vykreslia dva priesečníky (`intersection-1`) a (`intersection-2`). Posledný príklad v ukážke nájde všetky priesečníky, ich počet dá do premennej `\t` a následne ich v cykle `\foreach` vykreslí.

```

\begin{tikzpicture}[scale=.6,rbA/.style={red!60!black,fill}]
\clip (-3,-3) rectangle (3,3);
\path[save path=\ppA,name path=ppA] (-3.2,-2) .. controls (-3,2) and (0,-2) .. (3.2,2);
\path[save path=\ppB,name path=ppB] (-3.2,1) .. controls (0,2) .. (3.2,-1);
\path[save path=\ppC,name path=ppC,yscale=1.2] (0,0) circle (2.2);
\path[save path=\ppD,name path=ppD] plot[domain=-3:3,samples=50] (.8*\x*\x-2.75,\x-.5);
\fill[blue!20,opacity=.5] [use path=\ppC];
\fill[red!10,opacity=.5] [use path=\ppD];
\draw[blue!40!black,thick] [use path=\ppA] (2.8,1.9) node[scale=.7] {$a$};
\draw[red!40!black,thick] [use path=\ppB] (-2.7,.9) node[scale=.7] {$b$};
\draw[blue!40!black,thick] [use path=\ppC] (1.6,-2.1) node[scale=.7] {$c$};
\draw[red!40!black,thick] [use path=\ppD] (1.6,-2.5) node[scale=.7] {$d$};

\draw[name intersections={of=ppA and ppB,by={ppAB}}]
(intersection-1) [rbA] (ppAB) circle (.075) node[below,scale=.5] {$ab$};

\draw[name intersections={of=ppA and ppC,by={ppAC1,ppAC2}}]
(intersection-1) [black] (ppAC1) circle (.075)
++(0,-.1) node[right,scale=.5] {$ac\_2$}
(intersection-2) [black] (ppAC2) circle (.075) node[below right,scale=.5] {$ac\_1$};

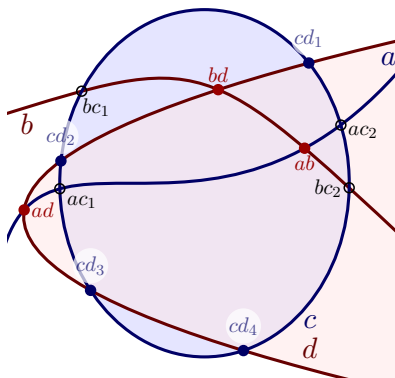
\draw[name intersections={of=ppA and ppD,by={ppAD}}]
(intersection-1) [rbA] (ppAD) circle (.075) node[right,scale=.5] {$ad$};

\draw[name intersections={of=ppB and ppC,by={ppBC1,ppBC2}}]
(intersection-1) [black] (ppBC1) circle (.075)
++(-.1,0) node[below right,scale=.5] {$bc\_1$}
(intersection-2) [black] (ppBC2) circle (.075) node[left,scale=.5] {$bc\_2$};

\draw[name intersections={of=ppB and ppD,by={ppBD}}]
(intersection-1) [rbA] (ppBD) circle (.075) node[above,scale=.5] {$bd$};

\draw[name intersections={of=ppC and ppD,name=ppCD,total=\t},blue!40!black,fill]
foreach \i in {1,...,\t} {(ppCD-\i) circle (.075) ++(0,.1)
node[circle,fill=white,inner sep=.2pt,above,scale=.5,opacity=.7] {$cd\_{\i}$}};
\end{tikzpicture}

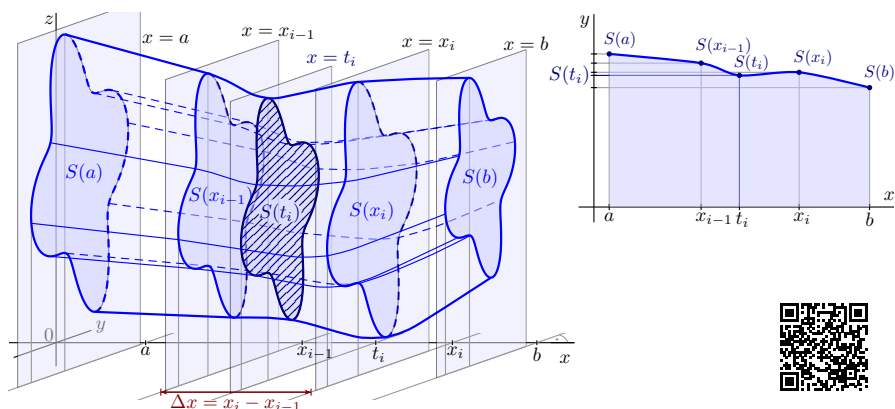
```



Obr. 6. Použitie knižnice `intersection` na hľadanie priesečníkov kriviek

Na záver ešte ukážem jeden obrázok 7, pri tvorbe ktorého som použil balíček `intesection`. Obrázok ilustruje známe tvrdenie pre výpočet objemu telesa, kde poznáme funkciu $y = S(x) \geq 0$, $x \in \langle a; b \rangle$, ktorá vyjadruje plošný obsah prierezu telesa rovinou kolmou na súradnicovú os x :

$$V_x = \int_a^b S(x) dx.$$



Obr. 7. Objem telesa so známym prierezom kolmým na os x

Poďakovanie. Príspevok vznikol s príspevom grantu KEGA 019ŽU-4/2023 „Inovatívne učenie matematiky s podporou Open Source“ podporeného Slovenskou kultúrne-edukačnou grantovou agentúrou.

Literatúra

- [1] Blaško, R., *Neurčitý a určitý integrál a funkcie viacerých premenných*, 2024, online, <https://frcatel.fri.uniza.sk/users/beerb/ma1/sa-2.pdf>.
- [2] Manuál TikZ, <https://www.ctan.org/pkg/pgf>.
- [3] Casteleyn, J. P., *Visual TikZ*, <https://mi-rroirs.nic.cz/tex-archive/info/visualtikz/VisualTikZ-fr.pdf>.

Kontaktná adresa

RNDr. Rudolf Blaško, PhD., Katedra matematických metód, Fakulta riadenia a informatiky, Žilinská univerzita, Univerzitná 8215/1, 010 26 Žilina, Slovensko, *Aktuálna adresa*: SOIT, 010 01 Žilina, Slovensko,
E-mailová adresa: beerb@frcatel.fri.uniza.sk, <http://frcatel.fri.uniza.sk/~beerb/>

VÝUKA MATEMATIKY POMOCÍ UMĚLÉ INTELIGENCE

ELIŠKA CÉZOVÁ (CZ)

Abstrakt. Rozvoj umělé inteligence (AI) přináší nové příležitosti i výzvy v oblasti vzdělávání, zejména ve výuce matematiky. Tento článek se zabývá možností využití AI nástrojů k podpoře výukového procesu. Srovnává výhody a nevýhody jednotlivých přístupů. Dále analyzuje přínosy i rizika spojená s nasazením umělé inteligence ve výuce, zejména v kontextu výpočetních metod a porozumění matematickým pojmům. Cílem je nabídnout vyvážený pohled na potenciál umělé inteligence jako nástroje pro modernizaci a zkvalitnění výuky matematiky.

Klíčová slova. Výuka matematiky, tradiční výuka, moderní metody, digitální technologie, umělá inteligence, pedagogika, matematická analýza, pravděpodobnost.

TEACHING MATHEMATICS USING ARTIFICIAL INTELLIGENCE

Abstract. The development of artificial intelligence (AI) brings new opportunities as well as challenges in the field of education, particularly in the teaching of mathematics. This paper explores the potential of using AI tools to support the educational process. It compares the advantages and disadvantages of various approaches and analyzes the benefits and risks associated with the implementation of AI in teaching, especially in the context of computational methods and understanding mathematical concepts. The aim is to provide a balanced perspective on the potential of artificial intelligence as a tool for modernizing and improving mathematics education.

Keywords. Mathematics education, traditional teaching, modern methods, digital technologies, artificial intelligence, pedagogy, mathematical analysis, probability.

1. Úvod

Matematika je jedním ze základních pilířů vzdělávání a má klíčový význam nejen ve školní výuce, ale i v každodenním životě a celé řadě profesních oblastí od technických oborů až po ekonomii, informatiku nebo přírodní vědy. Díky těmto dovednostem je matematika považována za cenný nástroj rozvoje kritického myšlení a logického uvažování. Způsoby výuky matematiky se však v průběhu času proměňují. Zatímco v minulosti dominovala tradiční forma výuky, kde měl učitel hlavní roli a výklad probíhal převážně frontálně, dnešní přístupy kladou důraz na aktivní zapojení studentek/tů, individualizaci výuky a využívání moderních technologií. Tato změna je reakcí nejen na nové vzdělávací potřeby a cíle, ale i na proměnu společnosti, v níž studentky/ti vyrůstají. Dnešní generace studentek/tů

je stále více digitálně a technologicky orientovaná, a proto je potřeba výuku více propojit s AI.

Cílem tohoto článku je porovnat tradiční a současné přístupy k výuce matematiky a zamyslet se nad tím, jakou roli v ní dnes hrají moderní technologie, včetně umělé inteligence. Zaměříme se jak na přínosy, tak na úskalí, která mohou tyto nástroje do výuky vnést.

2. Výuka matematiky dříve

Výuka matematiky měla v minulosti pevně danou strukturu a jasně stanovené postupy. Dominoval tzv. frontální způsob výuky, při kterém učitel stojí před třídou, vysvětluje učivo, zapisuje vzorce a příklady na tabuli, a studentky/ti si zapisují poznámky nebo řeší zadané úkoly. Tato forma byla efektivní zejména v době, kdy byly třídy početné a učitel měl omezené prostředky pro individuální práci s jednotlivými studentkami/ty.

Typická hodina matematiky se dříve skládala z krátkého opakování předchozí látky, následoval výklad nového učiva s několika názornými příklady na tabuli, a poté samostatná práce studentek/tů. Důraz byl kladen na přesnost, systematickост a opakování. Studentky/ti se učili algoritmy výpočtů a trénovali mechanické dovednosti, jako je postup při řešení rovnic, úpravy výrazů, výpočty sčítání, odčítání, násobení či dělení, derivování, integrování a další postupy.

Učebnice hrály klíčovou roli, byly jednotné a pevně dané. Matematika se často vyučovala jako předmět zaměřený na výkon, disciplínu a přesné zvládnutí výpočetních postupů.

2.1. Výhody a nevýhody tradiční výuky

Tradiční výuka matematiky měla jasnou strukturu a přehledně stanovená pravidla. Tento přístup byl rovněž efektivní při práci s velkým počtem studentek/tů, neboť poskytoval jednotné učivo a standardizované výukové postupy. Silný důraz byl kladen na přesnost, kázeň a zvládnutí technických výpočtů, což mělo pozitivní vliv na zvládnutí formálních aspektů matematiky.

Na druhé straně však tradiční výuka nabízela jen málo prostoru pro aktivní zapojení studentek/tů. Výuka nebyla přizpůsobena individuálním rozdílům v tempu učení ani různým úrovním porozumění.

Z dnešního pohledu je zřejmé, že tradiční přístup měl své silné stránky, zejména ve své důslednosti a důrazu na zvládnutí základních výpočetních technik. Zároveň se však ukazuje, že pro rozvoj hlubšího matematického myšlení, tvořivosti a schopnosti aplikace znalostí v reálných situacích je nezbytné tradiční výuku doplnit o nové metody a přístupy, které lépe odpovídají současným požadavkům na vzdělávání.

3. Výuka matematiky dnes

Současná výuka matematiky prochází významnou proměnou, která reaguje na technologický pokrok, změny ve společnosti i nové poznatky z pedagogiky a psychologie učení. Klasická tabule a učebnice jsou doplňovány nebo nahrazovány moderními technologiemi, které blíže rozvedeme dále.

3.1. Moderní nástroje ve výuce

Učitelé dnes mohou využívat široké spektrum nástrojů, které dříve nebyly dostupné. Dnes je možné používat interaktivní tabule, které umožňují dynamickou práci s grafy, rovnicemi i geometrií. Nebo lze využívat powerpointové prezentace, které jsou vizuálně přehledné, snadno upravitelné, vhodné pro výklad a opakování. Dalšími výukovými materiály jsou online platformy, které jsou užitečné pro procvičování, okamžitou zpětnou vazbu, vizualizaci nebo simulaci. Používání Moodle, kde je možné nastavit různé formy testu. V době COVIDu se dal používat i jako zkoušková online platforma. Výukové aplikace se přizpůsobují úrovni studentkám/tům a umožňují samostatné studium. Používané digitální podklady pro výuky zažily rozmach v době COVIDu (videovýuka a tutoriály), měly výhodu spočívající v dostupnosti offline podkladů k výkladu a dalším podpurným materiálům kdykoli a odkudkoli. Tyto nástroje podporují lepší pochopení abstraktních pojmů, jako jsou funkce, derivace nebo prostorová geometrie, a umožňují názorné znázornění situací, které byly dříve jen těžko představitelné.

3.2. Zapojení umělé inteligence (AI)

Rychlý rozvoj umělé inteligence přináší nové možnosti také do výuky matematiky. Moderní nástroje založené na AI dokážou:

- Řešit rovnice a výrazy krok po kroku, často s komentářem.
- Vysvětlit matematické pojmy různými způsoby.
- Přizpůsobit úroveň úloh podle znalostí studentky/ta, čímž podporují individualizované učení.
- Poskytovat okamžitou zpětnou vazbu – studentce/tovi kteří tak mohou zjistit, kde udělali chybu.

AI je velmi užitečná např. při výpočtech v oblasti matematické analýzy – zvládá derivace, integrály, limity i diferenciální rovnice.

Při využívání AI je však potřeba mít na zřeteli, že umělá inteligence není vždy didakticky přesná, občas nevypočítá jednoduché matematické počty správně, viz obrázek 3. V některých případech používá zjednodušené nebo méně přehledné postupy, které mohou být pro studentky/ty matoucí. Zvláště v oblastech jako

je pravděpodobnost a statistika, kde je klíčové správné pochopení zadání, výběr vhodné metody a logické uvažování, je třeba výsledky ověřovat a doplnit je lidským vysvětlením.

3.3. Výhody a výzvy moderní výuky

Ve výuce matematiky dnes dochází k vyšší míře zapojení studentek/tů, což přispívá k aktivnějšímu a motivovanějšímu učení. Díky moderním technologiím je navíc možné lépe vizualizovat abstraktní matematické pojmy, což usnadňuje jejich pochopení. Digitální nástroje umožňují individualizaci výuky, tedy přizpůsobení obsahu a tempa potřebám jednotlivých studentek/tů. Významnou výhodou je také možnost okamžité zpětné vazby, která pomáhá studentkám/tům ihned rozpoznat chyby a poučit se z nich.

Jedním z rizik moderní výuky matematiky je přílišné spoléhání se na technologie bez hlubšího porozumění matematickým principům, což může vést ke ztrátě schopnosti samostatně přemýšlet. Kvalita výkladů poskytovaných umělou inteligencí je navíc často nekonzistentní – některé postupy mohou být nepřesné, zjednodušené nebo didakticky nevhodné. Významnou výzvou je také potřeba rozvíjet digitální gramotnost nejen u studentek/tů, ale i u učitelů, kteří musí umět s technologiemi efektivně pracovat. Využívání moderních nástrojů zároveň klade vyšší nároky na přípravu učitele, který musí důkladně promýšlet, jak technologie zapojit smysluplně a srozumitelně.

4. Používání umělé inteligence při výuce matematiky

V posledních letech se umělá inteligence (AI) stává významným nástrojem ve výuce matematiky. AI dokáže analyzovat řešení studentek/tů, poskytovat okamžitou zpětnou vazbu a navrhnout individuálně přizpůsobené učební plány podle jejich potřeb a pokroku. Umělá inteligence také pomáhá zobrazovat složité matematické koncepty interaktivním a vizuálním způsobem, což podporuje lepší pochopení látky.

AI je zvláště užitečná při řešení složitých úloh, například v oblasti matematické analýzy nebo integrálního počtu, kde může rychle a přesně vypočítat výsledky, které by studentkám/tům jinak trvaly dlouho. Na druhou stranu je důležité, aby studentky/ti rozuměli nejen konečnému výsledku, ale i logice a postupu výpočtu, což není vždy zaručeno automatizovanými nástroji.

Proto je klíčové, aby použití AI ve výuce bylo doplněno pedagogickým vedením a správným vysvětlením, tudíž aby umělá inteligence sloužila jako efektivní pomůcka, nikoli náhrada učitele.

4.1. Ukázky výpočtů

Na obrázku 1 jsou ukázky výpočtu pomocí bílé tabule, která nepotřebuje dalších komentářů. Na obrázku 2 je prezentováno řešení příkladu z obrázku 1 pomocí powerpointové prezentace.

$$\int_0^1 \int_0^{1-x} (x^2 + y^2) dy dx = \int_0^1 \left[x^2 y + \frac{y^3}{3} \right]_0^{1-x} dx = \int_0^1 \left(x^2(1-x) + \frac{(1-x)^3}{3} \right) dx = \int_0^1 \left(x^2 - x^3 + \frac{1-x^3}{3} \right) dx = \int_0^1 \left(\frac{4}{3}x^2 - \frac{4}{3}x^3 \right) dx = \left[\frac{4}{9}x^3 - \frac{4}{12}x^4 \right]_0^1 = \frac{4}{9} - \frac{1}{3} = \frac{1}{3}$$

Přeskočení kroků

Zapomnětlivost???

$$\int_0^1 \int_0^{1-x} (x^2 + y^2) dy dx = \int_0^1 \left[x^2 y + \frac{y^3}{3} \right]_0^{1-x} dx = \int_0^1 \left(x^2(1-x) + \frac{(1-x)^3}{3} \right) dx = \int_0^1 \left(x^2 - x^3 + \frac{1-x^3}{3} \right) dx = \int_0^1 \left(\frac{4}{3}x^2 - \frac{4}{3}x^3 \right) dx = \left[\frac{4}{9}x^3 - \frac{4}{12}x^4 \right]_0^1 = \frac{4}{9} - \frac{1}{3} = \frac{1}{3}$$

Dříve přeskočené kroky zde uvedeny.

Chyby v prezentaci se nedají ihned mazat.

Obrázek 1. Ukázky výpočtů – bílá tabule **Obrázek 2.** Ukázka výpočtu – prezentace (Čitelnost?)

Porovnáme-li obrázek 1 a 2, je jasné, že obrázek 2 je pro některé studenty lépe pochopitelnější než obrázek 1. U obrázku 2 nemusíme dlouze sepisovat postup, protože jej máme již připravený a lze pouze pomáhat slabším studentkám/tům, kteří potřebují dovysvětlit určité partie z uvedeného příkladu. Promítání předem připravené prezentace má však i úskalí, pokud si zavedeme chybu do prezentace, nemůžeme ji během výuky mazat. Pokud prezentaci odladíme, získáme naopak efektivně využitelnou pomůcku a nevystavujeme se riziku chybného zápisu řešení jako v případě frontální výuky. Pro další cvičení, už se nám nebude opakovat a nebude další jiná vznikat, jestliže nebudeme upravovat prezentaci.

Výpočet dvojného integrálu pomocí AI je na obrázku 3, první iterace výsledku je chybná.

ChatGPT napsal. „Děkuji ještě jednou za upozornění – opravdu jsem minule špatně spočetl druhý integrál.“ Druhá iterace výsledku je správná, viz obrázek 4.

Studenti se snaží učit umělou inteligenci matematickým výpočtům. Bohužel sami u zkoušky většinou neuspějí, protože se často ztratí v matematických postupech, viz obrázek 5. Na obrázku 5 je ukázka výpočtu dvojného integrálu studenta, který nepochopil princip výpočtu.

Generování obrázků pomocí ChatGPT, zdrojový kód je na obrázku 6 a na obrázku 7 je zobrazení oblasti dvojného integrálu.

ChatGPT dokáže matematický postup přepsat do $\text{\LaTeX}$ ového kódu, viz obrázek 8, který je velkým pomocníkem. Nemělo by se však zapomínat na nutnou kontrolu výpočtu, které provedlo ChatGPT, jak je vidět na obrázku 3. Dokáže udělat chyby, s kterými bychom nepočítali.

Máme vypočítat dvojný integrál:

$$\int_0^1 \int_0^{1-x} (x^2 + y^2) dy dx$$

1. Vnitřní integrál podle y :

Integrujeme $x^2 + y^2$ podle y , kde x je konstanta:

$$\int_0^{1-x} (x^2 + y^2) dy = \int_0^{1-x} x^2 dy + \int_0^{1-x} y^2 dy$$

Vypočteme:

$$\begin{aligned} \int_0^{1-x} x^2 dy &= x^2 y \Big|_0^{1-x} = x^2(1-x) \\ \int_0^{1-x} y^2 dy &= \frac{y^3}{3} \Big|_0^{1-x} = \frac{(1-x)^3}{3} \end{aligned}$$

Sečteno:

$$\int_0^{1-x} (x^2 + y^2) dy = x^2(1-x) + \frac{(1-x)^3}{3}$$

2. Vnější integrál podle x :

$$\int_0^1 \left[x^2(1-x) + \frac{(1-x)^3}{3} \right] dx$$

Spočítáme zvlášť:

První člen:

$$\int_0^1 x^2(1-x) dx = \int_0^1 (x^2 - x^3) dx = \left[\frac{x^3}{3} - \frac{x^4}{4} \right]_0^1 = \frac{1}{3} - \frac{1}{4} = \frac{1}{12}$$

Druhý člen:

$$\int_0^1 \frac{(1-x)^3}{3} dx = \frac{1}{3} \int_0^1 (1-x)^3 dx$$

Upravíme a spočítáme:

$$\begin{aligned} (1-x)^3 &= 1 - 3x + 3x^2 - x^3 \\ \frac{1}{3} \int_0^1 (1 - 3x + 3x^2 - x^3) dx &= \frac{1}{3} \left[x - \frac{3x^2}{2} + x^3 - \frac{x^4}{4} \right]_0^1 \\ &= \frac{1}{3} \left(1 - \frac{3}{2} + 1 - \frac{1}{4} \right) = \frac{1}{3} \left(\frac{4 - 3 + 4 - 1}{4} \right) = \frac{1}{3} \cdot \frac{4}{4} = \frac{1}{3} \end{aligned}$$

3. Celkový výsledek:

✓ Odpověď:

$$\frac{1}{12} + \frac{1}{3} = \frac{1+4}{12} = \frac{5}{12}$$

Obrázek 3. Ukázka výpočtu pomocí AI – první iterace

Máme spočítat dvojný integrál:

$$\int_0^1 \int_0^{1-x} (x^2 + y^2) dy dx$$

Krok 1: Vnitřní integrál podle y

Integrujeme:

$$\int_0^{1-x} (x^2 + y^2) dy$$

Rozdělíme:

$$= \int_0^{1-x} x^2 dy + \int_0^{1-x} y^2 dy$$

• První integrál:

$$\int_0^{1-x} x^2 dy = x^2(1-x)$$

• Druhý integrál:

$$\int_0^{1-x} y^2 dy = \left[\frac{y^3}{3} \right]_0^{1-x} = \frac{(1-x)^3}{3}$$

Celý vnitřní integrál:

$$x^2(1-x) + \frac{(1-x)^3}{3}$$

Krok 2: Vnější integrál podle x

$$\int_0^1 \left(x^2(1-x) + \frac{(1-x)^3}{3} \right) dx$$

Spočítáme jednotlivé části:

1. Integrál $\int_0^1 x^2(1-x) dx$

$$x^2(1-x) = x^2 - x^3$$

$$\int_0^1 (x^2 - x^3) dx = \left[\frac{x^3}{3} - \frac{x^4}{4} \right]_0^1 = \frac{1}{3} - \frac{1}{4} = \frac{1}{12}$$

2. Integrál $\int_0^1 \frac{(1-x)^3}{3} dx$

Vytkneme konstantu:

$$\frac{1}{3} \int_0^1 (1-x)^3 dx$$

Změníme proměnnou:

Nechť $u = 1-x \Rightarrow du = -dx$

Když $x = 0$, $u = 1$,

když $x = 1$, $u = 0$

Změna meze:

$$\frac{1}{3} \int_1^0 u^3(-du) = \frac{1}{3} \int_0^1 u^3 du = \frac{1}{3} \left[\frac{u^4}{4} \right]_0^1 = \frac{1}{3} \cdot \frac{1}{4} = \frac{1}{12}$$

Krok 3: Sečteme výsledky

$$\frac{1}{12} + \frac{1}{12} = \frac{2}{12} = \frac{1}{6}$$

✓ Správný výsledek:

$$\frac{1}{6}$$

Obrázek 4. Ukázka výpočtu pomocí AI – druhá iterace

$$\begin{aligned} \textcircled{1} \int_0^1 \int_0^{1-x} (\sin y + xy) dx dy &= \int_0^1 \left(-\cos y + x \cdot \frac{y}{2} \right) dy = \left(-\cos 1 + 1 \cdot \frac{y}{2} \right) - \left(-\cos 0 + 0 \cdot \frac{y}{2} \right) = (-\cos 1 + 1) \\ &+ \int_0^1 \left(-\cos y + \frac{y}{2} \right) dy = \left(-\sin 1 + \frac{y^2}{4} \right) - \left(-\sin 0 + 0 \right) = -\sin 1 + \frac{1}{4} \\ &+ \int_0^1 \left(-\cos y + \frac{y}{2} \right) dy = \left(-\sin 1 + \frac{y^2}{4} \right) - \left(-\sin 0 + 0 \right) = -\sin 1 + \frac{1}{4} \end{aligned}$$

Obrázek 5. Ukázka výpočtu dvojného integrálu studenta

```

\begin{center}
\begin{tikzpicture}[scale=5]
% Osy
\draw[>-] (0,0) -- (1,1) node[anchor=north] {$x$};
\draw[>-] (0,0) -- (0,1) node[anchor=east] {$y$};

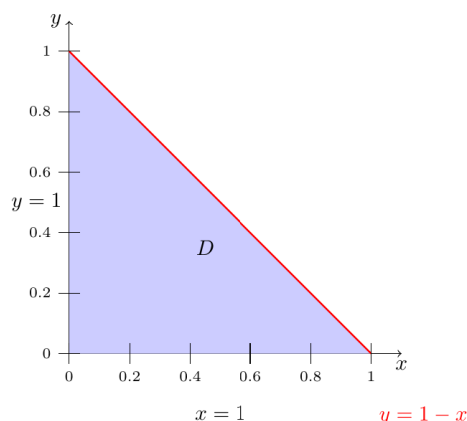
% Oblast integrace
\fill[blue!20] (0,0) -- (1,0) -- (0,1) -- cycle;

% Popisky
\draw (0.5,-0.15) node[anchor=north] {$x=1$};
\draw (0,0.5) node[anchor=east] {$y=1$};
\draw (0.45,0.35) node {$D$};

% Hrana y = 1 - x
\draw[thick, red] (0,1) -- (1,0) node[anchor=north west] {$y=1-x$};
\draw[thick, red] (1,-0.15) node[anchor=north west] {$y=1-x$};

% Osy + mřížka
\foreach \x in {0,0.2,0.4,0.6,0.8,1} {
  \draw (\x,1pt) -- (\x,-1pt) node[below] {\scriptsize $\x$};
}
\foreach \y in {0,0.2,0.4,0.6,0.8,1} {
  \draw (1pt,\y) -- (-1pt,\y) node[left] {\scriptsize $\y$};
}
\end{tikzpicture}
\end{center}

```

Obrázek 6. Kód v T_EXu vygenerovaný ChatGPT pro vytvoření obrázku 7Obrázek 7. Obrázek z T_EXu

5. Závěr

Výuka matematiky prošla v posledních desetiletích výrazným vývojem. Zatímco dříve dominoval klasický přístup založený na výkladu u tabule a počítání příkladů, dnes se výuka stále více otevírá moderním technologiím. Interaktivní prezentace, výukové aplikace a nástroje využívající umělou inteligenci přinášejí nové možnosti, jak látku zpřístupnit, zpestřit a lépe přizpůsobit individuálním potřebám studentek/tů.

AI má v matematice bezesporu své místo – dokáže nabídnout rychlé řešení, personalizované procvičování. Její využití je zvláště přínosné v oblastech, kde jsou výpočty složité nebo časově náročné. Na druhé straně je třeba mít na paměti, že AI může nabídnout nevhodný nebo nepřesný postup, a bez vedení učitele může vést k chybnému pochopení látky.

```

Uvažujme následující dvojný integrál přes trojúhelníkovou oblast:
\[
\int_0^1 \int_0^{1-x} (x^2 + y^2) \, dy \, dx
\]

\subsection*{Postup řešení}

Nejprve vypočítáme vnitřní integrál podle proměnné  $(y)$ :
\[
\int_0^{1-x} (x^2 + y^2) \, dy = x^2(1-x) + \int_0^{1-x} y^2 \, dy
= x^2(1-x) + \frac{(1-x)^3}{3}
\]

Nyní integrujeme podle  $(x)$ :
\[
\int_0^1 \left[ x^2(1-x) + \frac{(1-x)^3}{3} \right] dx
= \int_0^1 \left( x^2 - x^3 + \frac{1-3x+3x^2-x^3}{3} \right) dx
\]

Zjednodušíme pod integrálem:
\[
= \int_0^1 \left( \frac{1}{3} - x + 2x^2 - \frac{4}{3}x^3 \right) dx
\]

```

Obrázek 8. L^AT_EXový zdroj výpočtu

Nejefektivnější výuka matematiky dnes spočívá v promyšleném propojení tradičních a moderních metod. Učitel zůstává klíčovou postavou, která vede, vysvětluje a koriguje. AI by měla sloužit jako podpora, nikoli jako náhrada. Tento přístup vede k efektivnímu osvojení probíraného učiva, hlubšímu porozumění probírané problematiky a připravuje studentky/ty na reálné využití matematiky v praxi.

Kontaktní adresa

Ing. Bc. Eliška Cézová, Ph.D., Ústav konstruování a částí strojů, Fakulta strojní, ČVUT v Praze, Technická 4, Praha 6, CZ-160 00, Česká republika,
E-mailová adresa: eliska.cezova@fs.cvut.cz

HTCONDOR COMPUTATIONAL CLUSTER IN ACADEMIC ENVIRONMENT

MAREK ĎURANA (SK) AND MIROSLAV KVAŠŠAY (SK)

Abstract. This article addresses the design and implementation of a testing environment for research computing infrastructure utilizing educational computers. The environment was developed using the open-source HTCondor software, which enables the utilization of idle computational resources for distributed problem solving. Within this article, we present this software, explain how its test deployment was realized on a small 3-node centrally managed cluster, and introduce plans for its deployment across 320 educational computers located in individual laboratories of the Faculty of Management Science and Informatics of the University of Žilina.

Key words and phrases. HTCondor, computational cluster, high-performance computing, distributed computing.

HTCONDOR VÝPOČTOVÝ KLASTER V AKADEMICKOM PROSTREDÍ

Abstrakt. Článok sa zaoberá návrhom a vytvorením prostredia pre testovaciu prevádzku infraštruktúry zameranej na výskumné výpočty s využitím školských počítačov. Prostredie je vytvorené s využitím open-source softvéru HTCondor, ktorý umožňuje využívať neaktívne výpočtové zdroje na distribuované riešenie úloh. V článku je predstavený tento softvér, je vysvetlené, ako sa realizovalo jeho testovacie nasadenie na 3-uzlovom centrálne riadenom klastri, a sú predstavené plány s nasadením na 320 školských počítačov, ktoré sú lokalizované v jednotlivých laboratóriách Fakulty riadenia a informatiky Žilinskej univerzity v Žiline.

Kľúčové slová. HTCondor, výpočtový klastér, vysokovýkonné výpočty, distribuované výpočty.

Introduction

Educational institutions face a critical resource paradox, while research computing demands continue to grow exponentially, thousands of classroom workstations remain idle for many hours daily and throughout academic breaks. At our institution alone, 320 classroom workstations represent approximately 2,560 compute cores sitting unused outside class hours equivalent to a mid-sized High-Performance Computing (HPC) cluster worth approximately €480,000 in hardware costs.

Traditional solutions like dedicated research clusters require substantial capital investment that many institutions cannot afford. Cloud computing, while flexible, incurs ongoing operational costs that become prohibitive for sustained research

workloads. Our analysis shows that leveraging existing classroom infrastructure could provide 66.66 % of the computational capacity needed for our research programs at essentially zero additional hardware cost [1].

High-Throughput Computing (HTC) has become essential for modern research in computer science, computational biology, physics, and numerous other disciplines requiring substantial computational resources [1]. The open-source HTCondor system has evolved significantly since its inception as the Condor system in the 1980s, demonstrating remarkable scalability in production environments serving millions of core hours daily [2].

HTCondor's flexibility stems from its support for multiple execution environments called *universes*, each tailored for specific types of computational workloads. The *vanilla* universe handles standard executables and represents the most commonly used environment for general-purpose computing tasks. The *standard* universe provides advanced features like checkpointing and remote system calls, enabling fault-tolerant execution of long-running jobs that can survive machine failures. For distributed computing scenarios, the *grid* universe facilitates job submission to remote computing resources, while the *parallel* universe supports MPI and other parallel programming models. Modern research increasingly relies on containerized applications, supported through the *docker* universe, and Java-based scientific applications utilize the dedicated *Java* universe. The *VM* universe enables virtual machine-based computing for specialized environments, and the *local* universe allows execution of administrative tasks on the submit machine itself. This diversity of execution environments makes HTCondor particularly suitable for educational institutions where research spans multiple disciplines with varying computational requirements [1].

This paper describes the design and initial deployment of an HTCondor-based distributed computing infrastructure at the Faculty of Management Science and Informatics of the University of Žilina, with plans for expansion across 320 classroom workstations.

1. Cost-Effectiveness for Educational Institutions

This economic comparison reveals significant cost differences between approaches. HPC cluster deployment requires 320 dedicated compute nodes at €1,500 each, totaling €480,000 in initial investment. Annual operational costs include 10 % maintenance fees, continuous power consumption at 150 W per node operating 24/7, and dedicated cooling infrastructure requirements.

Cloud computing estimates for equivalent sustained computational capacity using 320 instances with 8 cores each range from €300,000 to €800,000 annually, depending on instance type and regional pricing. This approach becomes significantly more expensive than leveraging existing classroom infrastructure, particularly for sustained research workloads requiring continuous operation.

The classroom HTC approach leverages existing workstations equipped with Intel Core i5 processors and 16 GB RAM, avoiding substantial hardware investments. Initial implementation costs cover management infrastructure deployment across school classrooms at €1,500 per machine. Annual operational costs remain minimal, limited primarily to incremental power consumption during off-hours operation when classroom machines would otherwise remain idle.

Further comparison of these three approaches for scalable computing can be viewed in Table 1 .

Table 1. Qualitative comparison of scalable computing

Factor	Dedicated HPC	Cloud	Classroom HTC
Scalability	HW limitation	Unlimited	Classroom limit
Maintenance	High complexity	Vendor managed	Moderate
Energy use	Dedicated cooling	Not applicable	Existing HVAC
Student access	Limited	Limited	Learning value
Implementation	6–12 months	Immediate	2–4 months

1.1. Cost Model Analysis

The classroom-based approach demonstrates significant economic advantages through several factors:

Capital Cost Avoidance: By utilizing existing infrastructure, the approach avoids the substantial initial investment required for dedicated hardware. Our analysis assumes classroom workstations with an average specification of Intel Core i5 processors with 16 GB RAM, providing computational capability equivalent to entry-level server nodes.

Operational Cost Reduction: Unlike cloud computing models that incur per-hour usage charges, classroom HTC leverages existing power, cooling, and network infrastructure. Additional costs primarily include enhanced monitoring software and incremental power consumption during off-hours operation.

Return on Investment: Based on current utilization patterns showing 14+ hours during semesters and 20+ hours outside semesters daily of idle time per workstation, the classroom approach can provide 66.66 % availability while maintaining educational priority. This translates to approximately 40,960 hours of effective compute hours daily across 320 workstations.

2. Architecture

The current HTCondor deployment consists of a three-node centrally managed cluster designed to provide access to distributed resources. Each node corresponds

to one functional instance of HTCondor. The implementation utilizes dedicated management nodes running Windows 11 systems with Intel Core i5-4460 processors, 8 GB RAM, and Samsung SSD 870 EVO storage, connected via gigabit Ethernet infrastructure.

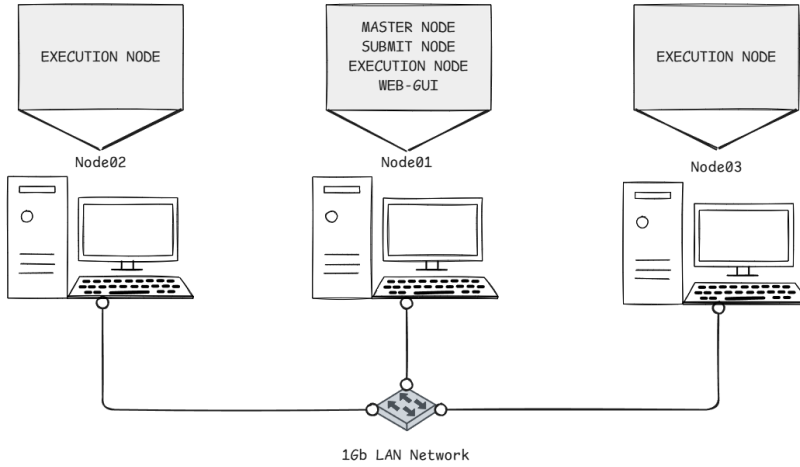


Figure 1. Topology of the current HTCondor testing environment

Figure 1 shows the topology of the current testing environment. The main node (Node01) serves as the Master, Submit, and Execution node, and also hosts the custom Web-GUI. The Master daemon acts as the primary management system for the entire cluster, while the Submit daemon serves as the endpoint for submitting jobs. The Execution daemon is present on all nodes, providing hardware resources to the cluster and managing the execution of jobs on each node.

3. Deployment

The HTCondor deployment utilizes a hierarchical configuration approach with a central management node and dedicated execution nodes. HTCondor was installed with latest available installer for Windows v.24.9.0 [3]. All nodes were installed manually with default options, only selecting Create or Join Cluster options for Master and Execution nodes respectively. The configuration implements modern security practices while accommodating the Windows-based educational environment.

3.1. Master Node Configuration

The central management node (node01.fri.uniza.sk) runs the complete HTCondor service stack. As the cluster coordinator, it hosts all central services including

the Collector (resource advertisement), Negotiator (job matching), and Schedd (job scheduling). The configuration (Figure 2) emphasizes security through IDTOKENS authentication while maintaining compatibility with Windows Active Directory integration.

```
# HTCCondor Master Node Configuration (condor_config)
# Installation and directory structure
RELEASE_DIR = C:\condor
LOCAL_CONFIG_FILE = $(LOCAL_DIR)\condor_config.local
LOCAL_CONFIG_DIR = $(LOCAL_DIR)\config

# Core cluster identification
INSTALL_USER = spravca
CONDOR_HOST = $(FULL_HOSTNAME)
COLLECTOR_NAME = FRI-HTC-01
UID_DOMAIN = fri.uniza.sk
CONDOR_ADMIN = htc-admin@fri.uniza.sk
SMTP_SERVER = smtp.fri.uniza.sk

# Security framework - modern token-based authentication
use SECURITY : recommended(SYSTEM, Administrator@*, spravca@*)
SEC_DEFAULT_AUTHENTICATION_METHODS = CLAIMTOBE, IDTOKENS

# Master node runs full service stack
use POLICY : DESKTOP
DAEMON_LIST = MASTER COLLECTOR NEGOTIATOR SCHEDD STARTD KBDD

# Multi-node authorization matrix
ALLOW_ADVERTISE_MASTER = \
    condor@node01, condor@node01.fri.uniza.sk, 10.10.10.1, \
    condor@node02, condor@node02.fri.uniza.sk, 10.10.10.2, \
    condor@node03, condor@node03.fri.uniza.sk, 10.10.10.3

ALLOW_ADVERTISE_STARTD = \
    -||-

ALLOW_WRITE = \
    -||-

ALLOW_READ = \
    -||-

# Local authentication for administrative tasks
SEC_CLAIMTOBE_INCLUDE_DOMAIN = False
CLAIMTOBE_USER_MAPFILE = $(LOCAL_DIR)\claimtobe.map
```

Figure 2. HTCCondor master node configuration

3.2. Execution Node Configuration

Execution nodes run a simplified configuration focused on job execution. These nodes connect to the master for job assignments and resource reporting while maintaining minimal local services. The configuration (Figure 3) prioritizes execution of jobs through the ALWAYS_RUN_JOBS policy. For automatically yielding resources when students log in for classroom activities, we would use DESKTOP policy.

```
# HTCondor Execution Node Configuration
# Same basic structure as master
RELEASE_DIR = C:\condor
LOCAL_CONFIG_FILE = $(LOCAL_DIR)\condor_config.local

# Points to central management node
CONDOR_HOST = node01.fri.uniza.sk
UID_DOMAIN = fri.uniza.sk
CONDOR_ADMIN = htc-admin@fri.uniza.sk
SMTP_SERVER = smtp.fri.uniza.sk

# Execution-only policy and services
use POLICY : ALWAYS_RUN_JOBS
DAEMON_LIST = MASTER STARTD

# Simplified security for execution nodes
SEC_DEFAULT_AUTHENTICATION_METHODS = IDTOKENS
use SECURITY : recommended(SYSTEM, Administrator@*, spravca@*)
```

Figure 3. HTCondor execution node configuration

3.3. Management Interface Development

A custom web-based management interface (Figure 4) has been developed by us to provide centralized control over the HTCondor infrastructure. This interface allows administrators to monitor cluster status and job queues, manage node availability and power states, configure job submission policies, view resource utilization metrics, and control Wake-on-LAN functionality for future classroom integration.

The web interface is implemented as a Flask-based Python application that serves as a middleware layer between users and the HTCondor command-line tools. The architecture employs a RESTful API design where the frontend communicates with the backend through AJAX requests to various endpoints such as `/api/status`, `/api/jobs`, and `/api/submit`. User authentication is handled through Flask-Login with password hashing using Werkzeug’s security utilities,

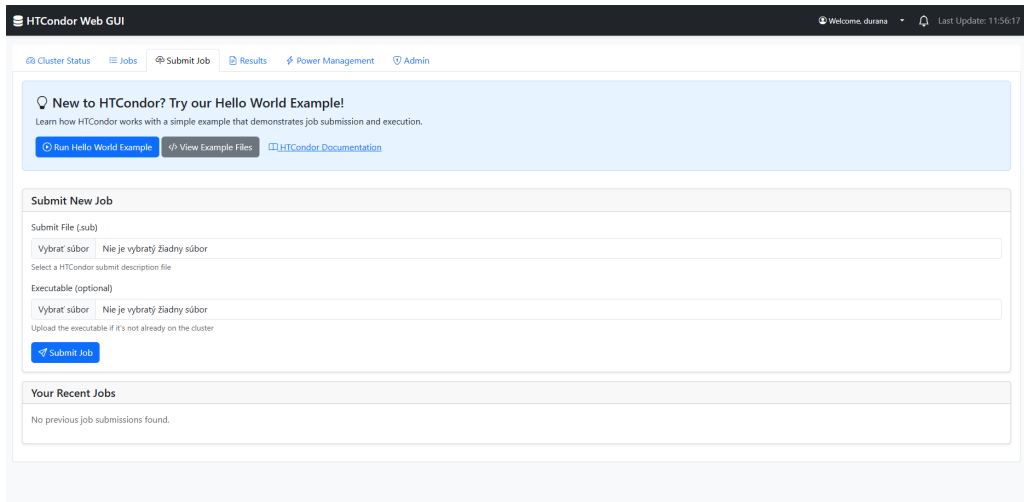


Figure 4. Web GUI submit page

supporting both administrative and regular user roles with appropriate access controls.

The interface maintains persistent data storage through JSON-based databases for user management, job history tracking, and node information. Real-time cluster monitoring is achieved by periodically executing HTCondor commands like `condor_status` and `condor_q` through Python's subprocess module, with results parsed and presented through responsive web templates. Job submission functionality allows users to upload HTCondor submit files and executables through a web form, with the system automatically creating isolated job directories and managing file transfers. The Wake-on-LAN implementation combines traditional magic packet broadcasting with HTCondor's native `condor_power` command, while maintaining a database of MAC addresses gathered from ARP cache lookups and static configuration files.

Installation of the web interface currently works as simple Python app, which requires Python 3.11 to be installed on the Web-Gui node and is accessible on port 5000.

Conclusions

This paper has presented the design and initial deployment of an HTCondor-based distributed computing infrastructure at the Faculty of Management Science and Informatics of the University of Žilina. The current three-node centrally managed cluster has been successfully deployed and tested, demonstrating the feasibility of HTCondor in educational environments with validated *vanilla* universe job

submission and execution, operational custom web interface with real-time cluster monitoring, and functional IDTOKENS-based security system.

Key achievements include successful deployment of robust central management infrastructure, implementation of secure IDTOKENS-based authentication, development of custom web management interfaces, and validation of basic HTCondor functionality. The approach offers a cost-effective alternative to traditional HPC solutions while making efficient use of existing educational infrastructure.

The planned expansion to 320 classroom workstations will leverage automated deployment tools, classroom-aware scheduling that suspends jobs when students log in and Wake-on-LAN integration for energy efficiency. Immediate next steps include pilot deployment on 20–40 workstations, automated power management integration, and university authentication system integration. Anticipated challenges based on similar deployments [4] include hardware heterogeneity requiring careful job matching, unpredictable classroom usage patterns, and managing token-based authentication across 320+ nodes.

Acknowledgment. The article was created with the contribution of the grant KEGA 019ŽU-4/2023 “Innovative teaching of mathematics with Open Source support” supported by the Slovak Cultural and Educational Grant Agency.

References

- [1] THAIN, D. – TANNENBAUM, T. – LIVNY, M.: *Distributed computing in practice: the Condor experience*, Concurrency and computation: practice and experience, vol. 17, no. 2-4, 2005, pp. 323–356.
- [2] BOCKELMAN, B. – LIVNY, M. – LIN, B. – PRELZ, F.: *Principles, technologies, and time: The translational journey of the HTCondor-CE*, Journal of Computational Science, vol. 40, 2020, article no. 101049.
- [3] University of Wisconsin-Madison HTCondor Team: *HTCondor Software Distribution Archive*, <https://research.cs.wisc.edu/htcondor/tarball/current/>, accessed May 16, 2025.
- [4] ACOSTA-SILVA, C. – DELGADO PERIS, A. – FLIX MOLINA, J. – FREY, J. – HERNÁNDEZ, J.M. – LIVNY, M. – PÉREZ-CALERO YZQUIERDO, A. – TANNENBAUM, T.: *Exploiting network restricted compute resources with HTCondor: a CMS experiment experience*, EPJ Web of Conferences, vol. 245, 2020, article no. 09007.

Contact addresses

Ing. Marek Ďurana, Department of Informatics, Faculty of Management Science and Informatics, University of Žilina, Univerzitná 8215/1, 010 26 Žilina, Slovakia,
E-mail address: Marek.Durana@fri.uniza.sk

doc. Ing. Miroslav Kvaššay, PhD., Department of Informatics, Faculty of Management Science and Informatics, University of Žilina, Univerzitná 8215/1, 010 26 Žilina, Slovakia,
E-mail address: Miroslav.Kvassay@fri.uniza.sk

ESHELL: SKUTOČNE CELOOBRAZOVKOVÝ A SKUTOČNE PRÍVETIVÝ EMULÁTOR TERMINÁLU

RICHARD FABO (SK)

Abstrakt. Článok pojednáva o rozšírení GNU Emacs s názvom eshell. Jedná sa o emulátor terminálu (nazývaného aj „konzola“) s mnohými jedinečnými vlastnosťami derivovanými z rozšíriteľného, flexibilného a výkonného interpretra jazyka Elisp s textovým interfejsom.

Kľúčové slová. Shell, eshell, terminál, Emacs.

ESHELL: A TRULY FULL-SCREEN AND REALLY FRIENDLY TERMINAL EMULATOR

Abstract. This article is about an extension to GNU Emacs called eshell. It is a terminal emulator (also called „console“), with many unique features derived from the extensible, flexible, and powerful Elisp interpreter with text interface.

Keywords. Shell, eshell, console, Emacs.

Úvod

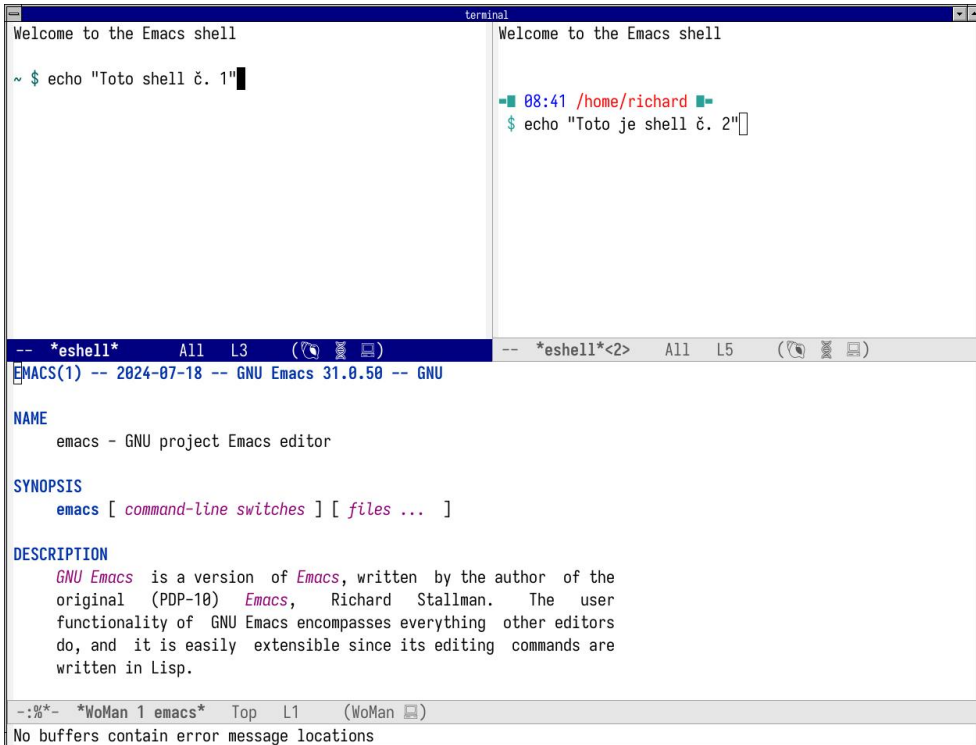
V súčasnosti sa používateľ GNU/Linuxu zaobíde aj bez emulátora terminálu. Drvivú väčšinu nastavení, pri zvolení vhodnej distribúcie, spraví cez grafické rozhranie. Ale kto „zrástol“ s príkazovým riadkom, na jeho efektivitu, priamočiarosť, rýchlosť a možnosť automatizácie nedá dopustiť.

1. Emulátorov terminálov je priveľa

A námatkou by sme mohli spomenúť najznámejšie – dodávané s desktopovými prostrediami, ako sú napríklad Konsole, Gnome-terminal, XCFE4-teminal alebo mnoho iných, ako Alacritty, ST, xterm, urxvt alebo jeden z najnovších – Ghostty. Ich autori sa snažia pridávať nové vlastnosti, avšak pričasto kladú dôraz na dizajnovú stránku emulátora terminálu. Usilujú sa o zobrazovanie obrázkov, ligatúry fontov, polopriehľadné pozadia alebo zaoblené rohy okien, pričom sa stráca hlavný zmysel týchto programov: efektívna práca s textovým výstupom.

Odlišný prístup zvolili autori emulátora terminálu bežiaceho pod softvérovou implementáciou lispového stroja – GNU Emacsom. Počas nasledovných kapitol prevedieme čitateľa jeho vlastnosťami, ktoré potvrdia jeho schopnosť efektívneho

pracovania s textovými príkazmi. Súčasne si z neho spravíme aj „drop-down“ (tzv. „Quake“) terminál.



Obr. 1. Rozdelené okno Emacsu, dva rôzne shell-y a výstup príkazu `man`

2. Pár motivačných vlastností

Eshell je napísaný v Elisp-e (modifikovaná verzia starobylého a stále používaného jazyka lisp). Snaží sa napodobniť správanie interpretra bash a príkazov z GNU Coreutils.

Ale to je asi primálo, aby to bolo niečim výnimočným! Akú pridanú hodnotu musí mať emulátor terminálu vôbec, aby malo zmysel o ňom písať?

Na úvod si treba povedať, že eshell nie je (paradoxne) emulátor terminálu – myslené ako samostatný program. Všetko, čo robí – od zobrazenia textu na obrazovke, až po načítanie obsahu adresára – robí prostredníctvom Emacsu a Emacs komunikuje s operačným systémom priamo. Takže nie je to „niečo“ nad iným shellom. Nie je to `bash` alebo `zsh` alebo `fish`, aj keď je nimi (alebo je to opačne?)

inšpirovaný. Skrátka a dobre – je potrebné na neho nazerať ako niečo iné a odlišné. Neporadí si s celoobrazovkovými textovými (ncurses) oknami, aké používa mutt či htop. Tie sa musia spustiť v oddelenom termináli, tzv. term, o ktorom sa zmienime neskôr.

Eshell umožňuje priame spúšťanie funkcií či programov napísaných v Elisp-e. Taktiež interne pracuje s parametrami ako so zoznamami. A ako vieme, Elisp je pri práci so zoznamami veľmi silný. A jeho silnou stránkou je i TRAMP, čo je nástroj na vzdialený prístup (ftp, ssh. . .). Ďalšou silnou stránkou je jeho existencia v textovom buffri Emacsu, na ktorý sa aplikujú podobné pravidlá a editačné metódy ako na ľubovoľný textový buffer. A toto je jeho najzaujímavejšia vlastnosť. Eshell beží v Emacse a ten je majster v práci s viacerými bufframi, oknami. Takže GNU screen či tmux alebo záložky emulátora terminálu nepotrebujeme.

3. Základné vlastnosti (a iné príjemnosti)

3.1. Multishell

Aj bez dodatočných programov je možné mať viacero shellov vedľa seba. Ak zavoláme napríklad príkaz `man`, tak ten sa otvorí v ďalšom okne, pričom focus ostane na pôvodnom. Potom si môžeme (napr. pomocou `M-PgDn`, `M-PgUp`) listovať nápovedu a súčasne zadávať údaje do príkazového riadku. Alebo – skočiť do druhého okna a odtiaľ si (aj bez myši) prekopírovať text do schránky (v Emacse je nazývaná kill-ring). Označenie v Emacse je trochu zmätočné – okno (window) je de facto rozdelené aktuálne okno Emacsu na viaceré a rámec (frame) je osobitné, oddelené okno desktopového prostredia alebo okenného manažéra. Či už je to okno alebo rámec, všetky tieto entity sú navzájom prepojené.

3.2. Eshell je buffer

A teraz najlepšia vec – keďže aj eshell je de facto bežný textový buffer, tak je možné kurzorom pohybovať všetkými smermi. A text označovať, kopírovať či mazať, alebo aj znovu spúšťať príkazy. Takže žiadne tzv. pager-y ako more či less nie sú potrebné. A toto je veľmi návykové. Nie je potrebné presúvať ruky z klávesnice na myš, využívať posuvníky. Navyše – ako v každom pokročilom textovom editore – máme možnosť vyhľadávania, zvýrazňovania pomocou regulárnych výrazov, potláčania zobrazenia nepotrebného textu, definovať makrá, regióny na aplikovanie zmien, upravovať príkazy zadané na ľubovoľnom mieste. . . , proste všetko, čo si človek môže priať a pokiaľ mu siahla fantázia, pochopiteľne v medziach práce s textom. Dĺžka výstupu nie je obmedzená, je bežné používať tento terminál s počtom riadkov presahujúcich sto tisíc. Číslovanie je riadkov je samozrejmosťou.

3.3. Presmerovanie do iného buffra

Na presmerovanie môžeme použiť okrem bežných `>` a `>>` aj `>>>`, ktorým výstup pridáme na pozíciu kurzora v inom buffri (napríklad buffer s poznámkami, emailovou správou, dokumentom).

Eshell pozná aj tieto pseudozariadenia:

- `echo "Ahoj eshell" > /dev/clip` – vloží výstup do X11 schránky.
- `echo "Ahoj eshell" > /dev/kill` – vloží výstup do vnútornej schránky Emacsu (kill-ring). Kill-ring i bežná schránka desktopového prostredia môžu spravovať ten istý obsah.

```

60 W: Nedá sa načítať /etc/apt/sources.list - open (13: Prístup odmietnutý)
61
62 09:36 /home/richard
63 $ sudo apt update
64 Už existuje:1 http://tux.rainside.sk/mxlinux/antix/bookworm bookworm InRelease
65 Už existuje:2 http://ftp.sk.debian.org/debian bookworm-updates InRelease
66 Už existuje:3 http://security.debian.org bookworm-security InRelease
67 Už existuje:4 http://ftp.debian.org/debian bookworm-backports InRelease
68 Už existuje:5 http://ftp.sk.debian.org/debian bookworm InRelease
69 Načítavajú sa zoznamy balíkov... Hotovo
70 Vytvára sa strom závislostí... Hotovo
71 Načítavajú sa stavové informácie... Hotovo
72 7 packages can be upgraded. Run 'apt list --upgradable' to see them.
73
74 09:36 /home/richard
75 $ sudo apt list --upgradable
76 Listing... Hotovo
77 gir1.2-javascriptcoregtk-4.0/stable-security 2.48.1-2~deb12u1 amd64 [upgradable from: 2.48.0-1~deb12u1.0nosystemd1]
78 gir1.2-webkit2-4.0/stable-security 2.48.1-2~deb12u1 amd64 [upgradable from: 2.48.0-1~deb12u1.0nosystemd1]
79 libjavascriptcoregtk-4.0-18/stable-security 2.48.1-2~deb12u1 amd64 [upgradable from: 2.48.0-1~deb12u1.0nosystemd1]
80 libjavascriptcoregtk-4.1-0/stable-security 2.48.1-2~deb12u1 amd64 [upgradable from: 2.48.0-1~deb12u1.0nosystemd1]
81 libwebkit2gtk-4.0-37/stable-security 2.48.1-2~deb12u1 amd64 [upgradable from: 2.48.0-1~deb12u1.0nosystemd1]
82 libwebkit2gtk-4.1-0/stable-security 2.48.1-2~deb12u1 amd64 [upgradable from: 2.48.0-1~deb12u1.0nosystemd1]
83 yt-dlp/stable-backports,stable-backports 2025.03.31~bpo12+1 all [upgradable from: 2025.03.27~bpo12+1]
84
85 09:36 /home/richard
86 $
-- *eshell* Bot L81 (C) (X) (E)

```

Obr. 2. Pohyb kurzora a výber textu bez obmedzení

3.4. Elisp, dired, autojump, história, záložky, registre

Volaním Elisp-ových funkcií je možné z príkazového riadku spraviť akýkoľvek úkon či príkaz Emacsu. Napríklad `find-file-other-window <súbor>` otvorí súbor v novom okne. Takto sa dajú volať ľubovoľné príkazy, aj tie, ktoré sa volajú pomocou `M-x`, modifikovať premenné Emacsu, definovať funkcie – a to všetko zo spomínaného shellu.

Dired je vstavaný nástroj na dávkovú vizuálnu prácu so súbormi. Takže ak narýchlo potrebujeme spraviť hromadné premenovanie, nejaké šikovné makro, komplikovanejšie kopírovanie, príkaz `dired` je okamžite poruke.

```

terminal
1024
[ 0.085437] rcu: Hierarchical SRCU implementation.
[ 0.085439] rcu: Max phase no-delay instances is 400.
[ 0.085682] smp: Bringing up secondary CPUs ...
[ 0.085776] smpboot: x86: Booting SMP configuration:
[ 0.085777] .... node #0, CPUs:      #1 #2 #3
[ 0.090076] MDS CPU bug present and SMT on, data leak possible. See https://www.kernel.org/doc/html/latest/admin-guide/hw-vuln/mds.html for more details.
[ 0.091014] smp: Brought up 1 node, 4 CPUs
[ 0.091016] smpboot: Max logical packages: 1
[ 0.091018] smpboot: Total of 4 processors activated (24742.64 BogoMIPS)
[ 0.114307] node 0 deferred pages initialised in 22ms
-- *eshell*<2> 27% L194 (🔍 📄 🖨)
[C-c C-e] Edit mode, [M-i] apply all buffers
task_cpu_ids=4.
178 [ 0.085239] RCU Tasks Trace: Setting shift to 2 and lim to 1 rcu_task_cb_adjust=
1 rcu_task_cpu_ids=4.
191 [ 0.085682] smp: Bringing up secondary CPUs ...
193 [ 0.085777] .... node #0, CPUs:      #1 #2 #3
194 [ 0.090076] MDS CPU bug present and SMT on, data leak possible. See https://www.kernel.org/doc/html/latest/admin-guide/hw-vuln/mds.html for more details.
195 [ 0.091014] smp: Brought up 1 node, 4 CPUs
218 [ 0.118047] cpuidle: using governor ladder
219 [ 0.118052] cpuidle: using governor menu
*Helm Swoop* L16 [22 Candidate(s)] C-h m:Help TAB:Act RET/f1..f12:NthAct C-!:Tog.susp
Swoop: cpu

```

Obr. 3. Rýchle vyhľadanie vo výstupe zadaním kľúčového slova, tentokrát o bezpečnostnej chybe v cache pamäte.

Autojump je rozšírenie na uľahčenie presunu medzi adresármi. Zadanie `j` bez argumentov zobrazí zoznam adresárov. Pomocou `j <regexp>` je možné sa do adresára presunúť. Zoznam adresárov je tvorený frekvenciou ich používania. K tomu to je potrebný balík `eshell-autojump`. Existuje i história `cd` príkazov, po zadaní `cd =` je možná voľba (pomocou čísla) zo zoznamu. Nie je to zlé, ale ak už máme inštalovaný rozširujúci framework `helm`, potom môžeme použiť príkaz `helm-eshell-history`, ktorý nám umožňuje v histórii tzv. fuzzy vyhľadanie a samozrejme vkladanie minulých príkazov na pozíciu kurzora.

Súvisiacim príkazom je i `helm-eshell-prompts`, umožňujúci presun kurzoru na minulý príkaz – de facto listuje v buffri po minulých príkazoch podobne ako `eshell-next(previous)-prompt` vizuálne.

Ľubovoľné miesto v shelli, súbore, manuáli, e-mailovej správe, súborovom systéme si môžeme označiť pomenovanou záložkou a potom podľa potreby sa na ne

```

82 libwebkitgtk-4.1-0/stable-security 2.48.1-2~deb12u1 amd64 [upgradable from: 2.48.0-1~deb12u1.ubuntu]
83 yt-dlp/stable-backports,stable-backports 2025.03.31-1~bpo12+1 all [upgradable from: 2025.03.27-1~bpo12+1]
84
85 09:36 /home/richard
86 $ sudo apt list --upgradable
-- *eshell* Bot L86 (🔍 📄 🖨)
C-j: Anonymous (keeping session)
Eshell history
sudo apt list --upgradable

sudo apt update

*helm eshell history* L1 [1025 Candidate(s)] C-h m:Help TAB:Act RET/f1..f12:NthAct C-!:Toq.suspend C-h c:Conf
pattern:

```

Obr. 4. Výber z predchádzajúcich zadaní, dynamicky sa zužujúci

presúvať. Tieto záložky môžu byť trvalé a použité pre ďalšie sedenia. Súčasne si rozloženie okien a ľubovoľný text môžeme uložiť po pomenovaných schránok. A tiež každá časť textového výstupu môže byť uložená do súboru s poznámkami aj s časovou pečiatkou – jednoducho, po stlačení klávesovej skratky.

3.5. Funkcie pre pohyb, kopírovanie a presun textu

Funkcie pre pohyb, kopírovanie a presun textu, na ktoré som si navykol a bežne používam, uvádzam len ich opisy, nakoľko uvádzať ich názvy príp. klávesové skratky by bolo nadbytočné:

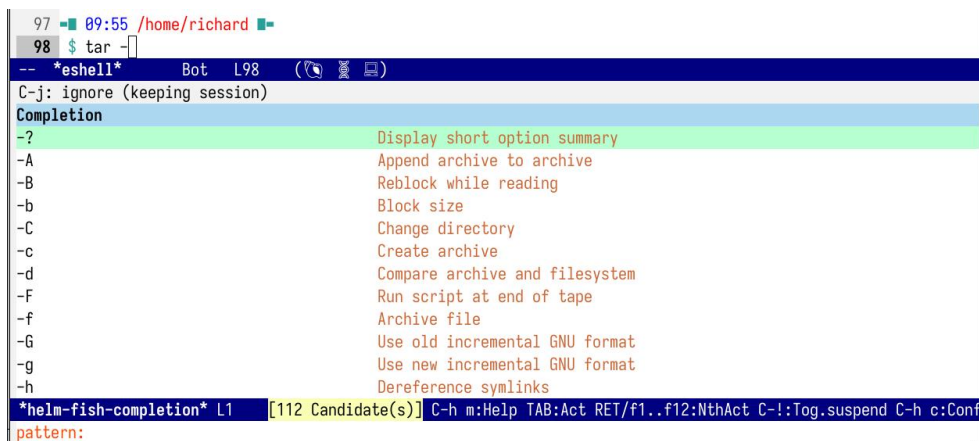
- presun v buffri na nasledujúce (predchádzajúce) zadanie v príkazovom riadku (de facto nie je potrebné sa presúvať pohybom kurzora na spustený príkaz cez všetky výpisy vykonaného príkazu),
- automatické doplnenie príkazu či parametra (ukážeme si aj masívnejšie rozšírenie tejto funkcionality),
- presun na začiatok výstupu posledného príkazu (odbúra „skrolovanie“ pri viacstránkovom výpise),
- doplnenie ďalšieho (predchádzajúceho) zhodného vstupu (napr. po zadaní `cd` a zavolaní funkcie je možné listovať medzi zadanými `cd` v minulosti),
- klasické listovanie v histórii, ako to poznáme z `bash-u`.

3.6. Automatické dopĺňanie

Vraveli sme, že Emacs je majster práce s textom? No zrejme niekoľkokrát a stále je to pravda. A automatické dopĺňanie k práci s textom neodmysliteľne patrí. Dopĺňanie je možné prevádzať viacerými spôsobmi, ktoré sa dajú navzájom kombinovať.

Dopĺňanie názvov súborov a príkazov je klasické ako v `bash-i`, po stlačení `TAB`. Toto je základná funkcionality, ktorá nikoho neohúri, hádam iba človeka, ktorý do včera používal DOS.

Dopĺňanie navyše i parametrov príkazov, t. j. identické správanie ako v `zsh` alebo `fish` shell-i. Je potrebné mať k tomu nainštalované `fish` doplnenia (najjednoduchšie nainštalovaním samotného balíka `fish` pomocu správcu balíkov operačného systému), ktoré sa zvyčajne nachádzajú v `~/.local/share/fish`. Ďalej je potrebný balík s názvom `helm-fish-completion`. A počas zadávania sa po spustení rovnomeného príkazu (alebo nastavenej klávesovej skratke, napr. i znovu `TAB`) sa objaví možnosť vyhľadávania a doplnenia parametrov príkazu.



```

97 09:55 /home/richard
98 $ tar -
-- *eshell* Bot L98
C-j: ignore (keeping session)
Completion
-? Display short option summary
-A Append archive to archive
-B Reblock while reading
-b Block size
-C Change directory
-c Create archive
-d Compare archive and filesystem
-F Run script at end of tape
-f Archive file
-G Use old incremental GNU format
-g Use new incremental GNU format
-h Dereference symlinks
*helm-fish-completion* L1 [112 Candidate(s)] C-h m:Help TAB:Act RET/f1..f12:NthAct C-!:Toq.suspend C-h c:Conf
pattern:

```

Obr. 5. Doplnenie nekonečných parametrov príkazu `tar`

Dopĺňanie pomocou abbrev. `Abbrev` je základnou funkcionalitou Emacsu, ktorá je vždy predinštalovaná. Umožňuje pomocou zadania niekoľkých znakov (stačí i jeden) a volaním príkazu `dabbrev-expand` listovať medzi výskytmi reťazca (predvolene) v aktuálnom buffri. Napríklad pre doplnenie reťazcov, ktoré sú na obr. č. 2, by sme zadali `sudo apt-get install --only-upgrade libw` a niekoľkokrát stlačili `M-/` až do výberu požadovaného doplnenia.

Dopĺňanie pomocou company resp. auto-complete je podobné predchádzajúcemu dopĺňaniu, ale s využitím vyskakovacieho menu. Toto sa objaví automaticky po zadaní troch znakov (ak nie je inak určené inak) a predvolene dopĺňa z aktuálneho buffra, ale dokáže i z iných buffrov či zo slovníka. Primárne je to rozšírenie určené pre programovanie, ale v Emacs-e je možné všetko spolu kombinovať a využívať jedinečné vlastnosti pri všetkých úkonoch – od programovania, písania e-mailov, počúvania hudby až po správu a plánovanie času.

4. Vlastné funkcie GNU Coreutils a vyhľadávanie

Eshell môže používať implementácie tzv. „Coreutils“ príkazov projektu GNU, ale tiež ich vlastné verzie, naprogramované v Emacs-e. Jedná sa o príkazy `ls`, `cd`,

```

temd1]
libwebkit2gtk-4.0-37/stable-security 2.48.1-2~deb12u1 amd64 [upgradable from: 2.4
libwebkit2gtk-4.1-0/stable-security 2.48.1-2~deb12u1 amd64 [upgradable from: 2.48
yt-dlp/stable-backports,stable-backports 2025.03.31-1~bpo12+1 all [upgradable frc
abc libjavascriptcoregtk
abc libwebkit2gtk
09:55 /home/richard
98 $ sudo apt-get install --only-upgrade lib
-- *eshell* Bot L98 (company-dabbrev)

```

Obr. 6. Doplnenie názvu súboru pomocou vyskakovacieho menu

rm, ..., ale najmä **grep**, **find** či **locate**. A posledne menované prinášajú ďalší užívateľský komfort.

4.1. grep, find-dired, locate

Spustenie funkcie **grep** zobrazí dialóg, v ktorom stačí doplniť vyhľadávaný reťazec, prípadne prepínač **-r** pre rekurzívne hľadanie. Výsledky vyhľadávania sú nielen farebné, ale fungujú aj ako hyperlinky, ktoré nás presunú do konkrétneho súboru na konkrétny riadok. Okrem toho existujú ešte tieto varianty: **lgrep** vyhľadáva len v aktuálnom adresári, **rgrep** vyhľadáva rekurzívne, bez toho aby to bolo potrebné povoľovať parametrom **-r** a súčasne rozpoznáva verzovacie systémy, takže sa „vyhýba“ prehľadávaniu nepotrebných adresárov.

```

118 $ 
-- *eshell* Bot L118 (company)
*- mode: grep; default-directory: "~/emacs.d/vendor/" -*-
Grep started at Fri Apr 11 11:55:05

grep --color=auto -nH --null -e backwardp* * -r
addressbook-bookmark/addressbook-bookmark.el:499: (search-backward addressbook-separator nil t)
command-log-mode/command-log-mode.el:82: '(nil self-insert-command backward-char forward-char
command-log-mode/command-log-mode.el:83: delete-char delete-backward-char backward-delete-char
command-log-mode/command-log-mode.el:84: backward-delete-char-untabify
command-log-mode/command-log-mode.el:258: (search-backward "[" (line-beginning-posi
*tion -1) t))
command-log-mode/command-log-mode.el:260: (backward-char) ; skip over either ?\newline or ?\space
*before ?[
ctrlf-master/ctrlf.el:44: ([remap isearch-backward] . ctrlf-backward)
ctrlf-master/ctrlf.el:46: ([remap isearch-backward-regexp] . ctrlf-backward-regexp)
U:%*- *grep* Top L12 (Grep:exit [matched] [867])

```

Obr. 7. Ukážka vyhľadávania reťazca v súborovom systéme

Analogicky fungujú i alternatívy príkazov **find[-dired]** (vyhľadávanie súborov a adresárov podľa zadaných pravidiel) a **locate** (vyhľadávanie súborov v aktualizovanej databáze).

5. TRAMP

TRAMP (skratka Transparent Remote Access, Multiple Protocols) umožňuje pracovať s vzdialenými súbormi a adresármi tak, ako by boli lokálne, so zachovaním všetkých výhod eshell-u a Emacsu: editovanie s použitím všetkých funkcií, ako napríklad zvýrazňovanie syntaxe, dopĺňanie..., používanie všetkých modulov, prekladanie, debuggovanie a spúšťanie kódu. Podporuje ssh, ftp, sftp a scp protokoly a tiež presun dát s využitím rsync.

6. Celobrazovkové (TUI) programy, systémové nástroje

Len v rýchlosti uvedieme, že eshell je tzv. „hlúpy“ terminál a neporadí si s celobrazovkovým textovým interfejsom. Na programy typu htop, mutt alebo mc je potrebné spustiť rozšírenia term alebo vterm. Našťastie aj tie majú mnohé, skôr uvedené praktické vlastnosti.

Taktiež môžeme využiť vstavané Emacs-ové funkcie, ktoré imitujú mnohé systémové nástroje. Napríklad `proced` (alt. `top`), `system-packages-*` (alt. `apt`, `yum`, `rpm`, ...), `vc`, `magit` (rozšírenie `git`) a iné.

7. Eshell ako vyskakovací terminál

Tilda, Guake či Yakuake, to všetko sú terminály, ktoré sa po stlačení klávesovej skratky vysunú z vrchu obrazovky, podobne ako v počítačovej legende Quake.

Podobnú funkcionality je možné dosiahnuť pomocou programu `tdrop` (<https://github.com/noctuid/tdrop>). Tento si vypomáha o. i. programami ako `xprop` (na zistenie vlastností okna), `xdotool` (na zmenu veľkosti a polohy okna) a `xwininfo` (na o. i. zmenu názvu okien).

Vhodné je priradenie klávesovovej skratky desktopového prostredia alebo okenného manažéra pre spustenie kódu:

```
tdrop -ma -h 95% -w 98% -y 30 -s dropdown emacs -f eshell
      -title Eshell
```

Pričom:

- `-ma` – automatické zistenie manažéra a rozlíšenia obrazovky,
- `-h 95%` – nastavenie výšky vysunutého okna (dá sa i absolútne),
- `-w 98%` – detto šírka,
- `-y 30` – offset (posun) okna voči ľavému hornému rohu (ak nechceme, aby vysunuté okno prekryvalo hornú lištu),
- `dropdown` – potrebný vnútorný príkaz `tdrop-u`,
- `emacs -f eshell` – konkrétny program s parametrami (v našom prípade `emacs` so zavolaním funkcie `eshell`),
- `-title Eshell` – názov novovytvoreného „vyskakovacieho“ okna.

8. A teraz negatíva

K zodpovednej recenzii patrí aj kritika. Hlavné nedostatky eshell-u sú:

- V Emacse platí, že buffer je interná reprezentácia textového dátového typu a všetky výmeny dát idú cez (virtuálne) buffre. To sa podpisuje na rýchlosti. Napríklad v adresári `/var/log` príkaz `time cat * | wc -l` prebehne v bash-i za 13 ms, ale v eshell-i to trvá 500 ms. Pri bežnej práci v shell-i tieto rozdiely nie sú badateľné, ale pri nejakom zatažkávajúcom narábaní napr. so sed-om by to mohlo byť obťažujúce.
- V eshell-i je možné pustiť príkaz na pozadí (ako vždy s `&` na konci). Ale ak raz beží na popredí, nie je možné (napr. pomocou známeho C-z) ho presunúť na pozadie. Parafrázujem scénu z 2. časti Hobbita, kde skupinu trpaslíkov (s Gandalfom a Bilbom) naháňa Grizlo. Nakoniec Gandalf na nevyslovené otázky odpovedá: „Nie je zakliaty, je proste taký!“
- V porovnaní s ostatnými nástrojmi Emacsu je vstavaná dokumentácia pomerne strohá.
- Síce je prínosné zažiť výhody a prostredie shell-u i napr. vo Windows, kde je použitie Emacsu tiež rozšírené, ale ak niekto (kto inak Emacs nepoužíva), zatúži po eshell-e, bude konfrontovaný s typicky „Emacsovským“ ovládaním – čo krivku učenia ešte zostrmí.

Záver

Verím, že som opísal všetky podstatné vlastnosti eshell-u a verím, že v dobrom svetle. Kto používa Emacs, a doteraz eshell neskúsil, nech neváha a pravdepodobne si na neho privykne a už s ním zostane.

Môže byť vhodným nástrojom i pre tých, ktorí občas musia zablúdiť i do sveta Microsoft Windows a súčasne chcú mať zachovanú spomienku (na), ovládanie a filozofiu GNU/Linux-u.

Kontaktná adresa

Ing. Richard Fabo, Triton Famme s.r.o., Levočská 862/28, 058 01 Poprad,
E-mailová adresa: triton@famme.sk, <https://triton.famme.sk>

POD KAPOTOU „OPENPLC“

RICHARD FABO (SK)

Abstrakt. Článok opisuje fungovanie programu OpenPLC. Je to integrované vývojové prostredie na programovanie PLC kontrolérov vo viacerých grafických i textových jazykoch v súčinnosti so zaužívanou normou. Jedná sa o slobodné a otvorené riešenie, ktoré je dostupné pre každého – od študentov po profesionálov – má vzdelávací potenciál. Vďaka otvorenej koncepcii je flexibilný a rozšíriteľný. Spolu s inými projektami prispieva k otvorenej a nezávislej priemyselnej automatizácii.

Kľúčové slová. PLC, OpenPLC, Ladder, funkčné bloky, programovanie.

UNDER THE HOOD OF “OPENPLC”

Abstract. This article describes the operation of the OpenPLC program. It is an integrated development environment for programming PLC controllers in multiple graphical and text-based languages in compliance with established standards. It is a libre and open solution that is accessible to everyone – from students to professionals – has educational potential, is flexible and extensible due to its open concept. Together with other projects, it contributes to open and independent industrial automation.

Keywords. PLC, OpenPLC, Ladder, functional blocks, programming.

Úvod

PLC (Programmable Logic Controller) je programovateľný kontrolér, ktorý sa používa na automatizáciu priemyselných procesov a strojov. Slobodné a otvorené riešenia PLC automatov prinášajú do sveta automatizácie obrovskú výhodu – slobodné nástroje pre ich programovanie. Jedná sa o nezanedbateľnú finančnú úsporu, pretože licencia bežných komerčných vývojových nástrojov, ktoré sú navyše previazané s konkrétnym hardvérovým zariadením, stojí rádovo tisíce eur, nehovoriac o novodobom trende v licencovaní softvéru, kde jednorázové a častokrát doživotné licencie nahrádza forma predplatenej služby, často vyžadujúce online pripojenie k serveru výrobcu.

Otvorené PLC zariadenia (A tu hovoríme o dospelých, testovaných, spoľahlivých, sériovo vyrábaných zariadeniach spĺňajúcich požiadavky nasadenia do priemyselného prostredia.), ktorých počet potešiteľne pribúda, námatkou spočítame už etablovaných výrobcov ako Industrial Shields, Controllino, Industrio, Arduino Opta, Productivity Open, resp. UniPi, obsahujú v sebe niektorý z rozšírených mikrokontrolérov (napríklad Arduino Mega, Arduino Leonardo, ESP32, RP2040, ...), ktoré sa najčastejšie programujú v derivátoch jazyka C

alebo **Python**. To je v kontraste so zaužívaným programovaním hlavne v grafických jazykoch. Preto vzniklo niekoľko vývojových prostredí, ktoré tento „neduh“ kompenzujú. Tie úvodzovky sú správne, nakoľko začínajú prevažovať názory, že grafické jazyky sú ťažkopádne pri komplikovanejších algoritmoch, práci s dátami alebo matematickými operáciami. A jedným z týchto vývojových prostredí je multiplatformový program **OpenPLC**.

O programovacích jazykoch pre priemyselné kontroléry sa napísalo veľa návodov, postupov a inej dokumentácie, preto nemá význam opakovať napísané. Taktiež tento príspevok nebude opisovať spôsob práce s programom, opis interfejsu, tlačidiel a pod. Zameriame sa na zaujímavejšiu oblasť: ako program funguje „na pozadí“ a aké nástroje používa.

1. V skratke o IEC 61131-3-3

Tretia časť normy IEC 61131-3 sa zameriava na programovacie jazyky, ktoré môžu byť použité na vývoj softvéru pre PLC.

Táto norma definuje päť programovacích jazykov:

- IL (Instruction List) – textový programovací jazyk podobný assembleru.
- ST (Structured Text) – textový programovací jazyk podobný jazyku Pascal.
- FBD (Function Block Diagram) – grafický programovací jazyk, podobný schéme elektrického obvodu založeného na malých IO (čítače, časovače, logické hradlá AND/OR/XOR/..., ap.).
- LD (Ladder Diagram) – grafický programovací jazyk podobný schéme elektrického obvodu založeného na relé.
- SFC Sequential Function Chart) – grafický programovací jazyk, ktorý definuje stavový stroj.

Je potrebné spomenúť, že implementácia tejto normy nie je vždy striktne rovnaká. Norma poskytuje len základný rámec a definície. Aj to je jedným z dôvodov neprenosnosti programov medzi zariadeniami rôznych výrobcov, ak by aj boli hardvérovo kompatibilné. Tu treba oceniť univerzálnosť OpenPLC, v ktorom po úprave definícií V/V portov zariadenia je program prenosný i medzi rôznymi typmi osadených mikrokontrolérov (pochopiteľne, v rámci ich vzájomnej zameniteľnosti, ako je napríklad počet V/V pinov či prítomnosť rôznych druhov zberníc).

1.1. Ako pracuje PLC v porovnaní s mikrokontrolérom?

Klasické PLC fungujú spôsobom, že sa najprv načítajú vstupy, potom sa vykoná algoritmus a až potom sa zapíšu výstupy. Tento proces sa nazýva „scan cycle“. Ich nevýhodou je potreba výkonnejšieho procesora v porovnaní s mikrokontrolérom (32-bit architektúra, viac jadier, veľa, t. j. viac ako 8 MB RAM, viac ako 256 MB FLASH) pre rýchle deje. Taktiež v moderných PLC beží častokrát neslobodný operačný systém (napr. Microsoft Windows CE).

Oproti tomu mikrokontroléry môžu čítať vstupy a zapísať výstupy už vo fáze vykonávania algoritmu. To znamená, že mikrokontrolér môže čítať vstupy a zapísať výstupy v reálnom čase, bez potreby načítania vstupov a zapísania výstupov v samostatných cykloch.

2. Závislosti a inštalácia

Pomerne presne môžeme povedať, že OpenPLC je kombináciou niekoľkých programov:

- Beremiz – program na vytváranie programu v spomínaných jazykoch,
- matiec – prekladač kódu vytvoreného editorom,
- arduino-cli – program pre prácu s mikrokontrolérmi.

OpenPLC ich navzájom prepája užívateľsky prívetivým spôsobom s vysokou mierou abstrakcie.

Program pre prácu potrebuje tieto programy a tiež knižnice, ktoré užívateľ musí mať nainštalované v prípade používania v operačnom systéme GNU/Linux. Autor článku neskúmal ako je to v prípade neslobodných operačných systémov, ale keďže program má vlastný inštalátor, existuje predpoklad, že si potrebné knižnice nainštaluje. Inštalátor existuje aj pre slobodné operačné systémy, ale my sa zameriame na inštaláciu najnovšej verzie z verzovacieho systému.

Proces inštalácie je uvedený pre distribúcie založené na projekte Debian, v ostatných bude analogický.

Zoznam závislostí: `git`, `bison`, `flex`, `automake`, `autoconf`, `make`. Balíky pre prostredie Python (inštalované cez správcu balíkov alebo `pip`): `jinja2`, `pubsub`, `matplotlib`, `serial`, `zeroconf`, `future`, `lxml`, `wxutils` a `pyro5`.

Pri inštalácii pomocou správcu balíkov operačného systému je tento balík častokrát iba v `sid` repozitári Debianu. V čase písania príspevku je možné nainštalovať balík príkazmi: `wget http://ftp.cz.debian.org/debian/pool/main/p/pyro5/python3-pyro5_5.15-1_all.deb && gdebi python3-pyro5*`.

Inštalácia sa vykonáva „naklonovaním“ aktuálneho repozitára:

- `git clone https://github.com/thiagoralves/OpenPLC_Editor`
a následne
- `cd OpenPLC_Editor/` a spustenie skriptu
- `./install.sh`.

Samotný inštalačný skript stiahne editor Beremiz a kompilátor matiec (`git submodule update -init -recursive "$OPENPLC_DIR"`), doinštaluje knižnice (`libgtk-3-dev`, `python3.9`, `python3.9-venv`, `python3.9-dev`, `attrdict3`, `wheel`), potom prebehne preklad programu matiec pomocou `autoreconf -i` a konečne nakopíruje vytvorené binárne súbory do `./editor/arduino/bin/` a vytvorí desktopový spúšťač.

3. Program matiec

V súčasnosti projekt matiec generuje dva prekladače kódu: `iec2c` a `iec2iec`. Oba kompilátory prijímajú rovnaký vstup: textový súbor s kódom ST, IL a/alebo SFC. Kompilátor pracuje v 5. fázach:

1. Lexikálny analyzátor – implementovaný pomocou flex, zjednodušene rozpoznanie tokenov (kľúčových slov, premenných, literálov, operátorov, ...).
2. Syntaktický analyzátor – implementovaný pomocou bison, čítanie tokenov, kontrola syntaxe, vytvorenie syntaktického stromu, ...
3. Sémantický analyzátor – matiec vykonáva len typovú kontrolu, neoveruje rozsah platnosti, referencie, bloky podmienok, ...
4. Generátor kódu – generuje kód ANSI C.
5. Generátor binárneho kódu – gcc, javac, atď., nie je integrovaný do kompilátora matiec. Tu sa použijú známe nástroje z arduino-cli.

4. Program Beremiz

Beremiz (<https://github.com/beremiz/beremiz/>) je integrované vývojové prostredie (IDE).

Parafrázovanú motiváciu autorov môžeme zhrnúť:

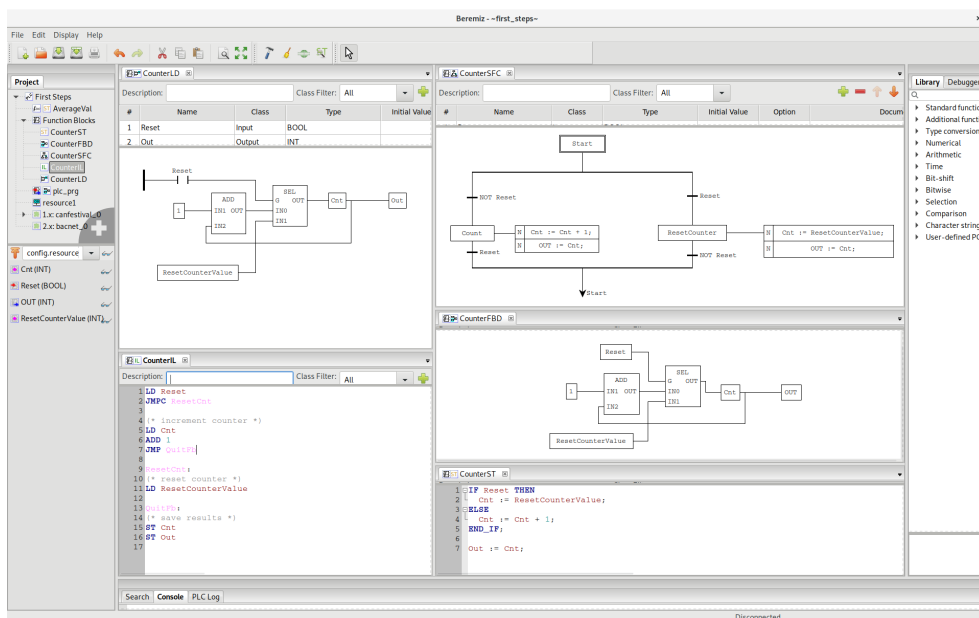
- Napriek otvoreným štandardom, ako sú IEC 61131-3, PLCOpen a CanOpen, je často potrebné prepísať program pre každú konkrétnu značku PLC, pretože sa poskytovatelia proprietárnych riešení plne nepodriaďujú štandardom, čím bránia interoperabilite (čo škodí ich obchodnému modelu a tiež rozvoju digitálnej spoločnosti).
- Napriek otvoreným štandardom, ako sú IEC 61131-3, PLCOpen a CanOpen, je často potrebné prepísať program pre každú konkrétnu značku PLC, pretože sa poskytovatelia proprietárnych riešení plne nepodriaďujú štandardom, čím bránia interoperabilite (čo škodí ich obchodnému modelu a tiež rozvoju digitálnej spoločnosti).
- Špecifikácia štandardov ponecháva výrobcovi určitú voľnosť „špecifickú pre výrobcu“. Poskytovatelia riešení majú z týchto detailov „špecifických pre výrobcu“ veľké výhody – na úkor slobody koncového používateľa.
- Projekt Beremiz vyplní medzeru medzi týmito nezávislými špecifikáciami noriem tým, že poskytuje ich integrovanú referenčnú implementáciu. Týmto nedôjde k tzv. „vendor lock-in-u“ (t. j. k závislosti na) dodávateľa a je to prínos k lepšej (otvorenej, slobodnej, nezávislej, ...) budúcnosti automatizácie.

Beremiz poskytuje:

- Integrované vývojové prostredie (IDE) – t. j. grafické používateľské rozhranie na konfiguráciu, písanie, zostavovanie a ladenie programov pre PLC a riadenie chodu PLC.

- Rozhranie príkazového riadku (CLI) pre zostavovanie PLC programu a pre ovládanie PLC „runtime“ v termináli alebo zo skriptu.
- Tzv. „runtime“, ktorý beží na cieľovej platforme, komunikuje s V/V a vykonáva program PLC.
- Debugovanie v reálnom čase, pomocou modbus (RTU, TCP).

OpenPLC projekt k tomu pridáva dodatočné knižnice reprezentované funkčnými blokmi a generovanie .ino kódu pre arduino-cli, správu vývojových dosiek, a i.



Obr. 1. Integrované vývojové prostredie Beremiz – ukážka všetkých spôsobov programovania.

5. Program arduino-cli

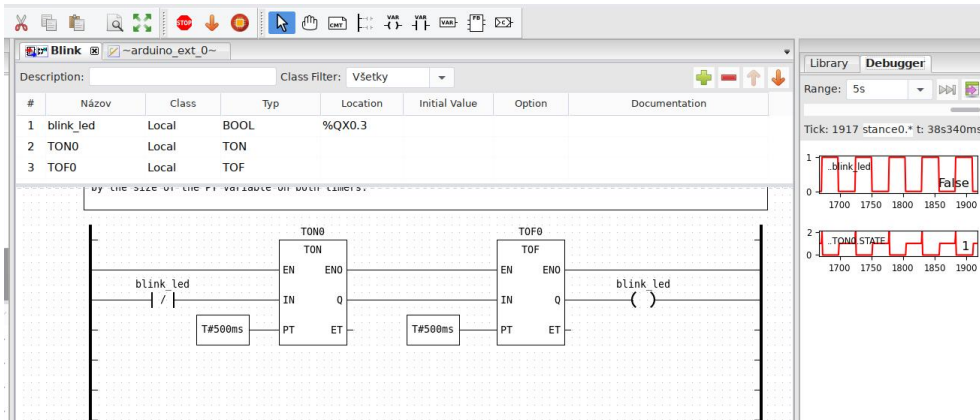
Projekt arduino-cli je command-line interface (CLI) pre Arduino, ktoré umožňuje používať (nielen) Arduino bez potreby grafického užívateľského rozhrania.

Arduino-cli je nástroj, ktorý umožňuje:

- kompilovať a nahrávať kód do vývojových dosiek ako sú Arduino, Espressif, Raspberry Pi, ...
- spravovať knižnice a závislosti,
- konfigurovať a monitorovať vývojové dosky,
- zostavovať a nahrávať programy do vývojových dosiek.

Záujemcov o hlbší rozbor iba odkážeme na prednášku či sériu článkov:

- https://famme.sk/ossconf-2024/prezent-ossconf-2024-progr_work_flow/out.html.
- <https://linuxos.sk/blog/richard/series/arduino-a-kamarati-bez-ide/>.



Obr. 2. Interaktívna simulácia behu programu.

6. Proces prekladu a nahrávania

Ako sme spomínali, OpenPLC zastrešuje niekoľko nástrojov tak, že užívateľ má plnohodnotný komfort práce so programovanými zariadeniami. Ale čo presne sa vlastne deje na pozadí?

6.1. Proces prekladu (napríklad pre potreby simulácie)

1. Beremiz vytvorí z LD, FBD či SFC diagramu program v jazyku ST.
2. Následne program matiec zistí informácie o datových typoch a o funkčných blokoch, ktoré sa používajú v programe.
3. Vygeneruje konfiguračné súbory.
4. Ďalším krokom je generovanie programu v jazyku C.
5. Tento preloží a vytvorí výsledné objektové .o súbory (Config0.o, Res0.o, plc_main.o, plc_debugger.o).
6. Prepojí objektové súbory do jedného: <názov_projektu>.so.
7. Vytvorí MD5 hash na overenie integrity.

6.2. Proces nahrávania programu

1. Opäť sa využije program matiec s tým, že vygenerovaný program v C je doplnený o hlavičky, knižnice a definície pre konkrétnu vývojovú dosku.
2. Následne dôjde k jeho prekladu do `.hex` súboru pomocou `arduino-cli`, napr.:

```
arduino-cli-164 --no-color compile -v --clean --library
~/OpenPLC_Editor/editor/arduino/src --library
~/OpenPLC_Editor/editor/arduino/src/lib
--export-binaries -b arduino:avr:leonardo ~/OpenPLC_Editor/
editor/arduino/examples/Baremetal/Baremetal.ino.
```
3. Z príkladu je zrejmé, že vytvorený zdrojový `.ino` súbor je vždy uložený v:
`arduino/examples/Baremetal/Baremetal.ino`.
4. Následné nahratie sa vykonáva príkazom:

```
arduino-cli-164 --no-color upload --port /dev/ttyUSB0 --fqbn
arduino:avr:leonardo ~/OpenPLC_Editor/editor/arduino/
examples/Baremetal.
```

Poznámka: Skompilované súbory a potrebné knižnice sa nachádzajú vždy v:
`/tmp/arduino/sketches/<vygenerované_číslo_projektu>`.

6.3. Čo sa nachádza v `Baremetal.ino`?

Rýchly pohľad do pripraveného kódu pre `arduino-cli` nám ozrejní, že sa pripojí knižnica `Arduino_OpenPLC.h`, ktorá pridaním `openplc.h` definuje údajové typy pre normu IEC, (nastavenú) rýchlosť jedného cyklu, definície V/V pinov a funkcie, ktoré sa starajú o inicializáciu pinov, ich čítanie a zápis.

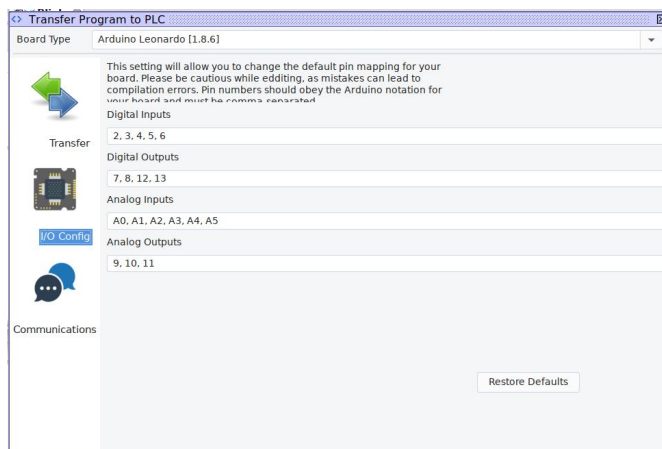
Pohľadom do kódu:

```
void plcCycleTask()
{
    updateInputBuffers();
    config_run__(__tick++);
    updateOutputBuffers();
    updateTime();
}

void scheduler()
{
    plcCycleTask();
    #ifdef USE_ARDUINO_SKETCH
        sketch_loop();
    #endif
}
```

zistujeme, že celý proces vykonávania programu sa deje v spomínanom postupe: čítanie vstupov → samotný algoritmus (riadok `config_run__(__tick++);`) to

práca s výstupmi. Ďalej vidíme, že Beremiz umožňuje pridať vlastný kód jazyka Wiring, pomocou direktívy preprocesora `#ifdef`. To užívateľovi dáva možnosti využitia veľkého množstva (prevažne komunitou vytvorených) knižníc (podpora nového hardvéru, hotové funkcie a moduly atď.).



Obr. 3. Definovanie parametrov mikrokontroléra pre `arduino-cli`.

Záver

OpenPLC je ukázkovým príkladom programu v duchu unixovej filozofie. (Zhrnutá v pár bodoch: „Robiť jednu vec a robiť ju dobre, modularita, prepojenia a presmerovania, dokumentácia, automatizácia.“) Autor Thiago Alves (<https://github.com/thiagoralves>, <https://autonomylogic.com/>) účelne prepojil existujúce dynamické projekty, pridal definície nových vývojových dosiek a tiež vytvoril nové potrebné bloky pre príslušné programovacie jazyky.

Slobodná licencia použitých nástrojov je zárukou, že nedôjde k spomínanému „vendor lock-in-u“, pretože ako nás opakovaná história vývoja slobodného a otvoreného softvéru učí, aj v prípade najsmutnejšieho scenára je otvorená digitálna spoločnosť dostatočne odolná a ochotná pracovať na pokračovaní a zlepšovaní softvéru.

PLC zariadenia postavené na otvorených mikrokontroléroch prekonali počiatočnú skepsu a najlepším dôkazom je množstvo zavedených výrobcov, ktorí sú takto oslobodení od vyvíjania vlastných programovacích nástrojov.

Kontaktná adresa

Ing. Richard Fabo, Triton Famme s.r.o., Levočská 862/28, 058 01 Poprad,
E-mailová adresa: triton@famme.sk, <https://triton.famme.sk>

JE PYTHON PRIATELOM $\text{\LaTeX}$ -U?

ALEŠ KOZUBÍK (SK)

Abstrakt. Dôležitým prvkom sadzby odborného textu je vkladanie zdrojového kódu do dokumentu a možnosťou jeho spustenia a zobrazenia výstupu. Tento príspevok v krátkosti predstavuje možnosti balíčka `pythontex`. Autor ilustruje spoluprácu systému $\text{\LaTeX}$ s jazykom Python.

Kľúčové slová. $\text{\LaTeX}$, Python, sadzba, zdrojový kód.

IS PYTHON A FRIEND OF $\text{\LaTeX}$?

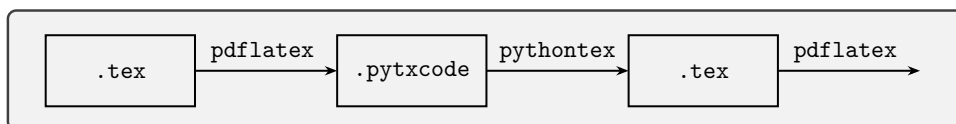
Abstract. An important element of typesetting a professional text is inserting source code into the document and the possibility of running it and displaying its outcome. This contribution briefly presents the capabilities of the `pythontex` package. The author illustrates the cooperation of the $\text{\LaTeX}$ system with the Python language.

Keywords. $\text{\LaTeX}$, Python, typesetting, source code.

Úvod

Dôležitým aspektom tvorby dokumentov je možnosť vkladania zdrojového kódu v nejakom programovacom jazyku s možnosťou jeho spustenia v dokumente a zobrazením výsledku. Vzhľadom na narastajúcu popularitu programovania v jazyku Python je prirodzené pátrať po možnosti spolupráce dokumentu sádzaného v $\text{\LaTeX}$ -u a programovania v Python-e. Odpoveďou je balíček `pythontex`, ktorého autorom je Geoffrey Poore, ktorý má na svedomí tiež balíčky `fvextra`, ktorý rozširuje `fancyvrb`, `latex2pydata` a najmä znovuoživenie balíčka `minted`, umožňujúceho syntax highlight zdrojových kódov v rôznych jazykoch s využitím knižnice `Pygments`.

Balíček vkladáme do preambuly dokumentu obvyklým spôsobom, s použitím príkazu `\usepackage`. Samotné spracovanie dokumentu s použitím tohto balíčka je pomerne jednoduché a prebieha v troch krokoch podľa nasledujúcej schémy.



To znamená, že zdrojový súbor dokumentu s názvom `meno.tex` preložíme pomocou `pdflatex`, pričom okrem obvyklých pomocných súborov sa nám v aktuálnom pracovnom adresári objaví súbor `meno.pytxcode`. Tento súbor sa následne spracuje príkazom `pythontex meno` a ak spracovanie prebehne bez chybových hlásení, súbor ešte raz skompilujeme pomocou `pdflatex`. Za zmienku stojí aj to, že pokiaľ v dokumente nedôjde ku zmene pythonovského kódu, tak pri opakovanej kompilácii stačí len kompilácia pomocou `pdflatex` a procedúru s utilitou `pythontex` nie je potrebné opakovať.

1. Prehľad príkazov a prostredí

Balíček `pythontex` poskytuje 7 príkazov, ktoré je možné použiť v rámci bežného textu v dokumentoch L^AT_EX-u:

- `\py` vyhodnocuje ý výraz a do textu vysádza jeho hodnotu. Napríklad príkaz `\py{2**5-10}` vloží do dokumentu hodnotu 22.
- `\pyc` vykoná pythonovský kód, ale výsledok sa zapíše do `stdout` a do textu sa hodnota nevysádza. Pre zobrazenie výsledku je treba použiť funkciu `print`. Napríklad `\pyc{print(2**5-10)}` dáva ako výsledok hodnotu 22.
- `\pyb` vykoná pythonovský kód, ale do textu sa vysádza samotný kód. Napríklad `\pyb{2**5-10}` dáva v texte výsledok `2**5-10`. Tu nepomôže ani funkcia `print`, lebo dostaneme výstup `print(2**5-10)`.
- `\pys` Podporuje premenné a substitúciu výrazov.
- `\pyv` Tlač kódu so syntax highlight, ale nič sa nevykonáva.
- `\pygment` Všeobecne tlač zdrojového kódu.
- `\pycon` Sprístupňuje premenné z predchádzajúceho bloku `pyconsole`.

Okrem týchto príkazov je k dispozícii 7 prostredí:

- `pycode` kód v tomto prostredí sa vykoná, ale nevysádza.
- `pyblock` kód sa vykoná, ale zobrazí sa len zdrojový kód so syntaxhighlight. Pre zobrazenie výstupu treba použiť príkaz `\printpythontex`.
- `pysob` Substitúcia výrazov a premenných.
- `pyverbatim` Sadzba zdrojového kódu, ale bez jeho spustenia. Rovnaké ako `\begin{pygments}{python}...\end{pygments}`.
- `pygments` Všeobecne sadzba zdrojového kódu.
- `pyconsole` Simuluje interaktívnu konzolu Python-u, t. j. vykonáva príkazy nasledujúce za `>>>`.
- `pythontexcustomecode` kód sa vykoná rovnako ako v `pycode`, ale tlač nie je zobrazená.

2. Vykonanie pythonovského kódu v `pycode` príkazov v `py` a `pyc`

Výpočty je možné vykonávať buď na riadku pomocou makra `\py` alebo v blokovom prostredí ako sú `pycode` alebo `pyblock`.

2.1. Výpočet v riadku s makrom `\py`

Príkaz `\py` odošle vložený kód do interpretu jazyka Python, ten vráti reťazec obsahujúci výsledok spracovania kódu. Tak napríklad príkaz `\py{2**4+2}` vloží do riadku výslednú hodnotu 18. Pripomeňme, že výsledok sa zobrazí až po kompletnej procedúre podľa uvedenej schémy. Po prvej kompilácii sa namiesto výsledku objaví symbol dvoch otáznikov „??“. Príkaz `\py` neumožňuje priradenie hodnôt do premenných, takže napríklad zápis `\py{a=2}` je neplatný. Príčinou je fakt, že priradenie nemožno previesť na reťazec.

Podobne ako `\py` pracuje aj `\pyc`, avšak tento len interpretuje kód a pre zobrazenie výsledku musíme použiť funkciu `print`. V tomto prípade teda musíme písať `\pyc{print(2+2)}` aby sme dostali 4. Príkaz `\pyc` však môžeme využiť na priradenie hodnoty do premennej. Rovnaký výsledok by sme teda dosiahli sekvenciou príkazov `\pyc{a=2+2}\py{a}`. Zobrazíme tak rovnaký výsledok 4. Pre úplnosť dodajme, že ak použijeme funkciu `print` v príkaze `\py`, tak dostaneme výsledok 4 None. Zobrazenie None je dané tým, že `\py` chce vyhodnotiť výraz, ktorý v tomto prípade nie je zadaný.

Ak chceme namiesto spustenia a vyhodnotenia kódu dosiahnuť jeho zobrazenie, a to aj s farebným zvýraznením syntaxe, použijeme príkaz `\pyb`. Tak môžeme vložiť do riadku napríklad kód `print(2+2)`.

2.2. Vykonanie jednoduchého kódu v prostredí pycode

Ako už bolo spomenuté, prostredie `pycode` slúži na vykonanie pythonovského kódu, bez jeho zobrazenia v dokumente. Ak teda do prostredia `pycode` vložíme nasledujúci zdrojový kód:

```
print(r'\begin{center}')
print(r'\textit{Odkaz od jazyka Python !} ')
print(r'\end{center}')
print("[pycode] Pozdravujem \LaTeX ")
```

tak ako výsledok dostaneme tento výstup:

Odkaz od jazyka Python !
[pycode] Pozdravujem L^AT_EX

Významnou vlastnosťou prostredia `pycode` je, že premenné vytvorené v rámci tohto prostredia je možné používať aj ďalej v dokumente, mimo tohto prostredia. Ak napríklad v prostredí `pycode` zavedieme dve premenné:

```
a=2; b=3
```

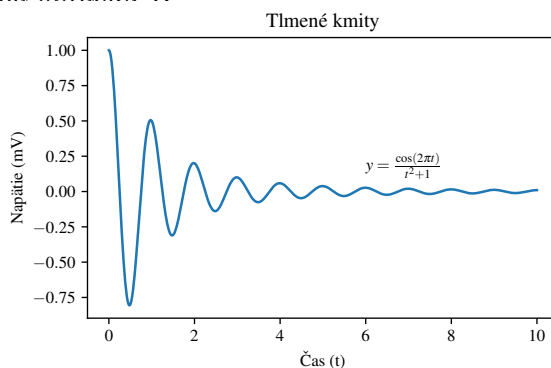
tak v dokumente môžeme príkazom `\py{a+b}` vložiť hodnotu 5.

3. Grafy funkcií s knižnicou matplotlib

Užitočnou funkcionalitou je možnosť vkladania grafov funkcií, vytvorených pomocou knižnice `matplotlib` do obrázkov v dokumente. Pre tieto potreby si musíme načítať aj knižnicu `numpy`, čo asi skúsených pythonistov neprekvapuje. Príslušný kód, ktorý vložíme do prostredia `pycode` potom bude vyzeráť takto:

```
from matplotlib.pyplot import *
from numpy import *
def f(t): return cos(2*pi*t)/(t**2+1)
t=linspace(0,10,500)
y=f(t)
clf()
figure(figsize=(5,3))
rc("text", usetex=True)
plot(t,y)
title(r'Tlmené kmity')
text(6,0.15,r"$y=\frac{\cos (2 \pi t)}{t^2+1}$")
xlabel("Čas (t)")
ylabel("Napätie (mV)")
savefig("tlmene.pdf",bbox_inches="tight")
print(r"\begin{figure}")
print(r"\includegraphics[width=0.65\textwidth]{tlmene.pdf}")
print(r"\caption{Graf funkcie z Python-u.}\label{obr:graf}")
print(r"\end{figure}")
```

Výsledok vidíme na obrázku 1.



Obr. 1. Graf funkcie z Python kódu.

4. Symbolické výpočty s knižnicou sympy

Symbolické výpočty sú neoddeliteľnou súčasťou sadzby matematických textov. Tu s výhodou využijeme knižnicu `sympy` jazyka Python. Na tomto mieste ilustrujeme len skrátenú verziu tabuľky integrálov a derivácií elementárnych funkcií:

$$\begin{array}{ll} \frac{d}{dx} \sin(x) = \cos(x) & \int \sin(x) dx = -\cos(x) \\ \frac{d}{dx} \cos(x) = -\sin(x) & \int \cos(x) dx = \sin(x) \\ \frac{d}{dx} \operatorname{tg}(x) = \operatorname{tg}^2(x) + 1 & \int \operatorname{tg}(x) dx = -\ln(\cos(x)) \end{array}$$

Do dokumentu, opäť do prostredia `pycode`, vložíme nasledujúci kód:

```
from sympy import *
var('x')
# Vytvoríme zoznam funkcií do tabuľky
funcs=['sin(x)', 'cos(x)', 'tan(x)']
print(r'\begin{align*}')
for func in funcs:
    # Vloženie vertikálnej medzery v prípade
    # cyklotrických a hyperbolických funkcií
    if func=='asin(x)' or func=='sinh(x)':
        print(r'&\\')
    myderiv='Derivative('+func+',x)'
    myint='Integral('+func+',x)'
    print(latex(eval(myderiv))+ '&=' +
            latex(eval(myderiv+'.doit()'))+ r'\quad & \quad')
    print(latex(eval(myint))+ '&=' +
            latex(eval(myint+'.doit()'))+ r'\\')
print(r'\end{align*}')
```

Všimnime si určitú disproporciu medzi zaužívaným označením niektorých funkcií v matematickej symbolike, kde napríklad funkciu tangens označujeme ako $\operatorname{tg} x$, kým syntax jazyka Python vyžaduje `tan(x)`, čo sa prejaví aj vo výstupe matematických vzťahov. Túto nezrovnalosť môžeme odstrániť predefinovaním pomocou príkazu `\renewcommand\tan{\mathop{\rm tg}\nolimits}`. Tak sme aj v tomto článku docielili adekvátne označenie funkcie tangens alebo pretypovanie prirodzeného logaritmu z pôvodného `log` na zaužívané `ln`.

Funkcia `latex` z knižnice `sympy` umožňuje automatizovať inak zdĺhavé úlohy. Namiesto manuálneho opisovania postupu integrácie krok za krokom alebo ich kopírovania z externého programu, ich používateľ môže automaticky generovať priamo v L^AT_EX-u. Napríklad výpočet dvojného integrálu v tvare:

$$\begin{aligned} \iint x \sin(x+y) dx dy &= \int (-x \cos(x+y) + \sin(x+y)) dy \\ &= -x \sin(x+y) - \cos(x+y) \end{aligned}$$

dostaneme po vložení kódu v podobe:

```
\begin{sympycode}
  x, y = symbols('x, y')
  f = x * sin(x+y)
  krok1 = Integral(f, x, y)
  krok2 = Integral(Integral(f, x).doit(), y)
  krok3 = krok2.doit()
\end{sympycode}
\begin{align*}
  \sympy{krok1} &= \sympy{krok2} \quad \&= \sympy{krok3}
\end{align*}
```

Takisto riešenie rovníc môžeme zautomatizovať pomocou funkcie `solve` z knižnice `sympy`. Do prostredia `sympycode` vložme napríklad nasledujúci kód:

```
x = symbols('x')
myeq = Eq(3*x**3+4*x**2+2*x-4,0)
print('Korene rovnice ')
print(latex(myeq, mode='inline'))
print(' sú')
print(latex(solve(myeq), mode='inline'))
```

Ako výstup dostaneme vyriešenú rovnicu.

Korene rovnice $3x^3 + 4x^2 + 2x - 4 = 0$ sú $[2/3, -1 - i, -1 + i]$.

5. Výpis zdrojového kódu

Okrem už skôr využitého zápisu zdrojového kódu s využitím balíčka `tcolorbox` prináša `PythonTeX` aj vlastné prostredie pre zobrazovanie zdrojového kódu. Výhodou je, že môžeme doplniť čísla riadkov a v texte dokumentu sa potom na tieto riadky odvolávať. Náš kód pre tabuľku derivácií a integrálov potom môžeme prezentovať v prostredí `sympyblock` takto:

Tabuľka derivácií a integrálov

```
1 from sympy import *
2 var('x')
3 # Vytvoríme zoznam funkcií
4 funcs=['sin(x)', 'cos(x)', 'tan(x)']
5 print(r'\begin{align*}')
6 for func in funcs:
7     # Úprava vertikálnej medzery pre cyklotrické
8     # a hyperbolické funkcie
9     if func=='asin(x)' or func=='sinh(x)':
10         print(r'&\\')
11     myderiv='Derivative('+func+',x)'
12     myint='Integral('+func+',x)'
13     print(latex(eval(myderiv))+ '&= ' +
14           latex(eval(myderiv+'.doit()'))+r'\quad & \quad')
```

```

15     print(latex(eval(myint))+'&='+
16           latex(eval(myint+'.doit()'))+ r'\\\')
17     print(r'\end{align*}')

```

6. Niektoré ťažkosti

Pri spracovaní tohto príspevku som narazil na niektoré problémy či nedostatky pri spracovaní slovenského alebo českého textu. Ide najmä o zaužívanú symboliku niektorých funkcií. Ako odstrániť vybrané problémy sme si už ukázali. Ale napríklad v prípade cyklometrických už takéto jednoduché pretypovanie nie je možné. Autor PythonT_EX-u totiž pre sadzbu označenia funkcií používa nástroj `\operatorname{}`, ktorý už nie je možné jednoducho zmeniť pomocou príkazu `\renewcommand`. Neostáva teda nič iné, len manuálne prehľadať pomocné súbory, ktoré vzniknú pri použití PythonT_EX-u a označenia prepísať.

Podobne, pri riešení rovníc si môžeme všimnúť, že imaginárna jednotka je zobrazená ako i , čo je v rozpore so zavedeným pravidlom sadzby vzpriameným rezom, teda i . S rovnakou nepresnosťou sa stretneme aj pri sadzbe symbolu pre Eulerovo číslo, kde opäť namiesto zavedeného vzpriameného e je použité e .

Navyše, v prípade numerického riešenia rovnice s knižnicou `numpy` dostaneme výsledok v takejto podobe:

$$\begin{aligned}
 &\text{Korene rovnice } 3x^3 + 4x^2 + 2x - 4 = 0 \text{ sú:} \\
 x_1 &= (-1.0000000000000002 + 1.0000000000000002j), \\
 x_2 &= (0.6666666666666664 + 0j), \\
 x_3 &= (-1.0000000000000002 - 1.0000000000000002j).
 \end{aligned}$$

Ako vidíme, tu sa pre označenie imaginárnej jednotky zachovalo označenie j používané v rámci syntaxe jazyka Python, na rozdiel od matematiky, kde je používané i . Oslovil som s týmito problémami autora PythonT_EX-u, avšak do uzávierky zborníka bez odozvy.

7. Záver

Aj keď balíček `pythontex` prináša pri sadzbe v slovenskom alebo českom jazyku určité komplikácie pri finalizácii výstupu, je možné ho hodnotiť ako veľkú pomoc pri vytváraní dokumentov obsahujúcich väčšie množstvo výpočtov. Navyše publikovanie v angličtine, čo sa stalo pri vedeckej práci štandardom, je takmer bezproblémové. Pretože ide o kombináciu dvoch nástrojov. Začiatovníkom pri vytváraní dokumentov v systéme L^AT_EX je možné odporučiť učebnice [8] alebo [5]. O čosi podrobnejšie zoznámenie so systémom ponúka [4] od autora typografického systému T_EX, v [2] alebo [6].

Pre záujemcov neznalých jazyka Python (ako aj samotný autor príspevku sa v tomto smere hodnotí ako večný začiatočník) možno odporučiť [9] alebo [3]. Niektoré ďalšie nápady a inšpirácie pre samotný balíček `pythontex` potom ponúka [1] a podrobný popis manuál k balíčku [7].

Veľkou výhodou tohto balíčka je, že spolupracuje aj s inými jazykmi, nielen s Python-om, ako by mohol evokovať jeho názov. Ako sa uvádza v manuáli [7], voliteľným argumentom `usefamily` je možné definovať prostredie rozličných programovacích jazykov. Od verzie 0.12 podporuje `pythontex` jazyky Ruby a Julia. Od verzie 0.13 pribudlo rozšírenie o Octave. Verzia 0.15 potom priniesla ďalšie rozšírenie o Burn shell `bash` a Rust. Konečne od verzie 0.17 je dostupné rozšírenie pre jazyky R, Perl, resp. Perl 6 a JavaScript.

Podakovanie. Príspevok vznikol s príspevom grantu KEGA 019ŽU-4/2023 „Inovatívne učenie matematiky s podporou Open Source“ podporeného Slovenskou kultúrno-edukačnou grantovou agentúrou.

Literatúra

- [1] ESSLINGER, B. & STUDENTS: *PythonTeX and L^AT_EX – Familiarize us with PythonTeX in order to build the Cryptool Book*, online dokument <https://www.cryptool.org/download/ctb/PythonTex-by-Examples.pdf>.
- [2] GRÄTZER, G.: *Text and Math Into L^AT_EX*, 6th ed., Springer Nature Switzerland AG, Cham, 2024, ISBN 978-3-031-55280-9, e-book: <https://doi.org/10.1007/978-3-031-55281-6>.
- [3] HUNT, J.: *A Beginners Guide to Python 3 Programming*, 2nd ed., Springer Nature Switzerland AG, Cham, 2023, ISBN 978-3-031-35121-1, e-book: <https://doi.org/10.1007/978-3-031-35122-8>.
- [4] KNUTH, D. E.: *The T_EXbook*, Volume A of *Computers and Typesetting*, Addison-Wesley Publishing Company (1984), ISBN 0-201-13448-9.
- [5] KOPKA, H. – DALY, P. W.: *L^AT_EX – Podrobný průvodce*, Brno, Computer Press, 2004, ISBN 80-722-6973-9.
- [6] ÖCHSNER, M. – ÖCHSNER, A.: *Advanced L^AT_EX in Academia*, Springer Nature Switzerland AG, Cham, 2021, ISBN 978-3-030-88955-5, e-book: <https://doi.org/10.1007/978-3-030-88956-2>.
- [7] POORE, G. M.: *The pythontex package*, dokumentácia balíčka PythonT_EX, dostupné online <http://mirrors.ctan.org/macros/latex/contrib/pythontex/pythontex.pdf>.
- [8] RYBIČKA, J.: *L^AT_EX pro začátečníky*, Brno, KONVOJ 2003, ISBN 80-7302-049-1.
- [9] SCHÄFER, CH.: *Quickstart Python*. Springer essentials, Wiesbaden, Germany, 2019, ISBN 978-3-658-33551-9, e-book: <https://doi.org/10.1007/978-3-658-33552-6>

Kontaktná adresa

RNDr. Aleš Kozubík, PhD., Katedra matematických metód, Fakulta riadenia a informatiky, Žilinská univerzita, Univerzitná 8215/1, 010 26 Žilina, Slovenská republika,
E-mailová adresa: alesko@frcatel.fri.uniza.sk

R BALÍČKY URČENÉ K PRÁCI S GEODATY A JEJICH VYUŽITÍ V PROJEKTU PERUN

ONDŘEJ LEDVINKA (CZ)

Abstrakt. Příspěvek představuje současné možnosti zpracování geodat funkcemi vysokoúrovňového programovacího jazyka R. V krátkosti upozorňuje na funkce navržené pro manipulaci, tvorbu a analýzu vektorových geodat. Stěžejní část se však věnuje práci s rastrovými geodaty, neboť ta tvoří podstatu projektu PERUN, ať již jde o odhad budoucího vývoje klimatologických charakteristik prostřednictvím scénářů modelu ALADIN-CLIMATE/CZ Českého hydrometeorologického ústavu, či o dovozování dalších veličin potřebných pro hydrology zabývající se historickým a budoucím vývojem vodních zdrojů na území Česka. Na příkladech geodat, se kterými v rámci projektu autor pracuje, jsou vybrány některé tipy, které se mohou hodit začátečníkům i pokročilým uživatelům jazyka R.

Klíčová slova. Vektorová geodata, rastrová geodata, tidyverse, funkcionální programování.

R PACKAGES FOR WORKING WITH GEODATA AND THEIR USE IN THE PERUN PROJECT

Abstract. The paper presents current possibilities of geodata processing by functions of the high-level programming language R. In short, it draws attention to functions designed for manipulation, creation and analysis of vector geodata. However, the substantial part is devoted to work with raster geodata, as they form the essence of the PERUN project, whether it is to estimate the future development of climatological characteristics through scenarios of the Czech Hydrometeorological Institute's ALADIN-CLIMATE/CZ model, or to derive other quantities needed for hydrologists dealing with the historical and possible future development of water resources in the territory of Czechia. Some tips are selected on examples of geodata that the author works with within the project, which may be useful for beginners and advanced users of the R language.

Keywords. Vector geodata, raster geodata, tidyverse, functional programming.

1. Úvod

Vysokoúrovňový programovací jazyk R byl původně navržen pro účely statistických výpočtů a analýz a zároveň pro kreslení zásadních rysů výsledků či vystižení podstatných vlastností a chování dat. Postupně si tento jazyk našel mnohé oblasti svých aplikací, o čemž svědčí i úctyhodný počet přídatných balíčků, kterých je v současnosti něco přes 22 tisíc, pokud tedy jde jen o ty zveřejněné na zrcadlech CRAN (Comprehensive R Archive Network) a nepočítají se jiné repozitáře. Jeli-kož empirická data přirozeně mívají vztah k místu sběru, bylo jen otázkou času, kdy se mezi balíčky objeví nástroje připomínající geografické informační systémy

(GIS) s možnostmi tzv. geoprocessingu. Spojí-li se potenciál takto promyšleného zacházení s geoprostorovými daty (geodaty) s nabídkou stavět si vlastní skripty především z důvodů automatizace nebo aplikace stejné funkce na velké množství jednotek najednou (či naopak více funkcí na jednotku) prostřednictvím tzv. for cyklů, vzniká velmi šikovný pomocník, usnadňující život nejednomu analytikovi, který by se jinak utápěl v současném světě nepřehledného kvanta dostupných a stále se otevírajících dat. To vše R poskytuje a je tedy vhodné se tomuto tématu seriózně věnovat.

Vzhledem k tomu, že i podmnožina balíčků určených pro práci s geodaty v R je poměrně bohatá a sběr informací pro tzv. CRAN Task Views je dobrovolný (geodatům nejbližší zřejmě ty na [1] nebo [8]), není možné postihnout na limitovaném počtu stránek příspěvku celý tento svět a je nutné se zabývat jen určitou částí. Vhodný odrazový můstek přitom představuje český projekt PERUN, který byl spuštěn v červenci 2020, má končit v prosinci 2026 a jehož stěžejní dílčí cíle se týkají tvorby budoucích klimatických scénářů, jejich korekcí a aplikací (např. v klimatologii a hydrologii). Následuje tedy přehled základních balíčků pro práci s vektorovými a rastrovými geodaty v R, přičemž zpracování rastrových geodat je upřednostněno z důvodu dominance těchto dat v projektu PERUN, na kterých mohou být demonstrovány ukázky.

Autor si klade za cíl ukázat několik tipů, které pomohou urychlit proces učení zájemců. Zejména půjde o tipy pro práci s geodaty v R, které mohou i po přečtení patřičných návodů k funkcím začátečníkům snadno uniknout. Čtenář nechť naopak neočekává přehled funkcí určených pro kreslení statických a dynamických map prostřednictvím R s využitím načtených geodat. Tomuto tématu se věnuje jiný příspěvek.

2. Vektorová geodata v současném R

Poté, co se jeden z neplodnějších programátorů R balíčků zaměřených na práci s geodaty Roger Bivand rozhodl, že v roce 2023 odejde do důchodu a tímto skončí další vývoj a podpora balíčků `rgdal`, `rgeos` a `maptools` (viz např. [9]), je nepochybně v současnosti nejdůležitější sadou funkcí pro tvorbu, načítání (i z databází), manipulaci a ukládání vektorových geodat balíček `sf` (z angl. *simple features*; [11, 10]). Takto lze pracovat s atributovými tabulkami připomínajícími klasické datové rámce (*data frames*), které jsou navíc obohacené o sloupec s geometrií, jež je v něm pro každý řádek (tj. *feature*) uchována formou tzv. *well-known textu* (WKT) za využití faktu, že sloupcem datového rámce může být i seznam. Balíček `sf` nabádá při skriptování k aplikacím přístupu *tidyverse* [13, 14], takže lze využívat především jeho slovesa i při manipulaci s geometrií. Samozřejmostí je podpora zápisu s tzv. pipe operátory (od R verze 4.1.0 hlavně nativní pipe `|>` namísto `pipu %>%` vypůjčeného z balíčku `magrittr`) a zejména vylepšeného datového rámce známého pod pojmem *tibble*, který naskýtá lepší kontrolu nad třídami

sloupců tabulky, ale také mnohem lepší tisk části dat pro nahlédnutí do konzole (včetně zvýraznění chybějících hodnot či záporných čísel odlišnou barvou). Velkou výhodou je však podpora funkcionálního programování s vektorizovanými funkcemi, jež do značné míry nahrazuje for cykly.

Balíček **sf** rozeznává hlavně tzv. velkou sedmičku typů geometrií, ale možností je mnohem více. Balíček zakládá na závislostech a integrování několika externích knihoven. Načítání, ukládání geodat a i některé operace s nimi probíhají přes knihovnu GDAL, vztahy mezi geometriemi a dotazy nad nimi (včetně výpočtu měr v rovině) jsou řešeny přes knihovnu GEOS. Podpora souřadnicových referenčních systémů (CRS) je umožněna propojením s knihovnou PROJ, přičemž zde je upřednostněno odkazování se na CRS pomocí WKT řetězců, což nevylučuje odkazy typu AUTORITA:KÓD. Avšak tzv. PROJ4 řetězce jsou považovány za zastaralé, protože je doporučováno se jim již vyhýbat. Propojení s Google knihovnou S2 podporuje výpočty na sféře. V neposlední řadě stojí za zmínku, že vývojáři balíčku **sf** si potrpí na upřesňování jednotek, u nichž ale existuje možnost je zahazovat. Podobně je to i s geometrií, po jejímž zahození vzniká obyčejný datový rámec s pouhými atributy. Poněkud zrádné mohou být zkratky používané v nápovědách k funkcím: **sfg** značí třídu *simple feature geometry* a **sfc** pro změnu *simple feature column*, tedy geometrický sloupec, ke kterému již lze přidávat atributy. Setkáme se též se zkratkou **agr**, která může připomínat jakousi agregaci, ale její význam zní *attribute-geometry relationship*, což prozrazuje, že se dbá na vztahy mezi atributy a geometrií, a to právě kvůli různým operacím s geometrií, jako je např. řezání polygonů čarami.

Vektorová geodata mají v projektu PERUN podružný význam. Dalo by se říci, že slouží spíše ke kontrolám vykreslováním jednoduchých map a k omezování se na území Česka a jeho okolí. V hydrologických analýzách však nacházejí svá uplatnění v kombinaci s rastrovými geodaty, kdy je např. nutné z hodnot rastrových buněk získat hodnoty agregované přes plošné jednotky, jako jsou povodí nad vodoměrnými stanicemi. Jinou aplikací mohou být prostorové interpolace geostatistickými funkcemi, které však autor příspěvku většinou přenechává jiným řešitelům projektu PERUN. Výjimkou byla tvorba týdenních časových řad gridů vodní hodnoty sněhu na území Česka pro zimní období mezi lety 1980 a 2023 [2].

3. Rastrová geodata v současném R

Stran zpracování rastrových geodat dnes v R všemu bez debat vévodí balíček **terra** [4], který od roku 2020 nahrazuje svého předchůdce s názvem **raster**. Mnoho funkcí balíčku **terra** je ve skutečnosti implementováno v jazyku C++, a proto jsou mnohem rychlejší než jejich protějšky pocházející z balíčku **raster**. Jsou zde velmi dobře podporovány anonymní funkce, a to především ve funkcích typu **app()** a **tapp()**, jejichž názvy připomínají základní funkce R nazvané

`apply()` nebo `tapply()`. Tím ale výčet těchto funkcí nekončí, existují zde i funkce příbuzné základním funkcím `lapply()`, `sapply()` apod. (tedy `lapp()` a `sapp()`).

Je třeba poznamenat, že pokud se někdo rozhodne aplikovat funkci, která není implementovaná v C++, jistě narazí na značné zpomalení výpočtů. Pro tyto účely vývojáři balíčku **terra** umožňují rozložit proces na více logických jader pracovní stanice, což má většinou za následek urychlení práce. Ovšem velmi záleží na konkrétním úkolu, struktuře dat, na stylu napsání využívané funkce (např. jestli je vektorizovaná) a samozřejmě na konfiguraci samotné pracovní stanice, jako je počet logických jader nebo kapacita RAM. Proto je před rutinním využíváním balíčku **terra** nezbytné počáteční experimentování. Teprve po rozpoznání typů úloh a nejlepšího způsobu jejich zpracování na té které pracovní stanici dochází k rapidnímu navrácení času investovaného do prvotních pokusů.

Práce s RAM zde hraje důležitou roli. Rastrová geodata jsou často tak velká, že se do paměti běžného osobního počítače ani nevejdou. Je proto třeba vymýšlet různé strategie, jak k takto velkým datům přistupovat. Jednou z nich je poctivě ukládání si pracně vytvořených výsledků, nebo dokonce mezivýsledků, do souborů na disku. V tomto **terra** závisí na knihovně GDAL (např. GeoTIFF soubory) anebo i jiných knihovnách, jako je např. **netcdf** pro ukládání v klimatologii velmi oblíbených NetCDF souborů. Pokud do souboru výsledky neukládáme rovnou, tak jako tak se k objektům v Globálním prostředí často přiřazují jen tzv. pointery (odkazy) na dočasné soubory, které **terra** ukládá za nás na místo dané funkcí `terraOptions()`, aniž bychom o tom vůbec museli zpočátku vědět. Skutečnost, že se objekt odkazuje na soubor, poznáme tak, že se vedle jeho názvu v Globálním prostředí objevuje sousloví *Formal class SpatRaster* namísto *SpatRaster*, což je třída pro rastrová geodata balíčku **terra** (vedle třídy *SpatVector* pro vektorová geodata). Proto není doporučováno objekty z Globálního prostředí v případě geodat ukládat do velmi známých RData souborů, po jejichž otevření stejně zjistíme, že v takových souborech se žádná rastrová geodata nenachází.

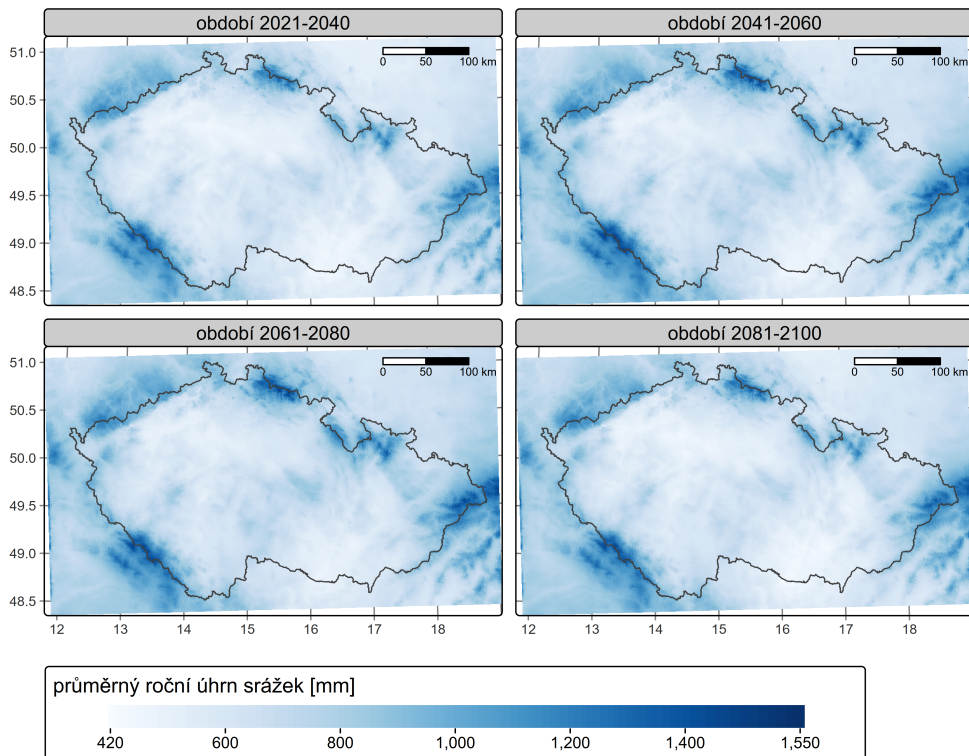
Načítání a ukládání rastrových geodat vyžadují speciální poznámky. Pro načítání všech typů souborů, lhostejno, zda GeoTIFF, nebo NetCDF (dokonce je jedno, zda s jednou, nebo s více vrstvami, pokud je splněn určitý limit na počet vrstev), slouží funkce `rast()`. Těto funkci ani nevádí načítat více souborů z adresáře najednou, když budou stejného typu (např. GeoTIFF), případně ani z různých adresářů či podadresářů, což lze zajistit předchozím připravením si vektoru cest k takovým souborům. Funkce `rast()` dokonce umí konvertovat na třídu *SpatRaster* z jiných tříd, jako jsou pole, seznamy nebo datové rámce. Najdeme zde i vlastnost mazání hodnot rastrových buněk a ponechání pouhé geometrické konstrukce rastru, což se hodí před přiřazením nových hodnot. Pro ukládání rastrových geodat naopak existuje více funkcí. V projektu PERUN jsou pro tyto účely nejvíce využívány funkce `writeRaster()` a `writeCDF()`. Tyto funkce existují odděleně patrně z důvodu využívání různých externích knihoven (viz výše) a podpory různého chování ukládaných souborů. Nežřídká se stává, že uživatelé

vyžadují každý bod časové řady gridů mít uložen do separátního souboru. Zřejmě je to proto, že jejich softvér, který používají k dalšímu zpracování, neumí načítat více vrstev najednou, nebo je to v něm velmi nešikovné. V projektu PERUN byly testovány různé strategie pro uložení individuálních vrstev do souborů. Byla k tomu zaměstnána i tzv. paralelizace mapovacích funkcí prostřednictvím tzv. *futures*, což umožňuje R balíček *furrr* [12]. Ovšem při lokálním zpracování bylo ve smyslu rychlosti bezkonkurenční zaměřit se na nepatrnou poznámku v nápovědě k funkci `writeRaster()`, a sice, že, pokud bude jako argument cest nových souborů nastaven vektor o stejném počtu prvků, jako je počet vrstev v původním *SpatRaster* objektu, dojde k sepnutí trhání do souborů implementovanému v C++.

Nedostatek balíčku *terra* lze spatřovat v nízké podpoře ovládání ve smyslu *tidyverse*. Funkce balíčku *terra* připomínají spíše funkce základního R. Je ale třeba podotknout, že z důvodu překlenutí tohoto nedostatku vzniknul další R balíček *tidyterra* [3], díky němuž lze na třídy balíčku *terra* aplikovat celou škálu funkcí známých ze světa *tidyverse*.

Náleží sem rovněž podpora kreslení rastrových geodat ve smyslu funkcí balíčku *ggplot2* (viz např. `tidyterra::geom_spatraster()`), který je součástí jádrového *tidyverse*. Někdy je však zapotřebí dávat si pozor na výkonnost některých takových funkcí. Při zpracování rastrových geodat s velkým počtem vrstev bylo upozorováno, že výběr vrstev pomocí funkce `tidyterra::select()` je velmi pomalý oproti nativnímu *terra* postupu s dvěma hranatými závorkami. Dobrou zprávou je, že objekty vzniklé funkcemi balíčku *terra* lze naskládat do sloupce datového rámce (lépe řečeno *tibble*) jako seznam (např. po aplikaci funkce `terra::as.list()`) a lze tedy na ně velmi dobře aplikovat funkcionální programování ve smyslu balíčků *purrr* či *furrr*. Při paralelizaci se ovšem ukazuje nedostatek implementace funkcí v C++, a sice, že bez načtení dat do paměti funkcí `terra::wrap()` zatím nelze informovat pracovníky clusteru o *SpatRaster* objektech, se kterými mají pracovat. Zdá se ale, že dosavadní vývoj balíčku *terra* má tendenci tuto mezeru vyplnit.

Jde-li o další *terra* funkce využívané v projektu PERUN, patří mezi ně zcela jistě funkce `extract()`, která umožňuje redukovat hodnoty rastru pro místa jednotek danými vektorovými geodaty. Mohou tak rychle vznikat časové řady srážkových úhrnů či teploty vzduchu pro jednotlivá povodí nad vodoměrnými stanicemi pro porovnání s časovými řadami průtoků [5] nebo pro hydrogeologické rajony z důvodu výzkumu reagování vydatnosti pramenů na klimatickou změnu [7, 6]. Užitečnou se ukázala být funkce `tapp()`, jež dokáže rychle dodat agregace pro časové jednotky (např. při přechodu z denního časového kroku na měsíční), což je velmi chtěná úloha. Vizualizaci průměrných ročních srážkových úhrnů na území Česka a jeho okolí podle modelu ALADIN-CLIMATE/CZ (scénáře SSP2-4.5) ve čtyřech budoucích obdobích pak znázorňuje obr. 1.



Obrázek 1. Průměrný roční úhrn srážek na území Česka a jeho okolí ve čtyřech budoucích obdobích podle modelu ALADIN-CLIMATE/CZ (scénář SSP2-4.5; zdroj: <https://www.perun-klima.cz/scenare/index.php>)

4. Závěr

Kombinace GIS funkcí a skriptování nabízí širokou škálu aplikací hlavně tam, kde není kladen důraz na vizuální stránku jako spíše na zpracování velkého množství geodat a vytěžení syntetické informace z více jednotek najednou. Jazyk R kombinuje tyto možnosti i s funkcionálním programováním (např. prostřednictvím přístupu *tidyverse*), které tvoří výborný základ pro aplikování metod strojového učení. Paralelizace procesů nebo implementace některých funkcí v jazyku C++ značně snižují dobu zpracování. Při analýze geodat v R je však třeba pamatovat na několik úskalí, a to především v případě rastrových geodat. Předně je vhodné se odnaučit ukládat objekty celého Globálního prostředí do RData souborů a serializovat jen vybrané objekty, které skutečně měly co dočinění s nahráním dat do paměti pracovní stanice (např. do souborů RDS). Další velmi doporučenou strategií je ukládat si výsledky a mezivýsledky, jejichž zdrojem jsou dočasné soubory, které mizí po končení každého sezení, do stálých souborů

(např. GeoTIFF nebo NetCDF). V neposlední řadě je nutné si ověřit co nejrychlejší způsob zpracování individuálně na té které stanici s daným počtem logických jader a nějakou kapacitou RAM. Rovněž tak se musí volit vhodný postup podle toho, jak zní zadání úlohy, jaká data máme právě po ruce a jestli funkce, které známe, jsou vektorizované či ne. Pokud je funkce vektorizovaná, lze proces snadno paralelizovat (např. s využitím anonymních funkcí), ale u rastrových dat jsme zatím nuceni před paralelizací je nahrát do paměti, což může při lokálních aplikacích činit nemalé potíže. Proto je dobré zauvažovat i o rychlosti postupů s funkcemi implementovaných v C++ a všechny možnosti srovnávat (např. měřením spotřebovaného času).

Jazyk R je obecně dosti pomalý a lze tedy reálně uvažovat o jeho kombinacích s externími knihovnami či dokonce s funkcemi externích GIS, jako je QGIS, SAGA GIS nebo GRASS GIS. To vše je možné přes specializované R balíčky. Při zpracování velkých rastrových geodat se lze poohlédnout i po jiném jazyce, jako je např. Julia. Při tomto jazyce najdeme velmi agilní skupinu programátorů, díky nimž vzniká čím dál více atraktivních balíčků pro zpracování geodat. Nechme se tedy překvapit, co přinese budoucnost.

Poděkování. Tento příspěvek vznikl s podporou Technologické agentury České republiky (TA ČR), projektu SS02030040 „Predikce, hodnocení a výzkum citlivosti vybraných systémů, vlivu sucha a změny klimatu v Česku (PERUN)“.

Reference

- [1] BIVAND, R. – NOWOSAD, J.: *CRAN Task View: Analysis of Spatial Data*, 2025, <https://CRAN.R-project.org/view=Spatial> [cit. 21.6.2025].
- [2] FERČÁK, J.: *Variability and Change of Snow Storage in Czechia at Different Elevations*, Bachelor Thesis, 2025, Charles University, Faculty of Science, Prague.
- [3] HERNANGÓMEZ, D.: *Using the Tidyverse with Terra Objects: The Tidyterra Package*, month = nov, Journal of Open Source Software, 2023, Vol. 8, Nr. 91, pp. 5751, ISSN 2475-9066, dostupné z DOI 10.21105/joss.05751 [cit. 20.4.2024].
- [4] HIJMANS, R. J.: *Terra: Spatial Data Analysis*, 2025, howpublished = <https://CRAN.R-project.org/package=terra>.
- [5] JENÍČEK, M. – NEDĚLČEV, O. – TYL, R. – LEDVINKA, O. – VAJSKEBR, V. – ŠERCL, P. – BERNSTEINOVÁ, J. – LANGHAMMER, J. – UNUCKA, J.: *CAMELS-CZ: Databáze atributů povodí pro hydrologické a klimatologické studie na velkém vzorku malých povodí*, První konference PERUN, 2023, Praha, Český hydrometeorologický ústav, ISBN 978-80-7653-063-8, s. 99–105, <https://www.chmi.cz/files/portal/docs/reditel/SIS/doi/Perun/2023/978-80-7653-063-8.14.pdf> [cit. 6.1.2025].
- [6] LAMAČOVÁ, A. – LEDVINKA, O. – BOHDÁLKOVÁ, L. – OULEHLE, F. – KREISINGER, J. – VLNAS, R.: *Response of Spring Yield Dynamics to Climate Change across Altitude Gradient and Varied Hydrogeological Conditions*, Science of The Total Environment, 2024, Vol. 921, ISSN 0048-9697, dostupné z DOI 10.1016/j.scitotenv.2024.171082 [cit. 11.4.2024].
- [7] LEDVINKA, O. – LAMAČOVÁ, A.: *Změny ve vydatnosti pramenů na území Česka v letech 1971–2020*, První konference PERUN, 2023, Praha, Český hydrometeorologický ústav,

- ISBN 978-80-7653-063-8, s. 175–178, <https://www.chmi.cz/files/portal/docs/reditel/SIS/doi/Perun/2023/978-80-7653-063-8.22.pdf>, [cit. 6.1.2025].
- [8] PEBESMA, E. – BIVAND, R.: *CRAN Task View: Handling and Analyzing Spatio-Temporal Data*, 2022, <https://CRAN.R-project.org/view=SpatioTemporal> [cit. 21.6.2025].
 - [9] PEBESMA, E. – BIVAND, R.: *R-Spatial Evolution: Retirement of Rgdal, Rgeos and Maptools / R-bloggers*, 2022, <https://www.r-bloggers.com/2022/04/r-spatial-evolution-retirement-of-rgdal-rgeos-and-maptools/> [cit. 21.6.2025].
 - [10] PEBESMA, E. – BIVAND, R.: *Spatial Data Science: With Applications in R*, The R Series, 2023, CRC Press, Boca Raton, ISBN 978-0-429-45901-6.
 - [11] PEBESMA, E.: *Simple Features for R: Standardized Support for Spatial Vector Data*, The R Journal, 2018, Vol. 10, Nr. 1, pp. 439–446, ISSN 2073-4859, dostupné z DOI 10.32614/RJ-2018-009 [cit. 7.3.2021].
 - [12] VAUGHAN, D. – DANCHO, M.: *Furrr: Apply Mapping Functions in Parallel Using Futures*, 2022, <https://CRAN.R-project.org/package=furrr> [cit. 21.6.2025].
 - [13] WICKHAM, H. – AVERICK, M. – BRYAN, J. – CHANG, W. – MCGOWAN, L. – FRANÇOIS, R. – GROLEMUND, G. – HAYES, A. – HENRY, L. – HESTER, J. – KUHN, M. – PEDERSEN, T. – MILLER, E. – BACHE, S. – MÜLLER, K. – OOMS, J. – ROBINSON, D. – SEIDEL, D. – SPINU, V. – TAKAHASHI, K. – VAUGHAN, D. – WILKE, C. – WOO, K. – YUTANI, H.: *Welcome to the Tidyverse*, Journal of Open Source Software, 2019, Vol. 4, Nr. 43, pp. 1686, ISSN 2475-9066, dostupné z DOI 10.21105/joss.01686 [cit. 7.3.2021].
 - [14] WICKHAM, H. – ÇETINKAYA-RUNDEL, M. – GROLEMUND, G.: *R for Data Science: Import, Tidy, Transform, Visualize, and Model Data*, 2023, 2nd, O'Reilly Media, Inc, Sebastopol, CA, ISBN 978-1-4920-9740-2.

Kontaktní adresa

Mgr. Ondřej Ledvinka, Ph.D., Oddělení hydrofondu a bilancí, Úsek hydrologie, Český hydrometeorologický ústav, Na Šabatce 2050/17, 143 06 Praha 4 – Komořany, Česká republika,
E-mailová adresa: ondrej.ledvinka@chmi.cz
 Katedra fyzické geografie a geoekologie, Přírodovědecká fakulta, Univerzita Karlova, Albertov 6, 128 00 Praha 2, Česká republika,
E-mailová adresa: ledvinka@natur.cuni.cz

POKUS O OPTIMÁLNE RIEŠENIE ÚLOHY OBCHODNÉHO CESTUJÚCEHO S DRONOM

ŠTEFAN PEŠKO (SK) A ZUZANA BORČINOVÁ (SK)

Abstrakt. V súčasnosti máme veľa správ o vojnovom využití dronov. My sa chceme podeliť o naše úvahy a prvý pokus o exaktné riešenia ich využitia pri hľadaní optimálnych zásobovacích trás jedného vozidla s jedným dronom. Tieto NP-ťažké úlohy sa v praxi riešia prevažne heuristicky, no v posledných rokoch aj exaktne. Domnievame sa, že prichádza do úvahy aj ich kompletné riešenie súčasnými efektívnymi MILP riešičmi.

Kľúčové slová. Úloha obchodného cestujúceho, dron, model MILP.

AN EFFORT TO OPTIMAL SOLUTION OF THE TRAVELING SALESMAN PROBLEM WITH DRON

Abstract. Currently, we have many reports about the military use of drones. We want to share our thoughts and first attempt at exact solutions for their use in finding optimal supply route for vehicle with one drone. These NP-hard problems are in practice solved mostly heuristically, but in recent years also exactly. We believe that their complete solution with current efficient MILP solvers is possible.

Keywords. Traveling salesman problem, dron, model MILP.

Motivácia

K optimalizácii okružných trás vozidiel s dronmi nás prilákalo množstvo článkov z posledných rokov 2006–2024. Podnetom však bolo efektívne riešenie klasickej úlohy obchodného cestujúceho pomocou modelu Gavish-Graves [5] z roku 1978, ktorý sme medzi známymi MILP modelmi nenašli. Benchmarkovú inštanciu s 225 mestami MILP riešiteľ Gurobi version 12.0.0 vyriešil exaktne za 12,95 min. A tak nás napadlo využiť túto možnosť pri niektorej z dronových verzií.

Poznamenajme, že pútavý prehľad histórie TSP možno získať z voľne dostupnej monografie z roku 1957 [3], ktorá ešte i po rokoch poteší nielen laikov. Profesionálom je určená monografia [2], ktorá už vyžaduje solídne základy z diskkrétnej matematiky.

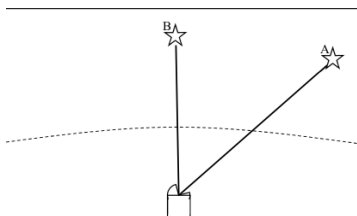
1. Úlohy obchodného cestujúceho

Klasickú úlohu obchodného cestujúceho (TSP – Traveling Salesman Problem) môžeme formulovať takto: *Je daná množina n miest $N = \{0, 1, \dots, n-1\}$ a tabuľka resp. matica vzdialeností medzi mestami $D = (d_{ij})$, $i, j \in N$. Úlohou obchodného*

cestujúceho je začať pochôdzku v niektorom východnom meste (napr. 0), navštíviť všetky ostatné mestá a vrátiť sa do východzieho mesta tak, aby prešiel minimálnu celkovú vzdialenosť.

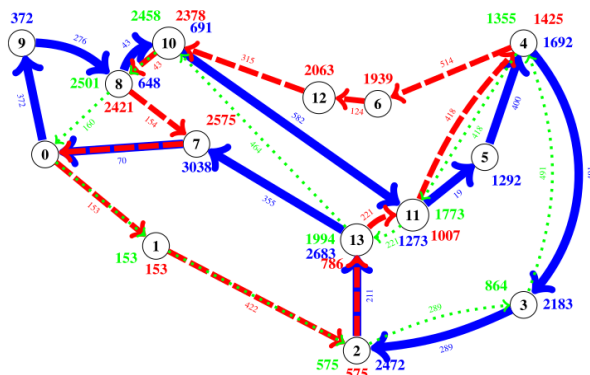
Úloha TSP má množstvo variánt, spomeňme apoň dve menej známe:

- Astro-TSP [6] – Vzdialenosti medzi pozorovanými objektami sa merajú v sekundách. Navyše sa požadujú časovo závislé dĺžky pozorovania, obmedzené pozorovacie okná a priority pozorovania. Autor navrhol riešenie založené na genetickom algoritme a riešil inštancie od 78 po 137 objektov (v rolách miest).



Obr. 1. Pozorovanie objektov A a B

- Consistent-TSP [4] – TSP tu má požiadavky zákazníkov na obsluhu na viaceré dni. Požaduje sa, aby bol zákazník obsluhovaný, pokiaľ možno, v tom istom čase s maximálnou možnou odchýlkou (napr. jedna hodina). Na obrázku 2 máme optimálne riešenie trojdennnej inštancie, kde tri farby zodpovedajú trasám TSP v troch dňoch. Autori diskutujú viaceré praktické modifikácie tohoto modelu, napr. s časovými oknami. Navrhujú tokový MILP model s využitím Bendersovej dekompozície pre väčší počet zákazníkov, resp. dní.

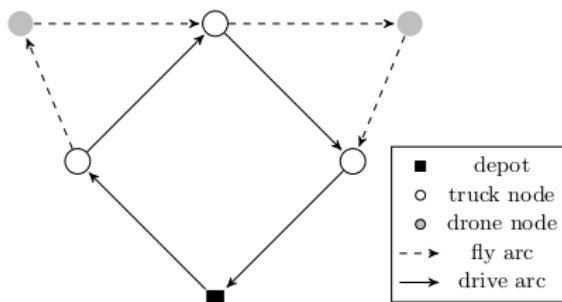


Obr. 2. Tri optimálne trasy TSP

2. Úlohy TSP s dronmi

Viacero internetových predajcov a poskytovateľov logistických služieb vrátane Amazonu, singapurskej pošty a DHL (sledovanie zásielok) experimentuje s používaním dronov na podporu doručovania balíkov a pošty. Jedným sľubným novým operačným modelom je použitie bežného doručovacieho vozidla, ktoré spolupracuje s dronom na doručovaní. Týmto spôsobom je možné kombinovať vysokú kapacitu vozidla a dlhý dolet, rýchlosť a flexibilitu dronu [1].

Efektívne spojenie dronu a nákladného vozidla vedie k novému problému plánovania, ktorý sa nazýva Problém obchodného cestujúceho s dronom (*Traveling Salesman Problem with Drone*) **TSP-D**. Cieľom problému je nájsť trasu pre doručovacie vozidlo aj dron, ktorá minimalizuje celkový spoločný čas na splnenie všetkých doručovacích úloh. Vzhľadom na svoju obmedzenú kapacitu sa dron musí pred každým novým dorúčením vrátiť k vozidlu, aby vyzdvihol balík. Z tohto dôvodu je potrebné trasu dronu synchronizovať s trasou nákladného vozidla.



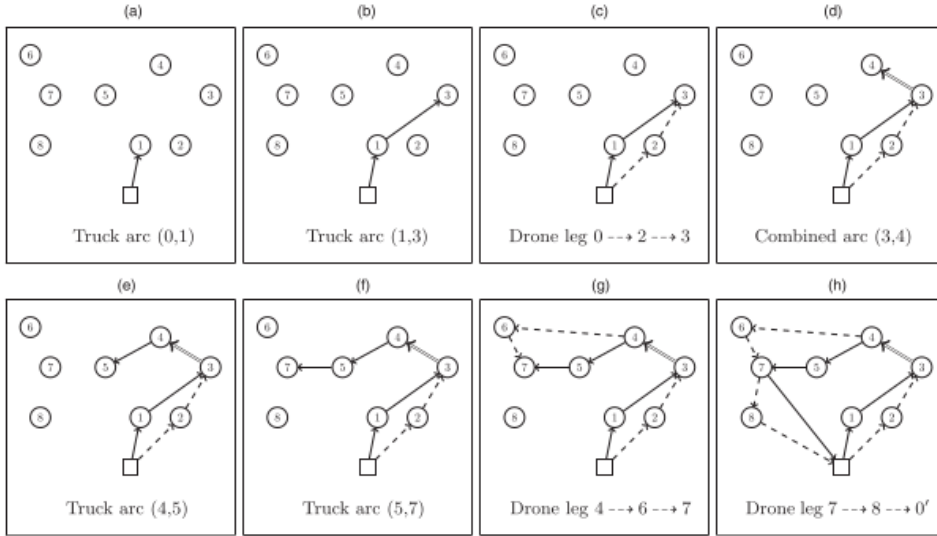
Obr. 3. Príklad prípustného riešenia TSP-D [1]

Obrázok 3 zobrazuje riešenie TSP-D, kde vozidlo pred obsluhou prvého adresáta vyšle dron, ktorý sa vráti k vozidlu u druhého adresáta atď, s konečným návratom k poslednému adresátovi a potom, na vozidle, do domovského depa. Prípustné riešenie vyžaduje koordináciu vozidla s dronom v uzloch dopravnej siete. Autori navrhli hľadať optimálne riešenie pomocou dynamického programovania, ktoré však poskytuje riešenie len s malým počtom obsluhovaných uzlov $n \leq 20$.

Z nedávnych prác v [11] autori navrhujú zmiešaný celočíselný lineárny model **MILP** pre niekoľko variantov TSP-D, ktoré sú založené na včasnej synchronizácii tokov nákladných vozidiel a dronov. Experimentujú tiež s rekurziou dynamického programovania v metóde vetiev a cien.

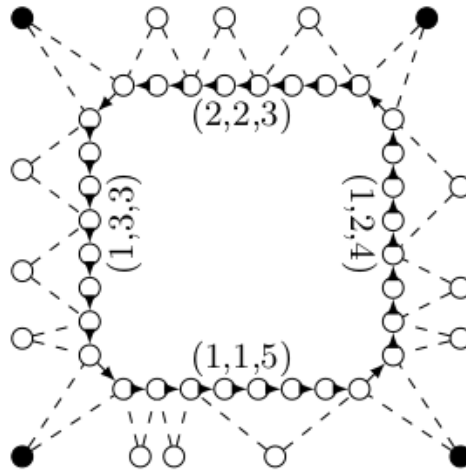
Na obrázku 4 máme zobrazenú postupnú tvorbu v jednej vetve riešenia. Pre-rušovaná čiara zodpovedá letu drona a plná jazde vozidla.

Štúdia [14] ponúka heuristické riešenie, ktoré umožňuje väčšie medzery, reprezentované rozkladmi, medzi odletmi a príletmi dronu. Vozidlo má obmedzenú kapacitu, a tak pripúšťa viacnásobný návrat do depa pre nakládku prepravovaného



Obr. 4. Tvorba prípustného riešenia TSP-D [11]

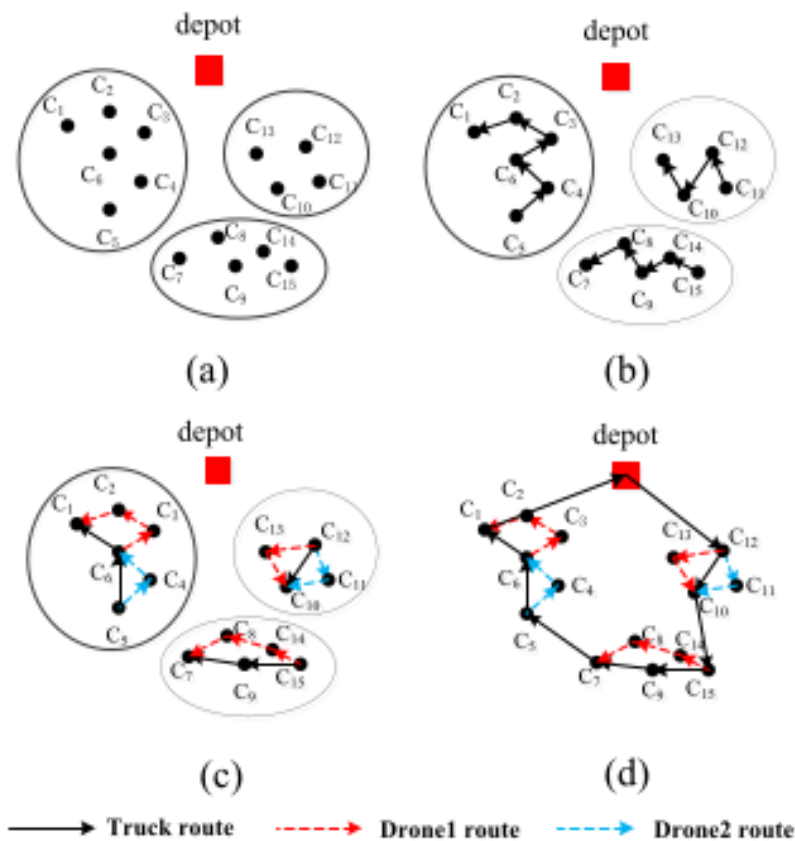
tovaru. Obmedzené programovanie (*Constraint Programming* – CP) umožnilo riešenie úloh až s $n = 100$ uzlami za 680 sekúnd.



Obr. 5. TSP-D s rozkladmi uzlov obsluhy [14]

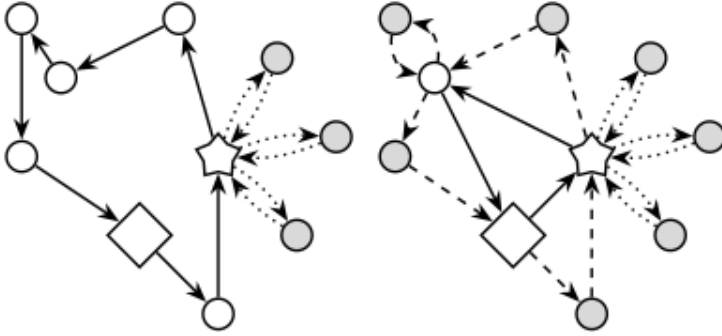
Pozoruhodnú aplikáciu našla TSP-D v čase covidu [7], prirodzeným zovšeobecnením TSP-D na **MTSP-MHD** (*Multiple visits Traveling Salesman Problem with Multiple Heterogeneous Drones*) s viacerými heterogénnymi dronmi. Model

umožňuje nákladnému vozidlu prepravovať flotilu heterogénnych dronov s viacerými návštevami pre kooperatívne doručovanie, kde drony sú schopné doručovať viacerým zákazníkom na jednej trase a ich let je obmedzený spotrebou energie a užitočným zaťažením. Na riešenie navrhli heuristickú metódu založenú na klastrovaní K-Means++ a Greedy vyhľadávaní najbližšieho suseda na vytvorenie realizovateľných riešení.



Obr. 6. a) Zákazníci rozdelení K-Means++, b) Konštrukcia trás vozidla,
c) Konštrukcia letov dronov, d) Komplettná trasa vozidla [7]

Poslednou verziou, ktorá nás zaujala, je úloha obchodného cestujúceho s asistenciou dronu so vzdialenými robotickými stanicami (*Drone-Assisted Traveling Salesman Problem with Robot Stations*) **TSP-D-RS** [12]. V depe máme kamión s dostatočnou kapacitou vybavený dronom (obrázok 7), mikro-sklad robotov (hviezda) a umiestnenia staníc, pričom každá z nich má rovnaký typ dopytu.



Obr. 7. Vľavo: bez robotov, vpravo: s robotmi [12]

Cieľom je obslúžiť všetky stanice tak, aby sa dosiahol minimálny čas doručenia alebo nákladov. Autori prezentujú exaktné riešenie pomocou Gurobi riešiteľa MILP modelom do 30 minút pre $n \leq 20$.

3. Modely

Najskôr sa budeme venovať modelu TSP, z ktorého vychádza náš model. Pre ľahšie vyjadrovanie budeme ďalej namiesto miest hovoriť o uzloch.

3.1. Gavish-Graves [5]

V tomto MILP modeli interpretujeme premenné nasledovne:

- $x_{ij} \in \{0, 1\}$, $\forall i, j \in N$ sú bivalentné premenné. Ak $x_{ij} = 1$, potom trasa TSP priamo spája uzol i s uzlom j ($i \rightarrow j$), ináč $x_{ij} = 0$.
- $f_{ij} \geq 0$, $\forall i, j \in N$ sú nezáporné tokové premenné.

$$\sum_{(i,j) \in N^2: i \neq j} c_{ij} \cdot x_{ij} \rightarrow \min, \quad (1)$$

$$\sum_{j \in N - \{i\}} x_{ij} = 1, \quad \forall i \in N, \quad (2)$$

$$\sum_{i \in N - \{j\}} x_{ij} = 1, \quad \forall j \in N, \quad (3)$$

$$\sum_{i \in N - \{k\}} f_{ik} = \sum_{j \in N - \{k\}} f_{kj} + 1, \quad \forall k \in N - \{0\}, \quad (4)$$

$$f_{ij} \leq n \cdot x_{ij}, \quad \forall (i, j) \in N^2, \quad (5)$$

$$f_{ij} \geq 0, \quad \forall (i, j) \in N^2, \quad (6)$$

$$x_{ij} \in \{0, 1\}, \quad \forall (i, j) \in N^2. \quad (7)$$

Cieľová funkcia (1) udáva celkovú prejdenu vzdialenosť. Podmienky (2) a (3) požadujú, aby cestujúci z každého uzla raz odišiel a raz doň prišiel. Tokové podmienky (4) zabezpečia, že po každej návšteve uzla (nie depa) ubudne jedna jednotka toku. Podmienky (5) koordinujú cestu a tok. Podmienky (6) a (7) sú obligatórne.

Za povšimnutie stojí, že podmienky (2) a (3) formulujú obmedzenia priradovacej úlohy v úplnom grafe bez slučiek s n uzlami, ktorá je riešiteľná v polynomiálnom čase $O(n^3)$. Podobne toková úloha o prípustnom toku formulovaná obmedzeniami (3) a obmedzením na tok $0 \leq f_{ij} \leq n$ je tiež riešiteľná aj grafovým algoritmom v polynomiálnom čase $O(n^2)$.

Ďalej budeme v našom modeli predpokladať, že je daný pomer $\rho \in (0, 1)$ medzi dobou trvania doby letu a doby jazdy vozidlom na jednotkovú vzdialenosť, t. j. $t_{ab}^d = \rho \cdot t_{ab}$ udáva dobu letu medzi uzlami a a b dopravnej siete, kde t_{ab} je doba jazdy vozidlom medzi uvažovanými uzlami.

3.2. Základný model TSP-DR

V tomto modeli sme si veľmi zjednodušili život tým, že za kritérium sme ne-tradične zvolili *maximálne rozpätie* medzi príchodmi vozidiel do uzlov. Navyiac na množine uzlov N je daný *rozklad* na množinu uzlov D obsluhovaných dronom a množinu uzlov V obsluhovaných vozidlom, t. j. $N = V \cup D$, $V \cap D = \emptyset$. Najskôr budeme predpokladať, že $|D| = |V - \{0\}|$. Bez straty všeobecnosti budeme značiť:

$$V = \{0, 1, 2, \dots, p\},$$

$$H = \{(i, j) : i \in V, j \in V, i \neq j\},$$

$$D = \{p+1, p+2, \dots, 2(p+1)+1\}, \quad P = \{(i, k, j) : (i, j) \in H, k \in D\}.$$

V modeli interpretujeme premenné takto:

- $x_{ikj} \in \{0, 1\}$, $\forall (i, j) \in H$, $\forall k \in D$ sú bivalentné premenné. Ak $x_{ikj} = 1$, potom je trasa dronu $i \rightarrow k \rightarrow j$, ináč $x_{ikj} = 0$.
- $y_{ij} \in \{0, 1\}$, $\forall (i, j) \in H$ sú bivalentné premenné. Ak $y_{ij} = 1$, potom trasa vozidla priamo spája uzol i s uzlom j , t. j. $i \rightarrow j$, ináč $x_{ij} = 0$.
- $f_{ij} \geq 0$, $\forall (i, j) \in H$ sú nezáporné tokové premenné vozidla.

Dostávame nasledujúcu úlohu MILP.

$$z \rightarrow \min \tag{8}$$

$$\sum_{k \in D} x_{ikj} = y_{ij}, \quad \forall (i, j) \in H, \tag{9}$$

$$\sum_{(i,j) \in H} x_{ikj} = 1, \quad \forall k \in D, \tag{10}$$

$$\sum_{j \in V - \{i\}} y_{ij} = 1, \quad \forall i \in V, \tag{11}$$

$$\sum_{i \in V - \{j\}} y_{ij} = 1, \quad \forall j \in V, \quad (12)$$

$$\sum_{i \in V} f_{ik} - \sum_{j \in V} f_{kj} = 1, \quad \forall k \in V - \{0\}, \quad (13)$$

$$\sum_{k \in D} (t_{ik}^d + t_{kj}^d) x_{ikj} - t_{ij} y_{ij} \leq z, \quad \forall (i, j) \in H, \quad (14)$$

$$\sum_{k \in D} (t_{ik}^d + t_{kj}^d) x_{ikj} - t_{ij} y_{ij} \geq -z, \quad \forall (i, j) \in H, \quad (15)$$

$$0 \leq f_{ij} \leq n y_{ij}, \quad \forall (i, j) \in H, \quad (16)$$

$$x_{ikj} \in \{0, 1\}, \quad \forall (i, k, j) \in P, \quad (17)$$

$$y_{ij} \in \{0, 1\}, \quad \forall (i, j) \in H, \quad (18)$$

$$z \geq 0. \quad (19)$$

Cieľová funkcia (8) udáva maximálne rozpätie medzi príchodmi dronu a vozidla, ktoré minimalizujeme. Podmienka (9) priradí každej hrane práve jeden dron. Podmienka (10) zase priradí dronu práve jednu vybranú hranu. Podmienky (11), (12) a (13) sú zhodné s podmienkami (2), (3) a (4) a zabezpečujú okružnú trasu (TSP) vozidla na uzloch V . Podmienky (14) a (15) definujú rozpätie. Podmienky (16) tokovo koordinujú vozidlo a dron. Podmienky (17), (18) a (19) sú obligatórne.

4. Experimenty a plány

Počítačové experimenty sme robili na notebooku HP (400Mz, 2-jadrový) pomocou riešiteľa Gurobi Optimizer version 11.0.3. Model Gavish-Graves vyriešil tsp225.tsp symetrickú benchmarkovú inštanciu TSPLIB [10] exaktne za 12,95 minút a inštancie pod 150 uzlov do 1 minúty.

Tabuľka 1. Výsledky experimentov

Inštancia	n	Max. rozpätie	TSP-DR
gr24	24	173	1,25 s
berlin52	52	451	42,51 s
gr120	70	274	2,14 min
ch130	100	240	10,27 min
gr120	120	227	29,64 min
ch150	50	127	35,83 min

Prvé pokusy na inštanciách z TSPLIB s modelom TSP-DR sú povzbudivé. V tabuľke (1) uvádzame výsledky experimentov. Pokúsime sa získať reálne inštancie od autorov vyššie uvedených „dronových“ článkov. Chceli by sme rozšíriť

náš model o kapacitné podmienky dopytu pri nakládke a vykládke, časové okná uzlov, doby obsluhy v uzloch, kombinácie fixovaných a voľných typov obsluhy uzlov, resp. návratom dronov do uzlov odletu a tiež flotily dronov.

V modeli sme požadovali nulový gap a $\rho = 0,5$. Vo vzdialenejšej budúcnosti by sme sa radi zamerali aj na okružné úlohy s dronmi (VRP-D), o ktorých je tiež veľa dostupných článkov.

Máme ešte jeden sen. Je známe, že v prípade riešenia NP-ťažkých optimalizačných úloh MILP modelmi sa takmer štvrtinu času doby výpočtu venuje hľadaniu prípustného riešenia. Ten sa dá podstatne skrátiť, ak model štartuje z heuristického riešenia. No nie vždy platí, že čím kvalitnejšie štartovacie riešenie, tým je kratšia doba výpočtu. Ale to pri týchto úlohách neprekvapuje. Avšak zatiaľ sme nenašli prácu, ktorá by sa pokúšala skrátiť dobu výpočtu pred jej ukončením. Až príliš často sa totiž stáva, že algoritmus našiel optimálne riešenie no „nevie o tom“, a veľmi dlho ho overuje. Nuž a my by sme neskromne radi s tým aspoň trochu pohli.

Podakovanie. Za podnet k spísaniu príspevku ďakujeme neúnnavnému kolegovi Pavlovi Střížovi.

Podakovanie. Príspevok vznikol s príspevom grantu KEGA 019ŽU-4/2023 „Inovatívne učenie matematiky s podporou Open Source“ podporeného Slovenskou kultúrno-edukačnou grantovou agentúrou.

Literatúra

- [1] AGATZ, N. – BOUMAN, P. – SCHMIDT, M.: *Dynamic Programming Approaches for the Traveling Salesman Problem with Drone*, Published Online: 6 Apr 2018, <https://doi.org/10.1287/trsc.2017.0791>.
- [2] APPLGATE, D.L. – BIXBY, R.E. – CHVÁTAL, V. – COOK, W.J.: *The Traveling Salesman Problem, A Computational Study*, 2006 by Princeton University Press, ISBN-13: 978-0-691-12993, ISBN-10: 0-691-12993-2.
- [3] COOK, W. J.: *In Pursuit of the Traveling Salesman*, Mathematics at the Limits of Computation, Princeton University Press, Princeton and Oxford, 1957, ISBN 978-0-691-15270-7, dostupné <https://z-library.sk/>.
- [4] DIAZ-RIOS, D. – SALAZAR-GONZÁLEZ, J. J.: *Mathematical formulations for consistent traveling salesman problems*, European Journal of Operational Research 313, 2024, 465–477 <https://doi.org/10.1016/j.ejor.2023.08.021>.
- [5] GAVISH, B. – GRAVES, S. B.: *The Traveling Salesman problem and Related problems*, July 1978. Massachusetts Institute of Technology, Operations Research Center, Working Paper, OR 078-78, <https://dspace.mit.edu/handle/1721.1/5363>.
- [6] HUMPHRIES, M.: *Astro-TSP: Traveling Salesman Problem Based Solutions for Scheduling Astronomical Based Observations*, California State Polytechnic University, Pomona, 2023, Graduate Project, <http://hdl.handle.net/20.500.12680/f7623m070>.
- [7] LUO, Y. – DENG, X. – Ke, Y. – WAN, S. – OIAN, Y.: *Collaborative Intelligent Delivery With One Truck and Multiple Heterogeneous Drones in COVID-19 Pandemic Environment*, in IEEE Transactions on Intelligent Transportation Systems, July 2024, Vol. 25, No. 7, pp. 7907–7920, doi:10.1109/TITS.2024.3350876.

- [8] MORANDIA, N. – LEUSA, R. – MATUSCHKEB, J. – YAMANA, H.: *The Traveling Salesman Problem with Drones: The Benefits of Retraversing the Arcs*, Published Online: 11 Jul 2023, Transportation Science, Vol. 57, No. 5, <https://doi.org/10.1287/trsc.2022.0230>.
- [9] PAMBO, C. – GROBLER, J.: *An Evolutionary Algorithm for the Vehicle Routing Problem with Drones with Interceptions*, September 2024, [cs.AI], 21, DOI:10.48550/arXiv.2409.14173.
- [10] Reinelt, G.: *TSPLIB—A Traveling Salesman Problem Library*, ORSA Journal on Computing, 3(4), 1991, 376–384. <http://comopt.ifl.uni-heidelberg.de/software/TSPLIB95/>
- [11] ROBERTI, R. – RUTHMAIR, M.: *Exact methods for the traveling salesman problem with drone*. Transportation Science, 2021, 55(2), 315–335 <https://doi.org/10.1287/TRSC.2020.1017>.
- [12] SCHERMER, D. – MOEINI, M. – WENDT, O.: *The Drone-Assisted Traveling Salesman Problem with Robot Stations*, 2020, In book: Proceedings of the 53rd Hawaii International Conference on System Sciences (HICSS), 1308–1317, DOI:10.24251/HICSS.2020.161.
- [13] TADIČ, S. – KOVAČ, M. – ČOKORILO, O.: *The Application of Drones in City Logistics Concepts*, Promet – Transport & Transportation, Accepted: 11 Jan 2021, <https://traffic.fpz.hr/index.php/PROMTT/article/view/3721>.
- [14] TANG, Z. – SHAW, P. – VAN HOEVE, W. J.: *A Study on the Traveling Salesman Problem with a Drone*, January 2019, Lecture Notes in Computer Science, in book: Integration of Constraint Programming, Artificial Intelligence, and Operations Research, 557–564, DOI: 10.1007/978-3-030-19212-9_37.

Kontaktné adresy

doc. RNDr. Štefan Peško, CSc., Ústecká 1/23, 010 08 Žilina, Slovensko,
E-mailová adresa: pesko@frcatel.fri.uniza.sk

RNDr. Zuzana Borčinová, PhD., Katedra matematických metód a operačnej analýzy, Fakulta riadenia a informatiky, Žilinská univerzita, Univerzitná 8215/1, 010 26 Žilina, Slovensko,
E-mailová adresa: zuzana.borcinova@fri.uniza.sk

VÝZKUM DOKUMENTOVÉ GRAMOTNOSTI: STAV A PERSPEKTIVY

JIŘÍ RYBIČKA (CZ), PAVEL HALUZA (CZ) A LUCIE PACÁKOVÁ (CZ)

Abstrakt. Dotazníkový průzkum v oblasti tvorby a zpracování dokumentů zahájený v roce 2022 pokračoval v tomto akademickém roce sběrem dalších dat a zapojením třetí instituce – Technické univerzity v Košicích. Současně však došlo po diskusi s dalšími perspektivními partnery a po zkušenostech z první fáze výzkumu k částečným změnám v konstrukci dotazníku. Článek prezentuje důvody a výsledky těchto změn, dále sumarizuje některá zpracovaná data z posledního roku a porovnává je s daty předchozích let. Na závěr představuje perspektivy možného pokračování výzkumu, zpracování dat a aplikaci do výukového procesu.

Klíčová slova. Dotazníkové šetření, typografické zásady, technika dokumentů, technika dotazníků, koncentrace dat.

DOCUMENT LITERACY RESEARCH: STATUS AND PERSPECTIVES

Abstract. The questionnaire survey in the field of document creation and processing, which began in 2022, continued in this academic year by collecting additional data and involving a third institution – the Technical University of Košice. At the same time, however, after discussions with other prospective partners and after experiences from the first phase of the research, there were partial changes in the construction of the questionnaire. The article presents the reasons and results of these changes, further summarizes some processed data from the last year and compares them with data from previous years. Finally, it presents perspectives for possible continuation of the research, data processing and application in the teaching process.

Keywords. Questionnaire survey, typographic rules, document technique, questionnaire technique, data concentration.

1. Úvod

Myšlenka zjišťování určitých specifických znalostí a dovedností studentů v oblasti tvorby a zpracování dokumentů na počítači vznikla již v roce 2022 a od té doby byl již realizován dotazníkový průzkum, jehož dílčí výsledky byly publikovány mimo jiné v článku [1]. Koncepce stanovená v začátku byla založena na zjišťování stavu tří okruhů – jazykových pravidel, typografických pravidel a technických dovedností.

Vycházeli jsme přitom z obsahu Rámcových vzdělávacích programů pro gymnázia v České republice [2, s. 70], kde je popsáno, že středoškolsky vzdělaná osoba umí „prezentovat výsledky své práce s využitím pokročilých funkcí aplikačního

softwaru“, přičemž v učivu je potřebné zvládnout „formy dokumentů a jejich strukturu, zásady grafické a typografické úpravy dokumentu, estetické zásady publikování“.

Dotazník navržený v této první fázi navazoval na jiný průzkum týkající se finanční gramotnosti [3], z něhož jsme převzali konstrukci dvou částí – sebereflexe a test. Projektovali jsme tak možnost zjišťování korelace mezi tím, co si respondenti v této oblasti o sobě představují, s tím, co ve skutečnosti umí.

Tvorba dokumentů je do určité míry popsána také platnou normou [4], ta tvořila také důležitý zdroj při sestavování otázek. Do testovací části jsme zahrnuli segment týkající se jazykových pravidel, segment typografických pravidel a segment technických dovedností.

Dalším úmyslem bylo opakovat výzkum za pokud možno stejných podmínek i v následujících letech, abychom mohli vyhodnocovat i časové řady.

Vymezení okruhu respondentů se týkalo nově nastupujících studentů do prvních semestrů bakalářských studijních programů na Provozně ekonomické fakultě brněnské Mendelovy univerzity a Fakultě stavební Vysokého učení technického v Brně. Tyto dvě fakulty byly zvoleny souhlasně s pracovními působišti autorů.

2. Změny v organizaci výzkumu

Po dvou prvních letech sběru a elementárním zpracování dat došlo k vyhodnocení dílčích výsledků. Původní předpoklad vyjádřitelný zjednodušeným tvrzením, že „respondenti mají špatné jazykové znalosti“, se nepotvrdil – i bez hluboké analýzy dosažené výsledky ukazují, že jak typické („tradiční“) jazykové jevy ručně psaného textu, tak poněkud překvapivě i jevy spojené s psaním na počítači jsou zvládnuty v přijatelné míře a mezi školami se hodnocení výrazně neliší.

V oblasti typografických pravidel a technických dovedností se ovšem předpoklad, že proklamace definované v Rámcovém vzdělávacím programu nejsou naplněny, spíše potvrzuje. Výsledky důkladnějšího zpracování těchto dat by měly napomoci identifikovat problematická místa a odvodit také možnosti jejich eliminace.

2.1. Změny v dotazníku

Potřeba zjišťovat znalosti jazykových pravidel se z uvedeného důvodu minimalizovala a současně s tím se rýsovala možnost dotazník velmi zjednodušit. Jazykové otázky byly vytvářeny pro českou a pro slovenskou variantu dotazníku odlišně, což automaticky vnášelo do těchto dvou jazykových mutací určitou nerovnost.

V roce 2024 se v projektu navíc objevili zástupci dalších univerzit a rozvinula se diskuse o obsahu dotazníku. Bylo zřejmé, že stávající podobu bude potřebné upravit, i když snahou bylo provést změny jen v co nejmenší míře, aby bylo možné i nadále využívat maximum již získaných dat.

Dosavadní podoba dotazníku byla vytvořena v češtině, pouze dvě otázky týkající se pravopisných pravidel měly alternativu ve slovenštině. Nyní s příchodem zástupců slovenských univerzit vznikla potřeba mít plnou slovenskou verzi. O tu se postarali zástupci Žilinské univerzity.

Kromě odstranění zmíněných dvou jazykových otázek byly provedeny další úpravy:

- Přibylly otázky v anamnestické části, které zpřesňují nebo doplňují původně sbírané informace: navštěvovaná škola, studovaný obor (dříve nebylo součástí dotazníku, protože podle identifikace respondenta bylo možno zpětně doplnit), nově pak kraj trvalého bydliště a velikost sídla trvalého bydliště.
- Některé otázky v sebereflexní části, jejichž možnosti vyjadřují míru splnění nějakého kritéria (například otázka týkající se míry zvládnutí programového vybavení nebo sebehodnocení znalosti pravidel) byly upraveny na pětistupňové Lickertovy škály.
- Zjišťování typografických a technických znalostí a dovedností zůstalo stejné.

2.2. Technologie dotazníku

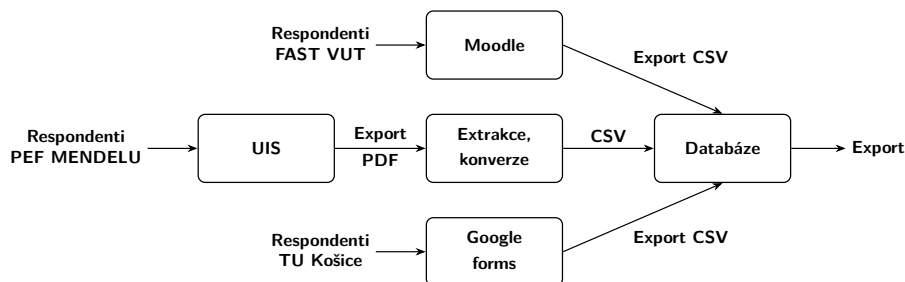
Pro sběr dat na MENDELU byl použit dotazník implementovaný ve studijním informačním systému UIS, na FAST VUT pak stejný dotazník implementovaný v systému Moodle. Hlavním důvodem, proč byly tyto systémy použity, byla možnost zpětně dohledat identitu respondenta. Pro samotný výzkum je tato informace nepotřebná, ale vyplnění dotazníku na obou těchto fakultách bylo odměňováno bonusovými body ve vybraných předmětech.

Obě tyto technické varianty však byly pro další univerzity neakceptovatelné. Systém UIS je navíc vnitřním systémem MENDELU a není vůbec k dispozici na většině dalších škol. Proto vznikla varianta dotazníku v prostředí Google Forms. Ta má v této chvíli pouze slovenskou variantu, ale do začátku dalšího akademického roku je v plánu vytvořit i variantu českou.

2.3. Koncentrace dat

Sběr a zpracování získaných dat se doposud prováděl odlišně na dvou pracovištích, což bylo dáno použitými systémy. Na schématu (obr. 1) je vidět, že data z UIS bylo dokonce potřeba složitě získávat z jediného možného výstupu ve formátu PDF, zatímco ze systému Moodle bylo možné vytvořit export celé skupiny testů ve formátu CSV. Podrobný popis těchto konverzí byl publikován ve zmíněném článku [1]. Schéma zároveň prezentuje velmi snadné doplnění dat z Google Forms, odkud lze získávat informaci ve formátu CSV.

Je velmi pravděpodobné, že i na MENDELU bude tento dosud problematický bod celého zpracování vyřešen přechodem na test v systému Moodle, neboť stále potřebujeme zachovat možnost identifikovat respondenty kvůli bonusové motivaci k vyplňování dotazníků. Nedořešeným místem zůstává hromadné nastavení



Obrázek 1. Koncentrace dotazníkových dat z různých zdrojů

studentských účtů pro tuto aktivitu, neboť neexistuje (a určitě existovat ani do budoucna nebude) datový propoj mezi UIS a Moodle.

2.4. Vhodný čas pro realizaci sběru dat

Z diskusí a ze zkušeností také vyplynulo, že optimálním okamžikem realizace výzkumu je některý z prvních týdnů zimního semestru. Potřebujeme respondenty, kteří po absolvování středních škol čerstvě nastupují ke studiu na univerzitě.

Podle našich zkušeností se během prvních týdnů stává, že studenti opouštějí studium (důvodů je několik), proto není vhodné toto testování posouvat za první třetinu semestru. Dotazník ve všech technických a jazykových variantách musí být připraven optimálně ještě před zahájením výukové části semestru, musí být zvolena vhodná organizace (například vyplnění dotazníku v rámci určitého předmětu) a technické zabezpečení (přístupy, účty v testovacím systému apod.).

3. Výsledky akademického roku 2024/25

V akademickém roce 2024/25 byly realizovány tři dotazníkové průzkumy – ke dvěma brněnským přibyl i průzkum zrealizovaný na Technické univerzitě v Košicích v začátku letního semestru, tedy v únoru 2025. Pro ilustraci a porovnání uvádíme některé údaje z dřívější doby.

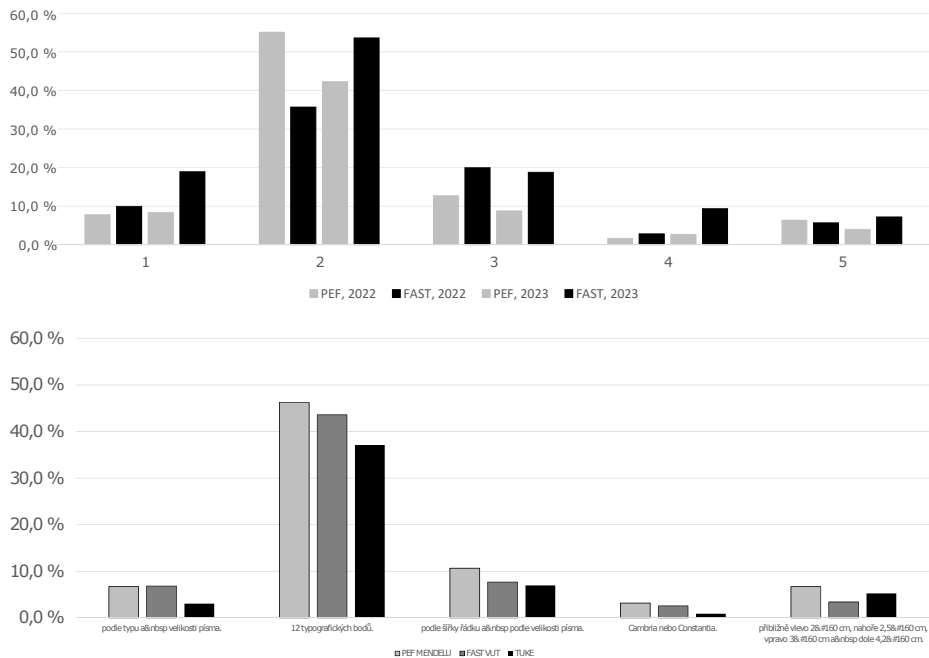
Během podzimu probíhala se zástupci univerzit poměrně rozsáhlá diskuse týkající se tvaru dotazníku, která se protáhla až téměř do konce října. Mezitím však byl aplikován na MENDELU tvar dotazníku neobsahující úplně všechny změny, na kterých pak vznikla všeobecná shoda. Nový tvar v cílové podobě byl aplikován na FAST a v únoru 2025 v Košicích.

Počty respondentů (tab. ??) – na brněnských pracovištích byly počty menší, protože tvar dotazníku se v té době teprve ustaloval a vlastní šetření bylo zahájeno podstatně později než v předchozích letech. Na FAST se nepodařilo z důvodu změny studijního plánu zorganizovat vyplnění dotazníku v jednom předmětu s velkým počtem studentů. Sloupec s počty respondentů označený rokem 2025 již není rozlišen na jazykové varianty, protože jazykové otázky byly odstraněny.

Tabulka 1. Počty respondentů v jednotlivých skupinách

Skupina	2023		2024		2025
	cz	sk	cz	sk	cz/sk
PEF	429	42	279	18	253
FAST	62	8	121	17	117
TUKE	—	—	—	—	229

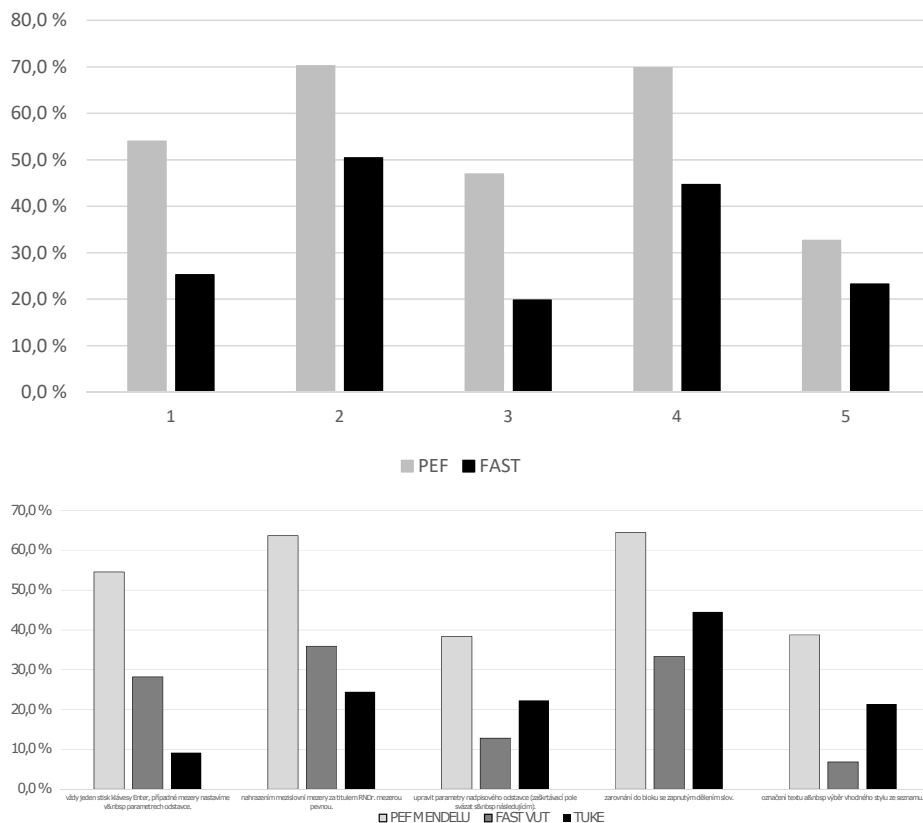
Odpovědi na otázky okruhu typografických pravidel pro srovnání vyjádříme stejným grafem jako v předchozích letech. Na obr. 2 jsou výsledky souhrnně za roky 2022 a 2023, v jeho dolní části pak výsledky na stejné otázky z akademického roku 2024/25.



Obrázek 2. Úspěšnosti otázek z okruhu typografických pravidel; souhrn za roky 2022, 2023, dole pak rok 2024/25

Pro srozumitelnost připomeneme, že otázka č. 1 se dotazuje na optimální řádkování (správná hodnota je dána typem a velikostí písma), otázka 2 se ptá, jakou jednotku má hodnota „12“ při volbě velikosti písma (typografické body), otázka 3 řeší optimální velikost odstavcové zarážky (závisí na šířce řádku a velikosti písma), čtvrtá otázka se týká volby vhodného písma pro sazbu hladkého textu na papír (Cambria nebo Constantia) a poslední otázka řeší správné nastavení okrajů (respektují optický střed stránky).

Okruh technických zásad vyjádříme podobně – grafem z předchozích let a stejným grafem roku 2024/25 (obr. 3).



Obrázek 3. Úspěšnosti otázek z okruhu použité technologie; souhrn za roky 2022, 2023, dole pak rok 2024/25

Zde první otázka řeší realizaci nového odstavce a jeho odstavcové zarážky (jeden Enter a nastavení ve vlastnostech odstavce), druhá otázka použití nezlomitelné (ve Wordu pevné) mezery (nikoliv zalomení řádku nebo posloupnosti mezer), třetí otázka se týká spojení nadpisu a následujícího odstavce (svázat s následujícím), čtvrtá otázka způsobu zarovnání při sazbě hladkého textu (zde bylo potřebné vědět, že nezbytností je zapnutí dělení slov) a pátá otázka směřovala k použití stylu (nikoliv vizuálního formátování klikáním na ikony).

3.1. Textové odpovědi

V nepovinné otázce na konci dotazníku mohli respondenti sdělit jakoukoliv informaci o dotazníku, jeho průběhu, obsahu nebo o oblasti zpracování textů na počítači.

Některé informace se opakují i z předchozích let, takže například připomínky k tvaru dotazníku už byly téměř všechny zapracovány. Zde se jedná pravděpodobně o respondenty, kteří jako jedni z posledních ještě dostali starší verzi dotazníku.

Třetina respondentů ponechala otázku bez odpovědi, případně jen uvedla, že nemají potřebu něco dodávat. U zbývajících odpovědí bylo nejčastěji uváděno následující (v závorce za námětem je uvedena stručná reakce):

- Přibližně desetina respondentů pochválila dotazník a jeho strukturu a poděkovala za možnost účastnit se výzkumu. (To nás vede k tomu, že respondentům, kteří uvedli na sebe elektronický kontakt, pošleme nějakou zpětnou vazbu.)
- Poměrně často respondenti díky dotazníku zjistili, že problematiku neovládají až tak dobře, jak se domnívali. (Tuto skutečnost bychom chtěli podpořit exaktním zpracováním získaných dat.)
- Respondenti se vesměs shodují v důležitosti znalostí a dovedností v dané oblasti a nutnosti posílení její výuky na středních školách, kde bývá často probírána jen okrajově nebo vůbec. (Tuto skutečnost bychom rovněž chtěli podpořit zpracováním získaných dat.)
- Někteří respondenti přiznávají, že při práci v Microsoft Office je často jejich cílem dosažení nějakého přijatelného výsledku, aniž by řešili skutečně správný postup. (Tím se dotýkájí dost závažného problému – absence znalosti pravidel a nedostatečná znalost technických principů vede k neefektivní, nebo dokonce k chybné práci, jenže zůstaneme-li u vzdělávání, většina učitelů to bere jako zcela okrajovou záležitost a řadu nedostatků vůbec nevidí nebo vědomě opomíjí, považuje je za nedůležité.)

Z námětů na vylepšení dotazníku pro budoucí roky lze zmínit především tyto (opět s reakcí v závorce):

- Doplnění možnosti „nevím“ u každé otázky jako prevence tipování a možného zkreslení výsledků. (To bylo diskutováno, jde o otázky, kde je cílem zvolit optimální postup. Tam se varianta „nevím“ neuplatní, protože výslednou informací je, do jaké míry je respondent schopen pracovat optimálně.)
- Uvedení počtu správných odpovědí u úloh k technologickým zásadám. (Patrně jde o jednu otázku na správný zápis matematického výrazu, kde byly přijatelné dva způsoby. Mimochodem z tohoto důvodu jsme výsledky této otázky neuvedli v přehledu výsledků z technického okruhu dotazníku.)
- Doplnění dotazů zjišťujících či zohledňujících specifické poruchy učení. (Taková informace v dotazníku dosud není a pravděpodobně bude vhodné se zabývat jejím začleněním.)
- Technické problémy u otázek obsahujících chybné jevy – některým respondentům nebylo jasné, zda mají jako správnou označit tu, která obsahovala

chybu, nebo tu, která správná opravdu byla. (Tady jde o jazykové otázky, které byly mezitím vyřazeny.)

- Upřesnění, zda za správnou odpověď je považováno jakékoliv řešení vedoucí k výsledku, nebo jen to „nejsprávnější“. (Dobrý námět, toto upřesnění by mohlo v zadání některých otázek skutečně být.)
- Problém s názvoslovím karet v anglické verzi Microsoft Office vs. použitá (česká) terminologie v dotazníku. (Dobrý námět, to by mohlo být do textu jednotlivých variant odpovědí doplněno.)

Byly zaznamenány také požadavky na vytvoření speciálního předmětu, který by se problematice podrobně věnoval.

Takový předmět na PEF MENDELU existuje, je však pouze volitelný a v posledních letech si jej zapisuje pouze kolem 10 studentů ročně. V prvním semestru se zde vyučuje předmět zabývající se tvorbou odborných prací, jehož součástí je (v mírném rozsahu jedné přednášky a jednoho cvičení, tj. celkem 4 hodin) i řešení formální stránky dokumentu. Podobná situace je i na FAST – speciální volitelný předmět určený především studentům řešícím závěrečné práce si prakticky nikdo nevolí, v úvodním semestru je v přehledovém předmětu (ani ne pro všechny obory) věnována úpravě dokumentů malá část o celkovém rozsahu 6 hodin. Domníváme se, že už ze středních škol jsou studenti vesměs přesvědčeni, že jejich úroveň znalostí a dovedností v oblasti zpracování textů je natolik dobrá, že žádnou takovou průpravu již nepotřebují.

Daleko větší potenciál by pravděpodobně měl při realizaci jiný podnět – videokurz šířený na platformě YouTube. Dobře udělaná sada videí by mohla oslovit větší okruh studentů, je to pro ně v poslední době preferovaná forma získávání informací. Stále ovšem platí, že také musí existovat nenulová snaha dostatečně fundovaně kontrolovat formální kvalitu dokumentů ze strany učitelů, hodnotitelů, oponentů závěrečných prací atd. Tam ovšem vidíme rovněž velké rezervy.

4. Perspektivy

Bylo by prospěšné výzkum rozšířit na více univerzit. Tohoto cíle se nepodařilo dosáhnout již minulý rok, který byl věnován hlavně dotažení změn v dotazníku, i když Technická univerzita v Košicích již svůj první průzkum realizovala. Podle současného stavu by výzkum mohl probíhat na těchto institucích:

- Mendelova univerzita v Brně (Provozně ekonomická fakulta),
- Vysoké učení technické v Brně (Fakulta stavební),
- Žilinská univerzita v Žilině,
- Technická univerzita v Košicích,
- Ekonomická univerzita v Bratislavě,
- Vysoká škola technická a ekonomická v Českých Budějovicích.

4.1. Získávání dat

Protože chceme získávat informace od studentů těsně po absolvování střední školy, provádění průzkumu by bylo optimální v úvodních týdnech po přijetí na vysokou školu. Hodí se k tomu například předměty spojené s úvodem do studia, kde lze samotné provedení testu i organizačně podpořit.

Data by bylo vhodné koncentrovat do jednoho místa, centrální databáze, odkud by bylo možné získávat potřebné výstupy. Tato data jsou již anonymizována, ale je zachována kompletní informace o vyplněném dotazníku každou osobou.

4.2. Analýzy

S rostoucím množstvím získaných dat roste i potenciál pro analýzy, kde se nabízí několik směrů:

- Popisné charakteristiky (např. úspěšnosti zjišťovacích otázek).
- Korelace (sebereflexní a zjišťovací část), závislosti výsledků na různých faktorech (obor studia, národnost, pohlaví, místo narození...).
- Komparace (mezi institucemi, mezi státy).
- Časové analýzy, trendy.

Domníváme se, že výsledky těchto analýz mohou ukázat zajímavé skutečnosti a zpřesnit celkový pohled na předmětnou oblast, mohou také ukázat možné směry dalšího úsilí ve výuce, což by měl být nejužitečnější výsledek celého výzkumu.

4.3. Zpětná vazba

Jednou ze součástí dotazníku je rovněž kontakt na respondenty, kteří projeví zájem o výsledky průzkumu. Pravděpodobně by bylo vhodné již nyní skupinu respondentů z let 2022 a 2023 obeslat informací s dílčími výsledky a sdělit jim možná i správné odpovědi ze zjišťovací části dotazníku, aby se mohli sami přesvědčit o stavu svého poznání v tomto okruhu.

5. Závěr

Efektivní tvorba a zpracování dokumentů pomocí počítače patří mezi dovednosti, jimiž by měl být vybaven bezesporu každý středoškolsky vzdělaný člověk. Přesto se může zdát, že jejich zvládnutí stojí poněkud na okraji zájmu – ano, je nesporné, že vzdělaný člověk v určitém oboru musí být vybaven dlouhou řadou dalších znalostí a dovedností, které se mohou zdát důležitější. Výzkum provedený v dostatečném rozsahu a dostatečné časové řadě může ukázat, jaká je skutečnost, na čemž lze pak dále stavět vzdělávací strategii. Zvýšení úrovně dovedností a znalostí v oblasti zpracování dokumentů bychom mohli považovat za legitimní cíl, který zúročí veškeré úsilí věnované tomuto výzkumu.

Reference

- [1] RYBIČKA, J., PACÁKOVÁ, L.: *Jak jsou středoškoláci vybaveni pro tvorbu dokumentů?*, Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach, Zborník príspevkov medzinárodnej konferencie OSSConf 2024, Žilina, 2.–4. júla 2024, str. 57–68, ISBN 978-80-554-2105-6.
- [2] MŠMT: *Rámcový vzdělávací program pro gymnázia, RVP G*, 2021, online, dostupné na https://www.edu.cz/wp-content/uploads/2021/09/001.RVP.GYM.-vyznacene_zmeny.pdf.
- [3] KOZUBÍKOVÁ, Z.: *Financial Literacy in Selected Groups of Students*, Proceedings of the Conference: Knowledge for Market Use 2016: Our Interconnected and Divided World, Olomouc, Societas Scientiarum Olomucensis II (2016), pp. 222–230.
- [4] ÚNMZ: *ČSN 01 6910 – Úprava dokumentů zpracovaných textovými procesory*, 2014, Praha.

Kontaktní adresy

doc. Ing. Jiří Rybička, Dr., Ústav informatiky, Provozně ekonomická fakulta, Mendelova univerzita v Brně, Česká republika,

E-mailová adresa: rybicka@mendelu.cz, <https://user.mendelu.cz/rybicka>

Ing. Pavel Haluza, Ph.D., Ústav informatiky, Provozně ekonomická fakulta, Mendelova univerzita v Brně, Česká republika,

E-mailová adresa: pavel.haluza@mendelu.cz

Lucie Pacáková, Ústav informatiky, Provozně ekonomická fakulta, Mendelova univerzita v Brně, Česká republika,

E-mailová adresa: pacakovalu@email.cz

O JEDNÉ MAPĚ NA ZAKÁZKU ČÁST 1: TEORÉM 4 BAREV PRAKTICKY

PAVEL STRÍŽ (CZ)

Abstrakt. Tento článek je úvodem do sazby map pomocí R a $\text{\TeX}$. Užíváme data z OpenStreetMap. V této první části z několika článků se zaměřujeme na užití teorému čtyř barev na obarvení regionů. Jádrem článku bylo užití jednoho z řešitelů (SAT, MIP, CP, SMT) a jejich stručné srovnání. Použili jsme Python, který generuje .pi soubor pro program Picat. Ten má výhodu v tom, že na jednom řádku se lze mezi řešiteli přepínat.

Klíčová slova. Optimalizace, R, Python, Picat, OpenStreetMap.

ABOUT PREPARATION OF ONE MAP PART 1: THE FOUR COLOUR THEOREM IN PRACTICE

Abstract. The article is an introduction to typesetting maps using R and $\text{\TeX}$. We are using data from OpenStreetMap. In this first part of several articles, we focused on applying The Four Colour Theorem to fill the regions. The main approach was to use optimization techniques (SAT, MIP, CP, SMT) and briefly compare them. We used Python to generate .pi code for the Picat software. It has feature that we can switch between mentioned solvers editing one line of code.

Keywords. Optimization, R, Python, Picat, OpenStreetMap.

1. Rešerše: The Four Colour Theorem

Během mých výletů ve světě wordcloudů jsem otevřel starší problém, a to vysázet mapu ČR a SR s erby měst. V pozadí jde o to, že místo bodů na spirále mohou využít jediný bod (GPS souřadnici), tedy by to mělo být výrazně rychle i při tisícovkách měst. Aleš Kozubík mi poradil, ať zkusím něco (kraje, okresy, města) v pozadí vybarvit přes teorém 4 barev (zkracováno na 4CT). Tento teorém je mezi informatiky a matematiky slavný, protože byl první, který byl dokázán za asistence počítačů. V sedmdesátých letech 20. století kontroverzní téma, nyní již záležitost běžně přijímána.

Ony významné publikace jsou [3,4]. Čtenářům bych doporučil článek Georgese Gonthiera [2], který poutavě píše o historii teorému, ale také o moderní asistenci, speciálně přes program Coq, nyní nazýván Rocq, <https://en.wikipedia.org/wiki/Rocq>. Z novějších knih zmíním *A Guide to Graph Colouring* od R. M. R. Lewise [1]. Poměrně podrobně se zabývá heuristickými algoritmy.

Populární formou o důkazu mluví na kanálu Numberphile, <https://www.youtube.com/watch?v=NgbK43jB4rQ>.

Jak v mé přítomnosti vyslovíte teorii grafů, ihned se mi vybaví nástroj NetworkX. Pokud však nahlédneme do <https://networkx.org/documentation/stable/reference/algorithms/coloring.html>, zjistíme, že se minimálního počtu barev nedočkáme. To jsem zbystřil. Na YouTube chlapík, <https://www.youtube.com/watch?v=AyrrZ4PCyws>, šel také do rekurze.

Optimální strategii nenalezneme ani v nástroji gcol v Pythonu. <https://rhydlewis.eu/gcol/> To je vedlejší produkt ke knize [1]. To už mi bylo divné.

Pak už to přece jenom začalo být zajímavější. Prvně jsem to klučinu viděl řešit přes knihovnu simpleai, <https://pypi.org/project/simpleai/>, potažmo pydot, flask a Flask-SQLAlchemy, viz <https://www.youtube.com/watch?v=hV6qTfFSwCk>. Dokumentace hovoří o CSP algoritmech, a to už je člověk doma.

Přes Python jsem zahlédl Knuthův Algoritmus X, <https://stackoverflow.com/questions/42863543/applying-the-4-color-theorem-to-list-of-neighbor-polygons-stocked-in-a-graph-arr>. To jsem nezkoušel, ale upozornil jsem. Přes mé oblíbené SAT řešitele jsem zahlédl tuto implementaci, <https://mathoverflow.net/questions/419808/is-there-any-fast-implementation-of-four-color-theorem-in-python>. Kdekoliv zahlédnu slavný program Gurobi, tak si představím svobodný program Symphony či Scip. To už jsem si říkal, to bude chtít něco zkusit vlastními silami. Pro tu srandu.

Pro seriální badatele doporučuji <https://cran.r-project.org/web/views/Optimization.html>, já jsem zde bádání pozastavil.

Aktuální a fascinující ukázky představuje <https://four-color-theorem.com>, resp. <https://github.com/stefanutti/maps-coloring-python>. Ještě zmíním jeden tip: u map mě fascinovaly experimenty v R na <https://fronkons.tin.com/>. Doporučuji cca 15 HTML stránek si prolistovat. Jsou to svěží grafické ukázky.

2. Vstup na pole R

Analýza dat se v těchto dnech nejčastěji bije na poli mezi Pythonem a R. Pro běžného smrtelníka je dobré umět obojí, byť pracovat s R je nezvyklé. Pro mne je stále nezvyk všude vidět `c()`, což je zápis dat do vektoru nebo seznamu (`combine`), `rbind` či `cbind`, což je rozhození dat do řádku či sloupce. Je to i po letech nezvyk.

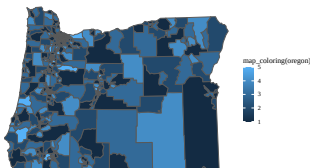
V každém případě jsem narazil na blog z roku 2015, <https://www.r-blogger.com/2015/11/solving-the-map-coloring-problem-in-r/>, a že to zkusím. Po chvíli hledání jsem našel autorův účet, viz <https://github.com/hunzikp/MapColoring>. Během instalace se však dozvíte, že CTAN nadále nepodporuje knihovny `rgeos` a `rgdal`, že je potřeba se obrátit na knihovnu `sf`. Knihovna vás pohltí, nemluvě o animaci hexaloga na <https://r-spatial.github.io/sf/>. Jak by řekl klasik: „Tohle je jiný svět!“

A to už začalo být zajímavé. Knihovna je podrobně popsána, a když nahlédneme na <https://r-spatial.github.io/sf/articles/sf1.html>, už to tam body a polygony jen lítá. Bližší zkoumání mě dovezlo až ke knihovně **ggredist**, která už obarvení mapy umí. Ačkoliv si lze navolit snahu o minimum barev, autoři píší, že umí jen *greedy* funkci. Pokud se pohybujete na poli optimalizace, tak na tohle slovo jste alergičtí, protože bývá hned vedle *naïve* řešení. Berte to, prosím, s rezervou. Na vysvětlení principů však stav ideální.

V Linuxu jsem si pracovně odinstaloval knihovnu **libmariadb-dev** a doinstaloval **libudunits2-dev** a **libgdal-dev**. Pokud tak neučiníte, R vás bude informovat o neúspěšné instalaci. Potřebujete totiž vedle knihovny **ggredist** i **units**, **sf** a **ggplot2**.

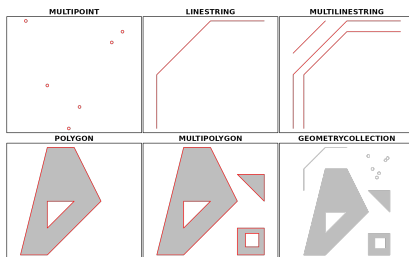
Pokud jsem si vzal základní ukázkou a vztáhl na celý stát Oregon, dostáváme:

```
library(ggplot2)
library(ggredist)
data(oregon)
map_coloring(oregon)
ggplot(oregon, aes(fill = map_coloring(oregon))) + geom_sf() + theme_map()
```



3. Hlouběji do R

Dobře, základní ukázkou je v pořádku, jak to však funguje? Struktura je taková, že z bodů vznikají úsečky, z těch polygony, či seznamy bodů, úseček či polygonů. Nejvyšší úroveň je kolekce geometrických útvarů. U seznamu polygonů je důležité, že na prvním místě je vnější křivka, další položky v seznamu jsou výřezy. Je to trochu nezvyk, ale u mapových podkladů to tak je.



Zkusíme ukázkou. Připravíme si šest čtverců a necháme si je obarvit. Z bodů vzniká seznam, ze seznamu polygon, z polygonů kolekce pro knihovnu **sf**. Následně si necháme proměnnou vypsát, vykreslit přes **plot** a **ggplot**. V posledním bloku se dopočítají sousedské vztahy, vypíše a vykreslí se výsledek.

```
p1<-rbind(c(0,0), c(1,0), c(1,1), c(0,1), c(0,0))
p2<-rbind(c(1,0), c(2,0), c(2,1), c(1,1), c(1,0))
p3<-rbind(c(2,0), c(3,0), c(3,1), c(2,1), c(2,0))
```

```

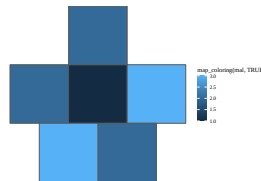
p4<-rbind(c(1,1), c(2,1), c(2,2), c(1,2), c(1,1))
p5<-rbind(c(1.5,0), c(2.5,0), c(2.5,-1), c(1.5,-1), c(1.5,0))
p6<-rbind(c(0.5,0), c(1.5,0), c(1.5,-1), c(0.5,-1), c(0.5,0))

mpol1<-st_polygon(list(p1)); mpol2<-st_polygon(list(p2))
mpol3<-st_polygon(list(p3)); mpol4<-st_polygon(list(p4))
mpol5<-st_polygon(list(p5)); mpol6<-st_polygon(list(p6))

mal1<-st_sfc(list(mpol1,mpol2,mpol3,mpol4,mpol5,mpol6))
mal1
plot(mal1)
ggplot(mal1) + geom_sf() + theme_map()

mal<-st_sf(geometry=mal1)
map_coloring(mal)
ggplot(mal, aes(fill = map_coloring(mal,TRUE))) + geom_sf() + theme_map()

```



Abych ještě zprůhlednil tvorbu výřezů do polygonu, zde je jednoduchá ukázka. Do čtverce naděláme dvě čtvercové díry, resp. dva nezávislé polygony, které se mají také obarvit. Tato úvaha se nám bude dál hodit: Brno versus Brno-okolí, Košice versus Košice-okolí.

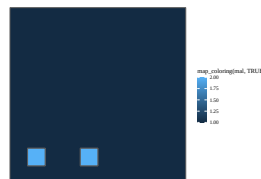
```

p10<-rbind(c(0,0), c(10,0), c(10,10), c(0,10), c(0,0))
p11<-rbind(c(1,1), c(2,1), c(2,2), c(1,2), c(1,1))
p12<-rbind(c(4,1), c(5,1), c(5,2), c(4,2), c(4,1))

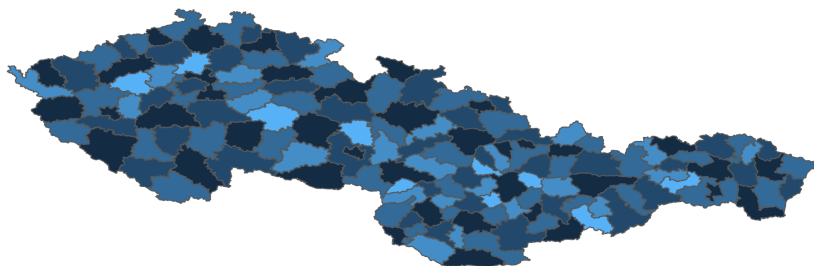
mpol10<-st_multipolygon(list(list(p10,p11,p12)))
mpol11<-st_multipolygon(list(list(p11)))
mpol12<-st_multipolygon(list(list(p12)))

mal2<-st_sfc(list(mpol10,mpol11,mpol12))
mal<-st_sf(geometry=mal2)
map_coloring(mal)
ggplot(mal, aes(fill = map_coloring(mal,TRUE))) + geom_sf() + theme_map()

```



Vzal jsem data za ČR a SR a R kód jsem si nechal vygenerovat. Skrz úsporu místa zde 22,6 MB zdrojový kód nevkládám, ale mohu říci, že mapa se vybarvila pěti barvami. My bychom si přáli ony slavné čtyři barvy. Teorém říká, že to jde, ale neříká, jak toho docílit.



4. Práce se surovými daty

Jsem surovomil, čím surovější data, tím milejší. Data jsem si stáhl z <https://osm-boundaries.com/>. Prvně jsem užil Overpass, <https://overpass-api.de/>, ale vedle hranic byly ve výsledku další údaje, neměl jsem dost sil na extrakci potřebného.

Pak mě napadlo, že by se hodilo znát sousedské vztahy. Stručně vysvětlím. Vzal jsem si všechny body hranic okresů a u nich jsem si uložil informaci, který okres je obsahuje. Pokud bod nemá souseda, je v takovém seznamu jeden okres. Ve většině případů jsou tam dva okresy, občas víc, jak se v jednom bodu setkává víc okresů. Z těchto informací, když je smrskneme na neopakující, lze získat sousedské vztahy. Tip: u Prahy bylo potřeba si dát pozor, že jsou obvody, ale i celá Praha. Tedy body na hranicích Prahy měly 3 údaje: Prahu-východ či Prahu-západ, Prahu celkovou, a pak některou z obvodů Praha 1–10. Další zvláštnost byla Brno, tam se nesmělo zapomenout, že se skládá z vnějšího a vnitřního polygonu.

Další úprava dat spočívala v tom, že byl požadavek mít společnou hranici pro Bratislavu I–V a Košice I–IV. Využil jsem předchozího bádání a vnější body o jednom údaji tvořily základ nové hranice. Plus se užily spojovací body mezi okresy. Byly to krajní body na těch úsecích. Tato finta mě potěšila, že jsem na to přišel. Jak by se to mělo řešit mimo surová data, ať už v R, Pythonu či jinými nástroji: nedokáží vám prozradit.

Pokud jste „rekurzíholik“ (s úsměvem: srovnejte alkohol a alkoholik, nebo excesivní rekurzik či snad dokonce rekurzák) jako má maličkost, tím se vždy práce začíná. Zde je ukázka kódu, který sice teoreticky funguje, ale časově nám začne kolabovat kolem 60 okresů.

```
def maldfs(position):
    global visited, colors
    if position==166:
        print(colors)
        exit()
    col=["red","green","blue","yellow"]
    for okres in okresy[position]["sousedci"]:
        if colors[okres] in col:
            col.remove(colors[okres])
    for c in col:
        visited[position]=True; colors[position]=c
        maldfs(position+1)
```

```

    visited[position]=False; colors[position]="none"

visited={}; colors={}
for k in range(165+1):
    visited[k]=False
    colors[k]="none"
maldfs(0)

```

Struktura vstupních dat, přesněji vyjádření sousedských vztahů, vypadala velmi jednoduše:

```

okresy={
# [...]
114: {'poradi':114, 'name': 'okres Nitra', 'sousedi': [90,96,110,119,124,126,145]},
115: {'poradi':115, 'name': 'okres Velký Krtíš', 'sousedi': [100,132,133,134,145]},
# [...]
}

```

Měl jsem radost, ale na 166 okresů to byl nedostatečný přístup.

5. Optimalizace na pomoc

Během jednoho večera jsem narazil na tuto stránku https://www.reddit.com/r/math/comments/7uhtci/four_colour_theorem_is_there_an_existing/. Zhlédl jsem tyto dva řádky od uživatele you-get-an-upvote na redditu a bylo mi hned jasné, jak to zpytlíkovat. Někdy stačí málo. Jen dopisuji čárky do druhého řádku, ty tam chyběly a souseda jsem upravil na k z j .

```

red(i)+green(i)+blue(i)+yellow(i) = 1
red(i)+red(k)<=1, green(i)+green(k)<=1, blue(i)+blue(k)<=1, yellow(i)+yellow(k)<=1

```

Mějme SAT model s proměnnými x_{ij} , kdy za i máme zvažované okresy, za j barvu 1–4. Dále chtějme tuto proměnnou binárního typu. Sečteme-li proměnné za okres i , musí být součet roven 1, tedy barva ze čtyř je použita jen jedna a právě jedna. To říká ten první řádek. Druhý řádek nám říká, že libovolní dva sousedi i a k mohou užít konkrétní barvu maximálně jednou. Záměrně jsem si model nepřevodl na minimalizační či maximalizační, chtěl jsem, řekněme, barvy rovnoměrně či náhodně jako překvapení.

Zde je zdrojový kód pro Python, který generuje soubor pro program Picat, <https://picat-lang.org/>, který se v závěru spustí. Struktura vstupních dat byla již představena.

```

import os
# Načtení vstupních dat.
kam=open("forpicat.pi","w")
kam.write("import sat.\n")
kam.write("main =>\n")
kam.write("Sol=[")

# Definování proměnných.
data=[]
for o in okresy:
    for b in [1,2,3,4]:
        data.append("Xo"+str(o)+"b"+str(b))

```



```

kam.write(",".join(data))
kam.write("],\n")
kam.write("Sol :: 0..1,\n")
kam.write("\n")

# Seznam okresů, které se neužijí, Praha, Bratislava a~Košice.
forbidden=[0,1,2,3,4,5,6,7,8,9, 112,141,144,147, 116,117,118]
# Jen jedna barva na okres.
for o in okresy:
    data=[]
    for b in [1,2,3,4]:
        data.append("Xo"+str(o)+"b"+str(b))
    if o not in forbidden:
        kam.write(",".join(data)+" #= 1,\n")
kam.write("\n")

# Pomocné proměnné, které se v závěru vypíší.
vypis=[]
for o in okresy:
    data=[]
    for b in [1,2,3,4]:
        data.append(str(b)+"*Xo"+str(o)+"b"+str(b))
    kam.write("Xo"+str(o)+" #= "+"+".join(data)+" ,\n")
    vypis.append("Xo"+str(o))
kam.write("\n")

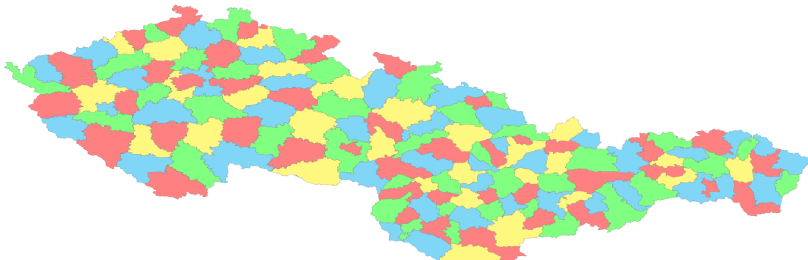
# Bonusy, u neužitých obvodů/okresů dej barvu "none".
for kkk in forbidden: # Praha, Bratislava a~Košice.
    kam.write("Xo"+str(kkk)+" #= 0,\n")
kam.write("\n")

# Jádro, sousedské vztahy.
for o in okresy:
    for soused in okresy[o]["sousedi"]:
        for barva in [1,2,3,4]:
            kam.write("Xo"+str(o)+"b"+str(barva)+" + Xo"+str(soused)+"b"
                      +str(barva)+" #<=1,\n")
kam.write("\n")

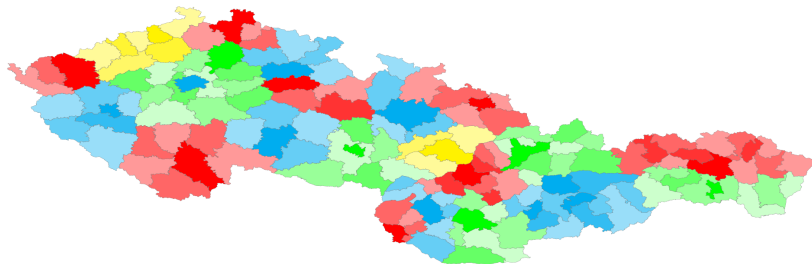
kam.write("Vypis=["+"".join(vypis)+"] ,\n")
kam.write("solve(Vypis),\n")
kam.write("writeln(Vypis).\n")
kam.close()

os.system("picat forpicat.pi")

```



Při nejnižším možném počtu barev jsem vybral první obarvení a užil. Jako bonus se krajské město (vyjma Středočeského kraje) nastavilo nejsytější barvou. Zde je závěrečná ukázka. Kdyby se šlo na úroveň ohraničení měst, dokáží si představit užití šrafování v dané barvě a intenzitě.



6. Úvaha místo Závěru

Zaujala mě tato webová stránka, <https://carto.com/blog/sql-graph-coloring>, byť pravidelně bych takto v SQL nechtěl programovat.

Bylo by zajímavé zjistit si velikost jednotlivých oblastí a nějak to zahrnout do SAT $\rightarrow$ ILP modelu. Například aby čtyři barvy byly co nejvíc rovnoměrně či naopak co nejvíc rozházené. Nikdy jsem to nezkusil a těžko se představují grafické výstupy, které by z toho mohly vzniknout. Třeba nápad pro někoho ze čtenářů!

Reference

- [1] R. M. R. Lewis: *A Guide to Graph Colouring: Algorithms and Applications*, 2nd edition, Springer Cham, 2021. <https://rhydlewis.eu/gcol/>
- [2] Georges Gonthier: *A computer-checked proof of the Four Color Theorem*. <https://inria.hal.science/hal-04034866/document>
- [3] K. Appel, W. Haken: *Every Planar Map is Four Colorable, Part I: Discharging*, Illinois Journal of Mathematics 21 (1977), 429–90. <https://projecteuclid.org/journals/illinois-journal-of-mathematics/volume-21/issue-3/Every-planar-map-is-four-colorable-Part-I-Discharging/10.1215/ijm/1256049011.full>
- [4] K. Appel, W. Haken: *Every Planar Map is Four Colorable, Part II: Reducibility*, Illinois Journal of Mathematics 21 (1977), 491–567. <https://projecteuclid.org/journals/illinois-journal-of-mathematics/volume-21/issue-3/Every-planar-map-is-four-colorable-Part-II-Reducibility/10.1215/ijm/1256049012.full>

Kontaktní adresa

Ing. Pavel Stríž, Ph.D., Nakladatelství Martin Stríž, U Škol 940, Bučovice, okres Vyškov, 685 01, Česká republika,
E-mailová adresa: pavel@striz.cz

MÍCHÁNÍ RASTROVÉ A VEKTOROVÉ GRAFIKY V TYPOGRAFII

PAVEL STRÍŽ (CZ)

Abstrakt. Článek představuje některé styčné body kombinování vektorové a rastrové grafiky na úrovni přípravy mraků slov.

Klíčová slova. Mrak slov, $\text{\TeX}$, Python, FontForge, Inkscape, GIMP, kvadrantové stromy.

COMBINING RASTER AND VECTOR GRAPHICS IN TYPOGRAPHY

Abstract. The paper introduces basic concepts in typography of combining vector and raster graphics at a preparation of wordclouds level.

Keywords. Wordclouds, Tag clouds, $\text{\TeX}$, Python, FontForge, Inkscape, GIMP, quadtrees.

1. O problému

Kolem roku 2007 jsem sázel popisky na CD a DVD a chtěl jsem vysázet drobnost: Rok bude vysázen tak, aby se cifry dotýkaly. Měla to být má tečka za konferenční řadou $\text{\TeX}$ perience.

2007! → 2007!

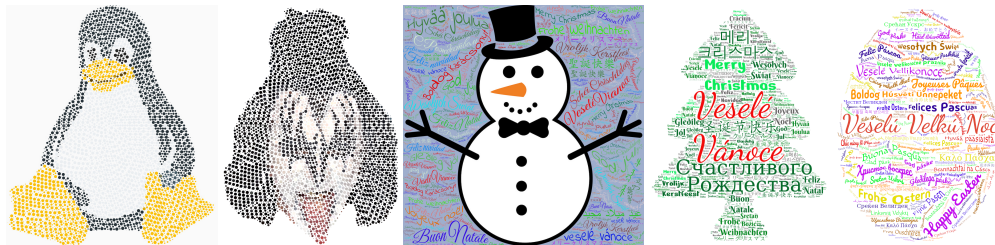
V $\text{\TeX}$ u získáme výsledek ručně za pár minut po několika iteracích, třeba takto:

```
{\bfseries\huge2\kern-2.4pt0\kern-1.9pt0\kern-2.4pt7\kern-2.4pt!}
```

Pokud se nad tím zamyslíme hlouběji a chceme-li takovou sazbu zautomatizovat, dostaneme se na úroveň křivek glyfů. K těm se dá dostat například přes METAPOST či program FontForge. Zjistit dotyk křivek je ale výpočetně náročná úloha. A to mluvíme o dvou křivkách, žádných složitých obrazcích. Začal jsem s tím experimentovat. Bohužel na zjištění toho, jestli je bod uvnitř či vně jisté křivky jsem použil nevhodný postup a mé cvičné pětihvězdičky jsem nebyl schopen vysázet na obálku tehdy vznikající knihy.

Matematik Honza Šustek mi za jeden večer poslal řešení v METAPOSTu na poskládání glyfů. Tam však byla otázka, jak přistupovat k libovolným obrázkům s blíže nespecifikovaným počtem křivek. Problém jsem odložil v duchu, že je nad mé síly. Po letech, v roce 2015, mi to stále vrtalo hlavou a vznesl jsem dotaz na $\text{\TeX}$.SE, <https://tex.stackexchange.com/questions/180510/how-to-get-intersection-points-of-two-glyphs>, ale neposunul jsem se. Zde je několik vzorků ze světa wordcloudů, to byla jistá inspirace, jak by vzorky mohly

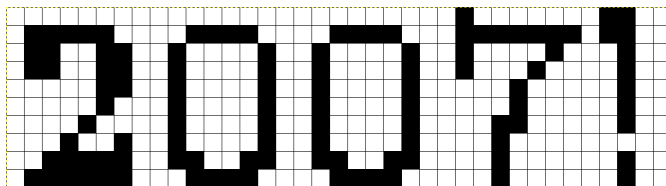
vypadat. Ukázky jsou od Aleše Kozubíka za pomoci <https://www.wordclouds.com/> a <https://wordart.com/>.



2. D3.js

S příchodem tohoto nástroje jsem si všiml častého užití rekurzivních datových struktur quadtree (kvadrantový strom) a octree (oktálový strom), viz například Poisson Disc Sampling, <https://www.jasondavies.com/poisson-disc/>. Kdybych to měl amatérsky přiblížit, tak je to rozšíření metody půlení intervalu v 1D pro 2D a 3D. Použití je masivní hlavně u vícerozměrných dat (typu mraků dat v geoinformatice či herním průmyslu) a rastrových obrázků (arkádové hry). Tehdy začal vznikat nápad, že by se, teoreticky řečeno, dal vektorový výstup sazby vzít, převést do rastrového obrázku, nějak s tím pracovat, a výsledky použít znovu do sazby, ale pro původní vektorové podklady. Teoreticky by mělo docházet k chybě maximálně jednoho pixelu, s možností si rastrový obrázek zjemnit, kdyby bylo potřeba. Pozn. o bojích s písmem více Sebastian La-gue, <https://www.youtube.com/watch?v=S083KQuuZvg>, o nových písmech ho-voří Erik Demaine, https://www.youtube.com/watch?v=zZcJq0T_FP8.

Zde je školní ukázka. Pokud provedeme posun znaků o 2 (první nula), 4 (druhá nula), 6 (sedmička) a 7 pixelů (vykřičník) dostaneme se na jejich dotyky. V každém případě vidíme, že počet pixelů neodpovídá posunům v první ukázce (jednotky pt). Pokud si velikost původního obrázku přepočteme na pixely, můžeme provést posun i v obrázku vektorovém. Pokud by chyba pixelu byla viditelná, zjemníme si rastrový obrázek, tím dojde k přesnějšímu výsledku.

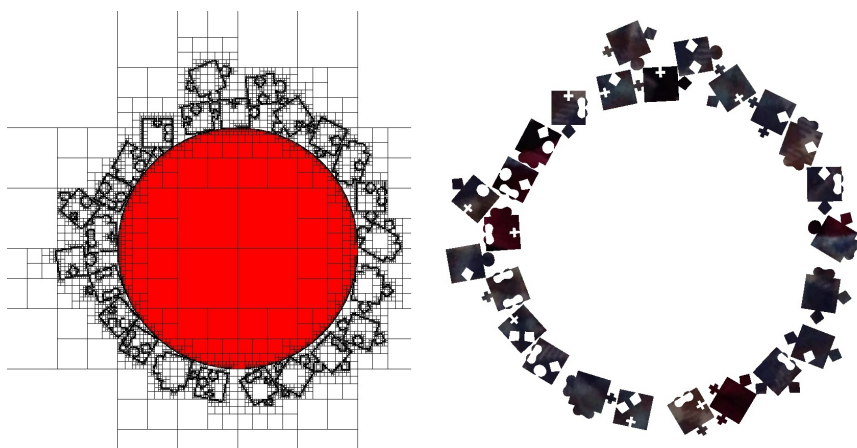


Jistý nápad to sice byl, ale rekurzivní datové struktury nebyly můj denní chleba, tak jsem se do testů nepouštěl. Jistou práci s XML, SVG či JSON formáty jsem měl sice za sebou, ale tohle bylo ještě o kus dále. Bez nich to však prakticky u rastrových obrázků nejde skrz nutný výpočetní čas.

3. Puzzle

V mezičase jsem řešil kombinatorické úlohy. Jedna z nich byla spočítat, kolik jedinečných dílků (i s ohledem na rotaci) lze získat u puzzle. Matematik by na to šel přes Burnsideův teorém, programátor přes zjištění kolik dílků je jedinečných/rotačně invariantních z celkového počtu. Obě cesty mi zafungovaly a za odměnu jsem si chtěl puzzle dílky nechat vykreslit. Lze to udělat náhodně, ale dílky by šly přes sebe a na kontrolu by to nebylo vhodné. Tak jsem použil vysázení dílků do mřížky. Tím jsem se pochlubil Alešovi Kozubíkovi do Žiliny. Odpověď byla jasná: není to náhodné, není to hezké. Tak jsem si říkal, že bych mohl zkusit to vysázet do spirály, přesně tak, jak je to v pozadí u wordcloudů. O těch jsem věděl, že právě užívají kvadrantové stromy.

Pustil jsem se do prvních pokusů. Málem jsem spadl ze židle, když jsem viděl svůj první zdařilý pokus: puzzle dílky naskládané jakoby mimo kruh (dále se to nazývá jako mustr). Zde je onen obrázek. V levé části je vidět vizualizace konečného stromu, v pravé části přechod zpět do vektorového světa sazby. A nejen to, kdyby bylo potřeba, tak lze puzzle dílky sázet jako rastrové obrázky.

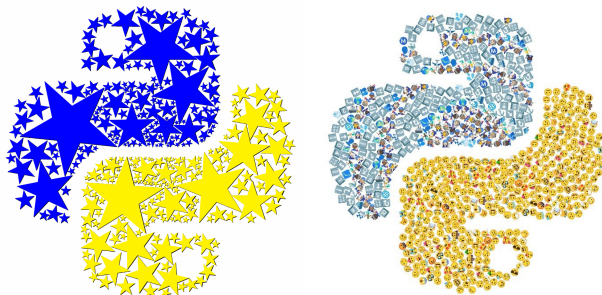


Abych zahladil efekt umisťování na spirálu, tak jsem si dílky náhodně zmenšoval. Dokonce lze ze spirály přejít na jiný typ (po řádcích, po sloupcích, více menších spirál, či dle libovůle). Ale za základ testů jsem vzal spirálu.

Kvadrantový strom se mi buduje tak, že zkoumám, jestli jsou v kvadrantu pixely bílé či černé, přiřadím kvadrantu nulu či jedničku. Pokud je však kvadrant směsí bílých a černých, rozčtvrtí se oblast a zkoumá rekurzivně hlouběji. Tímto způsobem si lze libovolný obrázek rozebrat na soustavu černých obdélníků, u kterých je jednoduché hledat průsečíky s jinými obdélníky. To je výpočetní jádro. Postupně z fronty obrázků zjišťujeme, jestli je lze či nelze umístit na spirálu. Upraví se mustr a poté se ve frontě přejde na další obrázek.

4. Ukázky z vlastní zahrádky

Když umíme obrázky takto skládat do mustru. Už to není o ničem jiném než o přípravě mustru a vhodné přípravě podkladů. Naprogramoval jsem si fičury, že si mohu definovat oblasti barev a také oblasti, kde se mohou objevovat jen chtěné symboly. Toho jsem využil například u loga Pythonu, kdy jsem upřesnil barvy (obarvené hvězdičky, ukázka vlevo) či typ (vybrané žluté a modré emoji, ukázka vpravo). Přidal jsem přepínač, zda-li může či nemůže podklad do jiné oblasti.



Abych se dokázal zpátky vrátit do původní sazby používám $\text{\TeX}$ ový příkaz `\typeout`. Pak nemusím do konečné sazby užívat průběžné rastrové (např. PNG) či vektorové obrázky (např. PDF). Jedná se o řádovou úsporu velikosti souboru. Do článku se mi mnoho ukázek nevleze, ale příchozí účastníci na OSSConf si budou moci v Žilině prohlédnout výstavu. Některé vzorky budou vlastní a některé inspirativní wordcloudy budou od jiných autorů.

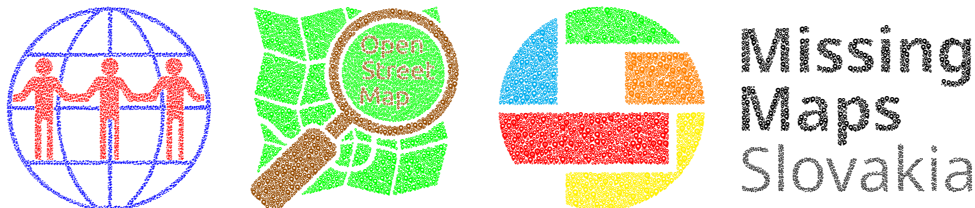
5. První zákazníci

Ačkoliv se mi zatím nepodařilo napojování kvadrantových stromů, dělám to zatím úpravou rastrového obrázku a novým generováním stromu, je program relativně pomalý. I tak jsem dal organizátorům OSSConf vědět, že je to již použitelné. Mezi první „zákazníky“ lze počítat Aleše Kozubíka a Mila Ofúkaného.

Pro Aleše vzniká logo konference, obrázky na jednotlivé sekce a experimenty s grafikou hudební skupiny KISS.



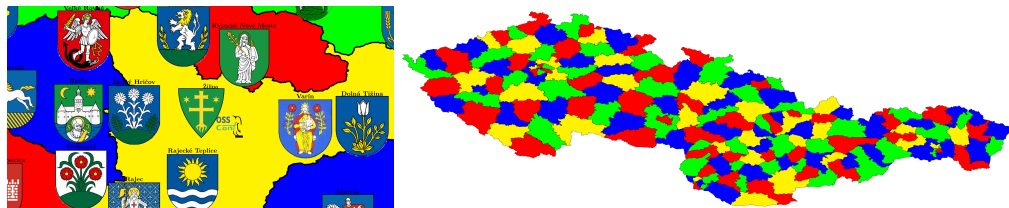
Milo Ofúkaný má obrovský záběr a řadu nápadů. Zkusili jsme doladit loga pro jeho sekci (Open GIS & Open Data).



6. Mapa ČR a SR a teorém 4 barev

Zesnulý T_EXpert Karel Horák se těšil na mou knihu erbů s popisky. Před mnoha lety se mi podařilo získat data za Českou republiku (systém REKOS), ale nikdy jsem neměl dost dat za Slovensko, tak jsem se do toho nepustil. Stále jsem erby měl v hlavě, a říkal jsem si, že zkusím alespoň mapu. Za pomoci dat Michala Páleníka, <https://www.oma.sk>, mapových podkladů z OpenStreetMap a údajů z Wikidat jsem začal mapu skládat.

Aleš Kozubík nadhodil nápad na problém 4 barev a jestli jej nějak nezpracovat. Připomínky posílal Rudolf Blaško, matematický model nezávisle testoval Štefan Peško. Já se naučil model ILP generovat pomocí Pythonu jako .pi soubor pro program Picat, <https://picat-lang.org/>. Zde je ukázka z pracovní verze z okolí Žiliny a jak vypadá vykreslení okresů, aby sousedi neměli stejnou barvu.



7. Rekurzivně

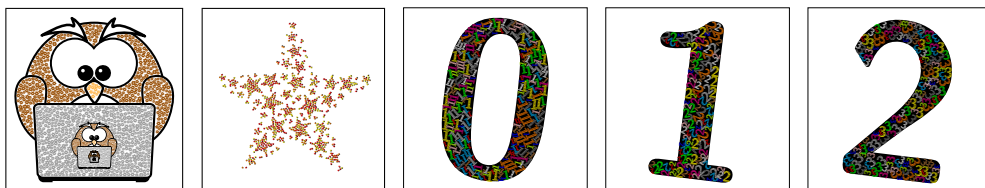
Vedle matematického modelování se snažím řešit úlohy rekurzí. Ne vždy je to technicky a časově možné, ale držím se tímto v programátorské formě. Proto jsem i u wordcloudů zkoumal možnost využití.

První možnost je jednoduchá a intuitivní. Vezmeme si hotový obrázek, v ukázce tučňák s notebookem, a v sazební postprodukci si vnořování připravíme jako vrstvení obrázků s jejich postupným zmenšováním.

Druhá možnost, která mě napadla, byla, že vezmeme nově vzniklý černobílý mustr a dále jej použijeme. Nemám to dostatečně zautomatizované, zde je ukázka hvězdy z hvězdiček z hvězdiček. Varianta to není ideální, protože si musíme hlídat velikost rastrového mustru. Čím se jde do větší hloubky, tím ztrácíme přesnost.

A na úrovni jednotek pixelů už nemá smysl pracovat. Proto jsem přemýšlel, jak na to jinak.

Třetí ukázka je zautomatizovaná verze, kdy si připravujeme podklady jakoby odzadu a postupně z nich sazbu vystavíme. V této ukázce je nula vyplněna jedničkami, ty jedničky jsou vyplněny dvojkami atd. Každá barva reprezentuje jeden typ, celkem bylo cca tucet barev. V ukázce je vždy první typ. Kompletní sazbu všech symbolů na jednu stranu nelze zrealizovat, protože PDF by dosáhlo neuvěřitelné velikosti. Můžeme však vnořování naznačit. Pro mne to byla fascinující úloha, u které jsem nevěřil, že s ní pohnu.



8. Několik tipů na nástroje

Ze starších nástrojů mohu zmínit METAPOST a pstoeedit. Bohužel METAPOST má omezení na práci s velkými čísly a je potřeba škálovat. Také pravidlo eofill je potřeba si ohnout nebo užít balíček `luamplib`, neb to Hans Hagen vyřešil.

Nástroj pstoeedit umí vzít PostScriptový soubor a vygenerovat zdrojový kód pro METAPOST. Dokonce umí SVG, autor psal, že některé části ještě veřejně nezveřejnil, ale že to plánuje. Bohužel se mi stalo, že občas konverze nedopadla dobře. Vypadá to tedy, že zpracovávat přímo SVG se jeví jako budoucnost. Můj pracovní cíl totiž je, že bych rád objekty vyřezával nikoliv používal barvu bílou či pozadí obrázku.

GIMP asi netřeba představovat. Programy ImageMagick a GraphicsMagick pomohly s následujícím: nechat levý a pravý dolní roh zaplnit průhledností. Užitečný je i přepínač `trim`, který promaže ochrannou zónu obrázku:

```
read w h <<(gm identify -format "%w %h" vstup.png)
let w=$w-1; let h=$h-1
gm convert vstup.png -matte -fill transparent temp.png
convert temp.png -fuzz 40% -fill none -draw "matte $w,$h floodfill" \
  -draw "matte 0,$h floodfill" -trim vysledek.png
```

První řádek zjistí velikost vstupního obrázku. Druhý řádek poníží zjištěnou velikost o jeden pixel, kvůli číslování pixelů od nuly. Třetí řádek zajistí, že obrázek bude mít průhledné pozadí. Poslední řádek využije spodních rohů a pokusí se zajistit průhlednost kolem erbu.

Program Inkscape také asi taky netřeba blíže představovat. Umí však i práci z příkazového řádku. První řádek vezme vstupní SVG a vygeneruje PDF. Co jsem u příprav zjistil je, že umí i zpracovat kroky, jako kdybych je kličal myší.

Na druhém řádku je ukázka, že se SVG prvně zmenší na velikost obrázku a pak vygeneruje PNG.

```
inkscape --export-filename=out.pdf in.svg
inkscape --actions=page-fit-to-selection --export-filename=out.png in.svg
```

FontForge je program na práci s písmy. Má své rozhraní, soubory s příponou .pe. Zde je ukázka, kterou lze spustit přes `fontforge -script malskript.pe`. Otevře se soubor s písmy, upraví se znak A a výsledek se uloží do nového souboru.

```
Open("MalPokus.otf");
Select("A");
ExpandStroke(10);
Simplify();
Save("MalPokusOut.otf");
```

Autor připravil i rozhraní pro Python. Zde je ukázka `svg-na-znak.py`, kdy lze do písma na určitou pozici uložit SVG obrázek.

```
import fontforge
maloutput=fontforge.font()
glyph=maloutput.createMappedChar("A")
glyph.importOutlines("owl_notebook.svg")
maloutput.fontname="MalPokus"
maloutput.save("MalPokus.otf")
```

Tímto způsobem jsem si připravil masku cifer z písma <https://www.wfonts.com/font/the-kiss-font>. V T_EXu jsem si znaky navrstvil. Je použit formát LuaL_AT_EX. O upravené písmo si případně napište, zatím nikde není.

```
\documentclass{standalone}
\usepackage{xcolor}
\usepackage{fontspec}
\setmainfont[Path=,Extension=.otf]{TheKISSFont}
\begin{document}
\makebox[0pt][l]{\color{red}\char"80\char"00BA}
\makebox[0pt][l]{\color{green}\char"80\char"03A9}
\makebox[0pt][l]{\color{blue}\char"80\char"00E6}
\makebox[0pt][l]{\color{red}\char"80\char"00F8}
\makebox[0pt][l]{\color{green}\char"80\char"00BF}
\makebox[0pt][l]{\color{blue}\char"80\char"00A1}
\makebox[0pt][l]{\color{red}\char"80\char"00AC}
\makebox[0pt][l]{\color{green}\char"80\char"221A}
\makebox[0pt][l]{\color{blue}\char"80\char"0192}
\makebox[0pt][l]{\color{red}\char"80\char"2248}
\end{document}
```

Takto vypadal výsledek. Zaujalo mě, že cifry 6 a 9 jsou rozdílné, to byl dobrý nápad. U této ukázky je vidět, že by nám obrys černých linek nestačil.



Barevná písma jsou stále nová a zkoumaná oblast. V T_EXovém světě jsem to poprvé zahlédl u balíčků `chessboard` a `xskak`. V posledních letech Sam Carterová vyvíjí zvířátka a postavy, technicky je to přes TikZ, tedy velikost je o něco větší

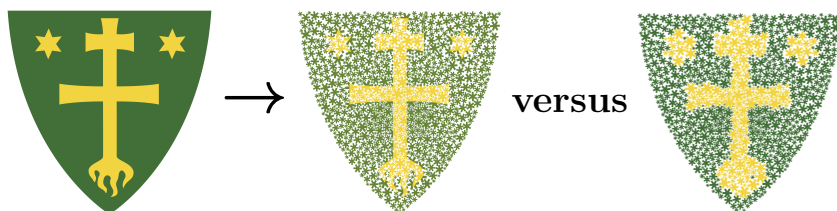
než kdyby to bylo přes znaky písma. Zaujatý čtenář nechť prozkoumá balíčky `tikzlings`, `tikzmarmots` a `tikzducks`.

9. Budoucnost a plány

V době psaní tohoto článku jsem stále ve fázi bádání a experimentování. Vize však je, že by mohl vzniknout nástroj s několika fičurami a měl by skončit zdokumentovaný na ctan.org. Původní nápad byl, že průvodce dokumentací bude bobr Erl a jeho proslavené stavby ze dřeva. Postupem času mě napadlo, že natolik míchám rastrový a vektorový svět, že se z toho stává blázonec. Tak jsem se přiklonil ke kočkopsovi (angl. catdog), tedy k vnitřnímu boji, resp. spolupráci, těchto dvou světů.

Mezi otevřené a řešené partie patří:

- Vzít znak a rozpůlit jej. Lze to řešit pomocí Voronoi diagramu. Výsledky povzbudivé nejsou. Vezmeme-li znak A, tak užití dvou barev je zvláštní, spíš se to tváří na tři barvy. Pokud se to podaří, bude to atypická záležitost.
- Podobně mám otevřený problém přípravy písma, tzv. ementálu. Tedy díry v písmu budou skutečné díry, nikoliv překreslení bílou barvou či barvou pozadí obrázku. Jistá vize by byla, ale chce to čas.
- Podařilo se mi půlit znaky za pomoci rastrového obrázku, viz ukázka. Chtělo by to však přejít na automatizované vyřezávání vektorové. Například podle barvy křivek.



V každém případě se mohu přiznat, že tohle byl můj letitý otevřený projekt, ať dělám na libovolné pasáži, učím se něco nového. Jestli se skutečně podaří jej ještě zrychlit a dostat do podoby použitelného nástroje pro veřejnost, to se ještě uvidí, byla by to však třešnička na dortu.

Kontaktní adresa

Ing. Pavel Stríž, Ph.D., Nakladatelství Martin Stríž, U Škol 940, Bučovice, okres Vyškov, 685 01, Česká republika,
E-mailová adresa: pavel@striz.cz

OPTIMALIZUJEME S PROGRAMEM CONCORDE

PAVEL STRÍŽ (CZ)

Abstrakt. Článek představuje základní užití programu Concorde. Jako samostatně spustitelný soubor, v programu R přes knihovnu `tsp` a přes NEOS server (v R a Pythonu). Na kresbu je užit příkaz `plot` a `ggplot`. NEOS server užívá GnuPlot. Velmi stručně jsou zmíněny alternativy TSP a možné práce do budoucna.

Klíčová slova. Concorde, optimalizace, R, Linux.

OPTIMIZATION WITH THE CONCORDE TSP SOLVER

Abstract. The article introduces basic use of the Concorde TSP Solver, as a standalone executable, in R with the help of the `tsp` library and via the NEOS server (R and Python interface). Drawing is done with `plot` and `ggplot` commands. The NEOS server is using GnuPlot. It also briefly introduces some TSP alternatives and future plans.

Keywords. Concorde, optimization, R, Linux.

1. Pár přejivých slov místo Úvodu

V roce 2009 (16 let nazpět v době psaní článku) mě tehdejší kolega Libor Sarga poprosil, jestli bych mu nepomohl se získáním vektorových obrázků u programu Concorde, <https://www.math.uwaterloo.ca/tsp/concorde>, pro připravovanou kapitolu v knize. Je to program řešící NP-úplně těžký problém obchodního cestujícího (Traveling Salesman Problem, TSP). Učil jsem operační výzkum, tak mi to přišlo jako zajímavý sazební úkol. Tehdy jsem nepracoval s Linuxem přímo, jen se CygWinem, tak jsem si na práci z příkazového řádku či hlubší pokusy netroufl. Objevil jsem však možnost uložit si výsledky do textových souborů v grafické verzi programu pod Microsoft Windows. Vznikla primitivní verze v Bashi.

Tehdy už byla Lua v $\text{\TeX}$ ovém světě docela zaběhlá, tak jsem zkusil tu. Vznikla tato Q+A na $\text{\TeX}$.SE, <https://tex.stackexchange.com/questions/235372/how-to-get-vector-graphics-from-concorde-program>. To se psal rok 2015 (10 let nazpět).

Shodou okolností se jeden z autorů, profesor Vašek Chvátal, vrátil z Kanady do Prahy a udělal jsem s ním rozhovor, viz tento sborník. Profesor se omlouvá, že na OSSConf nedorazí, ale ví o nás. Napadlo mě ty mé $\text{\TeX}$ ové pokusy kolem Concorde oprášit, možná až do podoby $\text{\TeX}$ ového balíčku. Tak jsem to začal zkoumat s novými programátorskými zkušenostmi.

2. Instalace

Vynechám instalační část pro Microsoft Windows, to jsem si kdysi zkusil a fungovalo to. Vrhnu se přímo do pro mne něčeho nového. Nahození v Linuxu bylo překvapivě rychlé. Ve složce jsem provedl tyto kroky. Plus lze nastavit složku do PATH, uživatelé Linuxu ví.

```
wget -c https://www.math.uwaterloo.ca/tsp/concorde/downloads/
      codes/linux24/concorde.gz
gzip -d concorde.gz
chmod +x concorde
./concorde
```

Podobně lze nahodit program Linkern, který za optimalizaci jde na věci jinak.

```
wget -c https://www.math.uwaterloo.ca/tsp/concorde/downloads/
      codes/linux24/linkern.gz
gzip -d linkern.gz
chmod +x linkern
./linkern
```

Vše se zdálo fungční.

Vyzkoušel jsem si i instalaci ze zdrojových kódů, to není nikdy na škodu, protože zásahy do zdrojových kódů člověk potřebuje častěji než by si to připouštěl. V nové složce jsem začal:

```
wget -c https://www.math.uwaterloo.ca/tsp/concorde/downloads/
      codes/src/co031219.tgz
tar xvf co031219.tgz
cd concorde
```

Plus jsem stáhl soubory pro QSOpt, jak radí instalační manuál.

```
mkdir -p QS; cd QS
wget -c https://www.math.uwaterloo.ca/~bico/qsopt/downloads/
      codes/ubuntu/qsopt{,.a,.h}
cd ..
```

Následovaly poslední dva příkazy:

```
./configure --with-qsopt=${PWD}/QS --with-cplex=${PWD}/LP
make
```

3. Základní užití

A hned jsem se pustil do testů. Manuál radí si zkusit 100 bodů s nastavením random seed (počáteční hodnota generátoru pseudonáhodných čísel).

```
cd TSP
./concorde -s 99 -k 100
./concorde -s 99 -k 100 -o test100a.sol
./concorde -s 99 -k 100 -f -o test100b.sol
```

První řádek vygeneruje 100 bodů a najde okružní cestu. Druhý řádek uloží výsledky do souboru, ale jen trasu. Poslední řádek to samé, ale trasa je vypsána nejen jako uzly, ale i s hranami. Soubory jsou textové, dá se s nimi dále pracovat. To bylo rychlé, že?

4. TSPLIB95

Abychom nepracovali se simulovanými daty, existuje krásná webovka s testovacími daty.

```
cd ..; mkdir -p TSPLIB95; cd TSPLIB95
wget -c http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/
    tsp/ALL_tsp.tar.gz
tar xvf ALL_tsp.tar.gz
for soubor in `find -iname \*.gz`; do echo $soubor; gzip -d $soubor; done
cd ../TSP
```

Nu, a můžeme si rovnou vyzkoušet vyřešit jeden z problémů.

```
./concorde ../TSPLIB95/a280.tsp
```

Výhoda těchto dat je, že lze nahlédnout na optimální cestu. To jsou soubory s příponou `.opt.tour`. A jsme doma!

Přes Python. Uživatelům vřele doporučuji <https://tsplib95.readthedocs.io>. Nezávislé testy jsou nad rámec tohoto příspěvku.

5. R

Zkoumal jsem, jestli si lze Concorde spustit v R.

Pokud nemáme, nainstalujeme si R. Podrobné návody lze nalézt na <https://cran.r-project.org/>. Spustíme si R a doinstalujeme knihovnu TSP. Detaily viz <https://cran.r-project.org/web/packages/TSP/vignettes/TSP.pdf>.

```
$ R
# install.packages("TSP")
```

Knihovnu si načteme, nastavíme cestu ke spustitelnému souboru a můžeme zkusit funkčnost knihovny. Asi nejrychlejší cesta, jak si zjistit cestu k adresáři je `echo ${PWD}`.

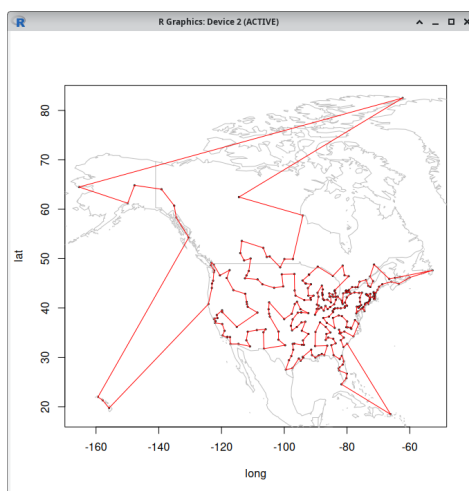
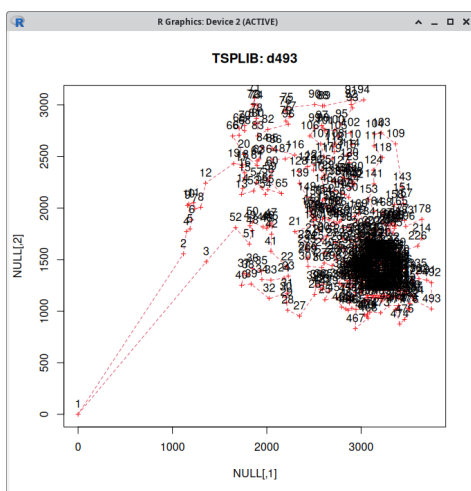
```
library(TSP)
concorde_path("...") # Nutné si upravit.
data("USCA312")
solve_TSP(USCA312, method="concorde", control=list(clo="-v"))
```

Knihovna TSP jednu ukázkou z TSPLIB95 obsahuje, můžeme rovnou zkoušet. Obrázek vlevo na další straně.

```
(drill<-read_TSPLIB(system.file("examples/d493.tsp",package="TSP")))
(tour<-solve_TSP(drill,method="concorde")) # Bylo nn,two_opt=TRUE.
plot(drill, tour, cex=.6, col = "red", pch = 3, main = "TSPLIB: d493")
```

Zkusíme si ještě jednu ukázkou z manuálu jako vzorek práce se skutečnou mapou. Obrázek vpravo.

```
# install.packages("maps")
library(maps)
data("USCA312")
(tour<-solve_TSP(USCA312, method="concorde"))
data("USCA312_GPS")
head(USCA312_GPS)
plot((USCA312_GPS[, c("long", "lat")]), cex = .3)
map("world", col = "gray", add = TRUE)
polygon(USCA312_GPS[, c("long", "lat")][tour,], border = "red")
```



Zájemcům o mapy přes R lze na další bádání minimálně doporučit <https://ggplot2-book.org/maps.html>. Více o tom zvaná přednáška Michala Páleníka.

6. R: ggplot2

Po bližším zkoumání jsem zjistil, že TSP sice používá Concorde, ale nějakým způsobem zasahuje do .tsp souboru před vyřešením, byť mi funkce `read_TSPLIB` a `write_TSPLIB` generovaly identický soubor. Jedná se konkrétně o to, že se snaží eliminovat desetinná místa, ale to jen před funkcí `solve_TSPLIB`.

Docházelo ke změně z 3 1.35760e+03 1.47900e+03 na 3 1e+03 1e+03. Nedokázal jsem to obejít. Zahlásil jsem to autorovi knihovny (verze 1.2-4). Opravil to 27. května 2025, <https://github.com/mhahsler/TSP/issues/21>, vznikla verze 1.2-5. Svět jsem sice nespasil, ale každý zlepšovák potěší!

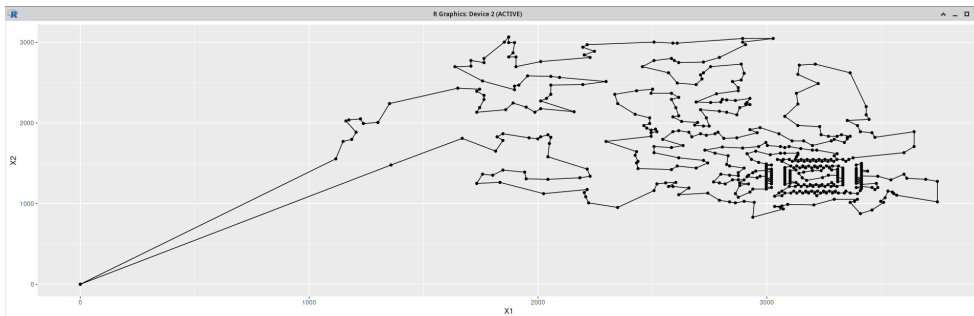
Zde je naznačení cesty, jak výsledky vykreslit přes knihovnu `ggplot2`.

```
# install.packages(c("dplyr", "data.table", "ggplot2"))
library(dplyr, data.table, ggplot2)
```

```

maldrill<-data.frame(drill); head(maldrill)
maltour<-data.frame(tour); head(maltour)
maldrill$poradi<-cbind(c(maltour$tour))
A<-maldrill[c(maldrill$poradi),]
B<-as.data.table(A)
C<-rbind(B,B[1])
ggplot(data=C,aes(x=X1,y=X2))+geom_point()+geom_path()

```



Poznámka 1. Finta s tabulátorem funguje i v R. Začneme psát příkaz a dáme Tab, jsou vidět dostupné možnosti.

Poznámka 2. Člověk to bral jako samozřejmost, předponu „gg“. Je to odkaz na knihu *The Grammar of Graphics*, <https://link.springer.com/book/10.1007/0-387-28695-0>.

Poznámka 3. Ony plusy mezi funkcemi není sčítání, ale vrstvení grafu. Jinými slovy přidávání podkladů do grafu.

7. Server NEOS

Poměrně nedávnou záležitostí je možnost využít komerční i nekomerční řešitele přes tento server. Dává to možnost si řadu věcí vyzkoušet, není třeba si programy instalovat či kupovat. Omezení jsou zřejmá: výpočet musí skončit do určité doby (je to závislé od programu, počítejte v řádu hodin), musíte zadání v nějakém formátu na server dostat, pak si stáhnout zpět atd. Pro výuku ideální záležitost.

Mezi řešiteli se objevil i program Concorde.

Mé první pokusy se starší knihovnou `rneos` nepřinesly žádané výsledky. Problém byl, že email je nově povinný parametr a je potřeba seznam parametrů upravit vůči ukázce z dokumentace.

```

# install.packages("rneos")
library(rneos)
NlistSolversInCategory(category = "co")
template <- NgetSolverTemplate(category = "co", solvername = "concorde",
  inputMethod = "TSP")
modc <- paste(paste(readLines("d493.tsp"), collapse = "\n"), "\n")
argslst <- list(tsp=modc, comment="", email="váš-email", ALGTYPE="con",

```

```

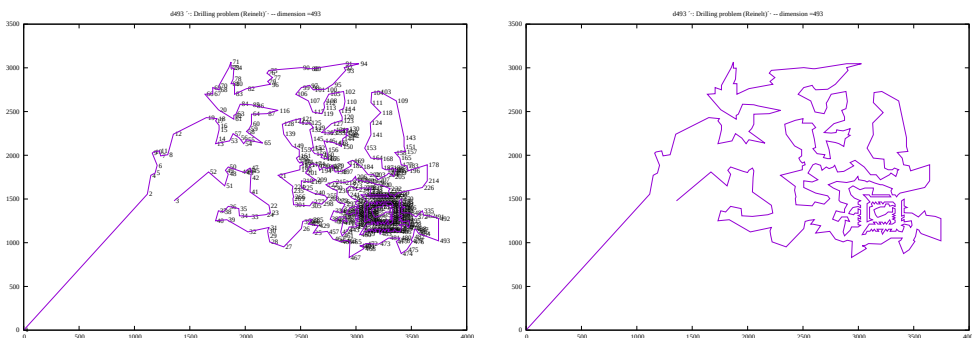
RDTYPE="fixed", PLTYPE="cp")
xmls <- CreateXmlString(neosxml = template, cdatalist = argslist)
NprintQueue()
(test <- NsubmitJob(xmlstring=xmls, user="váš-email", interface=""))
NgetJobStatus(obj = test)
NgetFinalResults(obj = test)

```

Zkusil jsem i novější knihovnu `roi`, tam se pracuje s pluginy. Plugin pro Concorde nevidím, napsal jsem tip autorovi. Seznam pluginů a jejich instalace je rozepsána na <http://roi.r-forge.r-project.org/installation.html>.

Pro náš test stačí webové rozhraní. Navštívíme <https://neos-server.org>, dáme Submit a job to NEOS, dohledáme concorde a naklikneme TSP. Dále vyberáme Browse... pod Concorde data (TSPLIB format file), a užijeme ukázkou z R: `~/R/x86_64-pc-linux-gnu-library/4.4/TSP/examples/d493.tsp`. Můžeme si nastavit další parametry a volíme Submit to NEOS. Za pár okamžiků dorazí výsledek, buď na webové stránce či přes email. Já jsem na serveru registrovaný, takže mohu zvolit pod svým jménem My Jobs a vidím seznam úloh. Volím příslušný řádek ve sloupci Results.

U prvních testů byla na posledním řádku informace, že lze stáhnout PDF, resp. ZIP verzi, mapy. Ale nebylo to klikatelné. Hans Mittelmann, tvůrce rozhraní Concorde na NEOSu, psal, že administrátoři NEOS serveru zrušili možnost sdílet binární soubory, kam spadá i PDF. Zvažoval tu možnost zrušit. Na mou přímluvu zařadil PostScriptový soubor. Podařilo se tak 27. května 2025. Zde jsou ukázky, jak na to jde. V pozadí spouští GnuPlot.



Kdo se nerad pročítá XML specifikacemi a má rád hackování, tak jeden tip. Lze si ve spodní části navolit Dry run. To vygeneruje XML, uložíme si, upravíme `<email>` a soubor `zaslat.xml` zkusíme podstrčit Pythonu, šly by určitě i jiné jazyky.

```
pip3 install neos_client --break-system-packages
```

U seriózní práce opět užijeme virtuální prostředí. A vlastní test by vypadal:

```

from neos_client import NeosClient
client=NeosClient(email="váš-email-na-NEOS-serveru")

```



```
client.submit_job(xml_path="zaslat.xml")
```

Za pár okamžiku dorazí číslo úlohy a heslo a u naší ukázky za pár vteřin se vypíší výsledky a dorazí email. Výstupy jsou ve třech sloupcích. Ze kterého uzlu a do kterého uzlu se spojujeme a na celá čísla zaokrouhlená vzdálenost.

Lze si z takového souboru rychle udělat šablonu, to už nechávám na laskavém čtenáři k bádání. V principu si vzít první řádky, vpassovat vlastní data a napojit poslední řádky. Struktura je:

```
<document>
<category>co</category>
<solver>concorde</solver>
<inputMethod>TSP</inputMethod>
<email> insert valid email address </email>
<dat2><![CDATA[...Insert Value Here...]]></dat2>
<dat1><![CDATA[...Insert Value Here...]]></dat1>
<tsp><![CDATA[...Insert Value Here...]]></tsp>
<ALGTYPE><![CDATA[...Insert Value Here...]]></ALGTYPE>
<RDTYPE><![CDATA[...Insert Value Here...]]></RDTYPE>
<PLTYPE><![CDATA[...Insert Value Here...]]></PLTYPE>
<comment><![CDATA[...Insert Value Here...]]></comment>
</document>
```

Volí se mezi běžnou vzdáleností `dat2`, Manhattan `dat1` či formát TSPLIB. V ALGTYPE se volí `con` (CPLEX), `cqs` (QSort) či `lk` (Lin-Kernighan). V RDTYPE, počáteční hodnota generátoru pseudonáhodných čísel, se volí `fixed` (99) či `variable` (náhodně). V PLTYPE pro vykreslení: `no` (žádné), `cp` (nalezená trasa s uzly) nebo `pf` (bez uzlů).

Výzva. Jak úlohu distribuovat na víc výpočetních strojů. To jsem zatím vzdal.

8. Varianty TSP

Existuje několik zajímavých variant problému obchodního cestujícího. Za prolistování stojí webové stránky TSPLIB95, <http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/>. S těmi nemá Concorde problém, např. asymetrická varianta TSP, uzly definované jako body v prostoru ap.

Pamatuji si variantu TSP ještě z dob studií. U TSP je každý uzel statický, představme si však, že každý uzel má svou trajektorii. Je to známá úloha sestřelení vesmírného odpadu, které jsou na oběžné dráze Země. Pracovně: Dynamický TSP.

Během příprav článků do tohoto sborníku se vyskytla ještě jedna varianta. Vezměme si mapu ČR a SR, například kraje. Řešme TSP a chceme navštívit všechny kraje. Jenže kraj nebude zjednodušen jako uzel (těžiště, administrativní centrum či nějaký zvolený bod), ale je to oblast/polynom, kde se stačí dotknout hranice na libovolném místě. Pracovně: Polynomiální TSP.

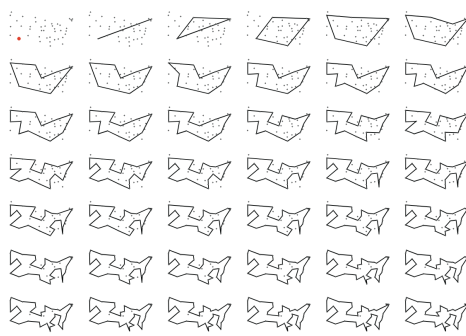
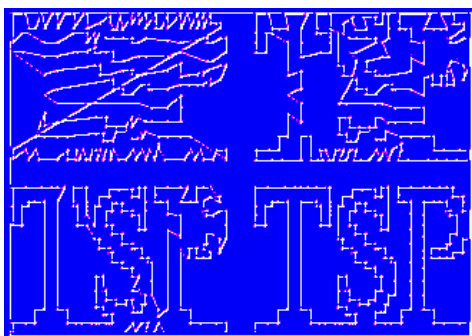
9. Vytyčené úkoly místo Závěru

Koho by tato oblast zajímala víc, doporučuji přednášku profesora Chvátala, <https://slideslive.com/38895310>, a jejich knihu *The Travelling Salesman Problem: A Computational Study* z roku 2007. To je hutnější čtení, kdo by rád o něco jemnější, lze doporučit knihu Williama Cooka *In Pursuit of the Traveling Salesman: Mathematics at the Limits of Computation*, podpořenou přednáškou <https://slideslive.com/39000207>.

Mé starší pokusy, jak získat vektorové obrázky z Concorde, už zastaraly. Jak je vidět, program R to má vymazlené. Pro mne jsou tu typografické výzvy:

- Čísla na hranách se sází s bílým kruhem či obdélníkem. To by chtělo obejít tak, že se hrana vysází jako dvě úsečky.
- Aby se čísla na hranách nepřekrývaly, nabízí se možnost posunout číslo ze středu hrany kousek jinam.

Oba vytyčené úkoly by stálo za hřích vyřešit a nechat spočítat, jen to před OSSConf 2025 nestihnu. Kdyby to bylo hezké, dalo by se to uvážít poslat Hans Mittelmannovi jako výstup z NEOS serveru.



Je zde ještě jedna výzva. Pokud se podíváme na logo TSPLIB95 nebo nahlédneme na str. 66–74 Cookovy knihy, je vidět průběh řešení TSP. To by bylo vhodné z Concorde získat též jako studijní pomůcku. To teprve zkoumám možnosti realizace. Štefan Peško upozorňuje, že ne každý výpočetní krok by se odrazil na mapě. Je to na uvaženu.

V článku jsme si ukázali vykreslení TSP ve 2D, ale Concorde umí i 3D. To by byla výzva za vykreslení, např. přes program Raylib.

Kontaktní adresa

Ing. Pavel Stríž, Ph.D., Nakladatelství Martin Stríž, U Škol 940, Bučovice, okres Vyškov, 685 01, Česká republika,
E-mailová adresa: pavel@striz.cz

Geocaching.com SLAVÍ 25 LET: GCB78KB, GCB78MA, GC8RXDE

PAVEL STRÍŽ (CZ)

Abstrakt. Článek připomíná letošní 25. narozeniny geocachingu. Vybral si k tomu kešky vedoucí na problém obchodního cestujícího. Článek zmiňuje klíčové kroky k jejich vyřešení. Autor také předkládá výzvu hackerům, a to jak automaticky řešit puzzle na jigidi.com.

Klíčová slova. TSP, kešky.

GEOCACHING.COM CELEBRATES ITS 25TH BIRTHDAY: GCB78KB, GCB78MA AND GC8RXDE

Abstract. The paper celebrates geocaching.com's 25th birthday! The selected geocaches lead to the Traveling Salesman Problem, the paper provides key steps how to solve them. There is also an open challenge for hackers: how to solve jigidi.com's puzzles automatically.

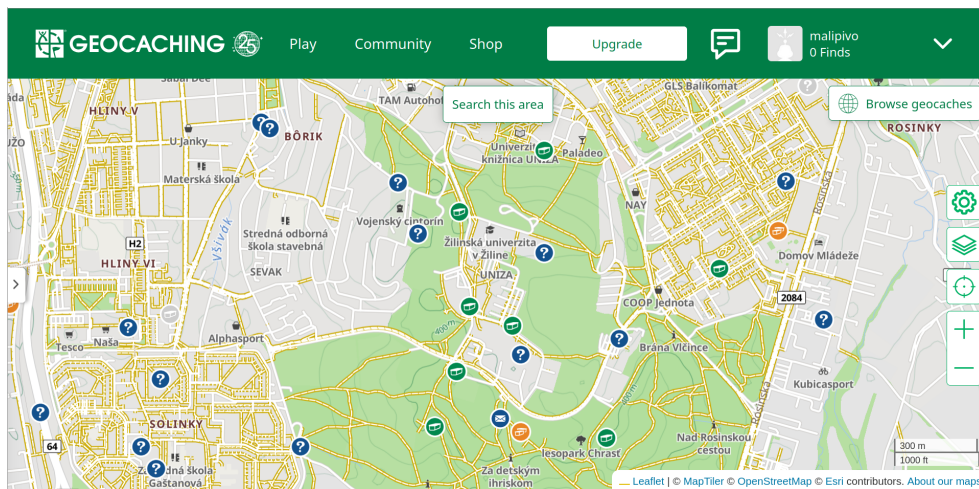
Keywords. TSP, geocaching.

1. O keškování

Pozor, nejsem kačer (hráč geocachingu), byť jsem jich ve skutečnosti pár našel (tzv. odlovil). Okolí mě přimělo několik úloh vyřešit.

Kdybych měl neznalému vysvětlit, o co v keškování jde, tak je to hledání skrytých schránek pomocí GPS navigace. Pro jejich nalezení je často nutné předem vyřešit nějakou úlohu či šifru. Tím získáte GPS souřadnice kešky a můžete vyrazit na lov. Když si tedy vyjdete ven za turistikou, může to být jeden z cílů vašich cest. O svých úspěších si vedete záznamy, můžete se podělit s ostatními, tj. logovat. Lovit chodí jednotlivci, týmy i celé rodiny. Více viz <https://www.geocaching.com/>. V roce 2025 slaví 25. narozeniny. I v Žilině kešky jsou, viz mapa 1 na další straně!



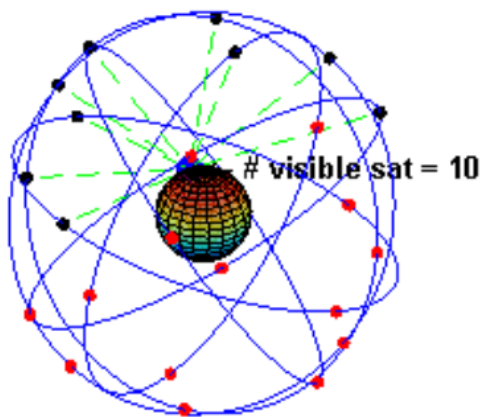


Obrázek 1. Žilinské kešky

2. Má první vzpomínka na kešky

Přišel za mnou spolužák, kačer Radek Vicherek, že náš učitel a cvičitel z Fakulty technologické ve Zlíně, Vosář (víme o vás, chovateli vos!), připravil řadu kešek, od GPS-1 (GC2MBZP) po Satelity – bonus (GC2JE81), a že s některými nepohne. Fajn, říkám si, tak jsem se do nich pustil. Matematicky založené kešky se zvyšující se obtížností. Dozvěděl jsem se zajímavé věci.

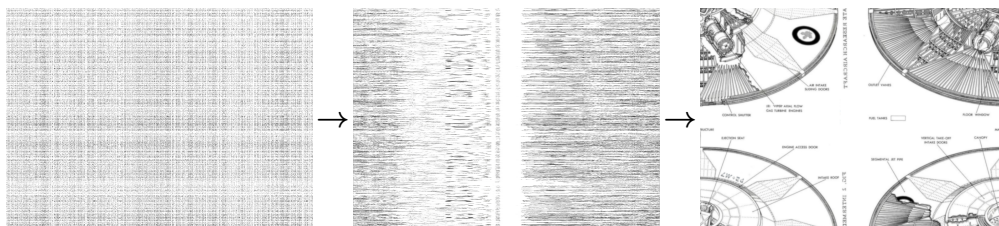
Docela mi to odsýpalo, ale u iGPS (GC2JE7V) jsem se na dlouhé měsíce zasekl. Bylo to hodně dané tím, že se stará data špatně sháněla. A až data byla, řešit to přes matematiku je zapeklitá záležitost. Doporučuji! Zapotil jsem se! Ano, to „i“ v iGPS znamená, že musíte být dostatečně chytrí, dodávám s úsměvem. Pan učitel se mnou ještě jednou hezky zacvičil! Své postřehy jsem zapsal do logu pod přezdívkou malipivo.



3. Zvířátková 12, 14 a Skartovačka

Několik týdnů před konferencí mě kolega poprosil, jestli bych se nepodíval na tyto: GCB78KB, GCB78MA a GC8RXDE. Problém je opět matematický, máme

rastrový obrázek a jsou proházené řádky, případně i sloupce. Úkolem je původní obrázek složit a vyčíst nějaké informace. Konkrétně: zadání, po proházení sloupců a po proházení řádků získáváme řešení (zamíchání se čte zprava doleva):



4. Spoiler Alert!

Celá úloha vede na problém obchodního cestujícího (TSP), kdy matice podobností je nějaká metrika z pixelů příslušných řádků, resp. sloupců. Účelová funkce je minimalizovat zvolenou metriku, jinými slovy se dostat do původního obrázku.

Počet uzlů je počet řádků rastrového obrázku v prvním zamíchání, případně v druhém zamíchání počet sloupců. Konkrétně: Zvířátková 12 je 890 na 579 pixelů, Zvířátková 14 je 871 na 592 pixelů a Skartovačka je 970 na 707 pixelů.

Vedle programu Concorde, viz článek ve sborníku, se svou heuristikou a Domino-Parity je LKH (Lin-Kerningham-Helsgaun) další možná heuristika. Keška vás navede na Github: image-unshredding od Robina Houstona.

První háček je, že musíte užít parametr `--branch`, abyste měli ve zdrojových kódech to, co potřebujete.

```
git clone --branch double-shuffling
https://github.com/robinhouston/image-unshredding.git
```

Další problém je, že kód je zapsán v Python2. Tedy buď musíte na přepis do Python3, či přes virtuální prostředí typu MiniConda.

Ale ani to vám nebude stačit. Program LKH v2.0.7 nezkompilujete na první dobrou. Musel jsem jít do poslední verze 2.0.10 z <http://webhotel4.ruc.dk/~keld/research/LKH/>, a to patřičně nasměrovat.

```
wget http://webhotel4.ruc.dk/~keld/research/LKH/LKH-2.0.10.tgz
tar xvf LKH-2.0.10.tgz
cd LKH-2.0.10; make; cd ..
```

Po probádání Makefile jsem dílčí řešící kroky usměrnil. Soubor `vstupxx.png` je zadání Skartovačky. `bin/lkh.sh` jsem si zkopíroval na `bin/lkh10.sh` a upravil jeden řádek odkazující na adresář s verzí 2.0.10.

```
malfile="xx" # Volba souboru.
# Proházení sloupců.
bin/compute_scores --cols vstup$malfile.png >vstup$malfile-cols.tsp
bin/lkh10.sh vstup$malfile-cols.tsp vstup$malfile-cols.tour
```



```

bin/reconstruct_image.py --cols vstup$malfile-cols.tour
    vstup$malfile.png >vstup$malfile-cols.png
# Proházení řádků.
bin/compute_scores --rows vstup$malfile-cols.png >vstup$malfile-rows.tsp
bin/lkh10.sh vstup$malfile-rows.tsp vstup$malfile-rows.tour
bin/reconstruct_image.py --rows vstup$malfile-rows.tour
    vstup$malfile-cols.png >vstup$malfile-vysledek.png

```

Vězte, že i když tohle všechno víte, nadřete se, poněvadž jsem určitě nějaký krok zapomněl zmínit! Nebudu prozrazovat, jak spustit či si vyčíst z `Makefile` zkompilování `compute_scores.c` či si vyzkoušet simulované žíhání u obrázků Nayuki: úkolem je nahodit si Javu v65.0.

Pozn. I když jsem použil virtuální prostředí pro Python2

```

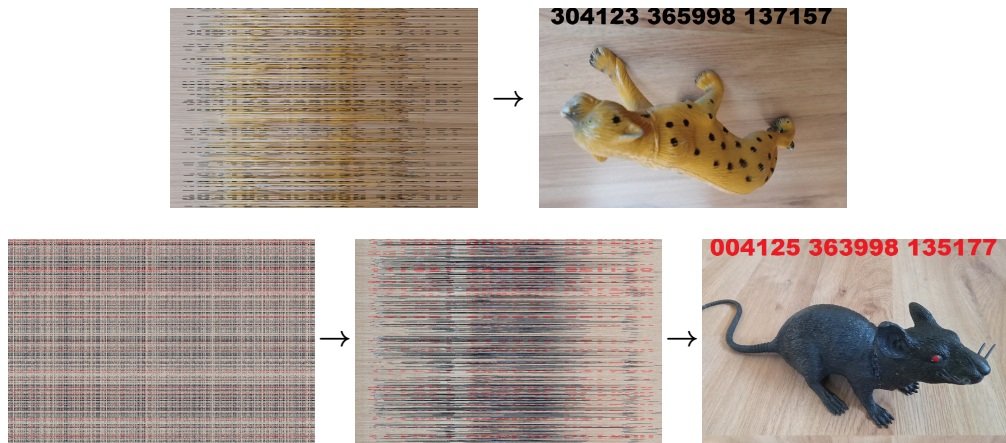
conda create -n python2p7 python=2.7
source activate python2p7

```

program si natahoval knihovnu `random` z knihoven Python3. Tedy zadání nevygeneruji v době psaní tohoto článku, pro kešku to nebylo třeba.

Měl jsem tak trochu dojem, že jsem se ocitl v reálném programovacím světě! Už jen to, že vstupní `.jpeg` soubory musíte exportovat do `.png`, aby se chytla knihovna `Pillow` pod Pythonem, je drobnost, která vás donutí nahlédnout do zdrojových kódů.

V každém případě se mohu pochlubit řešeními Zvířátkové 12 a 14.



Tři šestimístná čísla jsou zakódované souřadnice. Říkají tomu Reverse Wherigo (wherigo je typ keší). Existuje na to online dekodér, <http://www.geekfrog.ca/WIG/ReverseWigDecode.aspx>. Skartovačka obsahuje po sestavení obrázku ještě šifru.

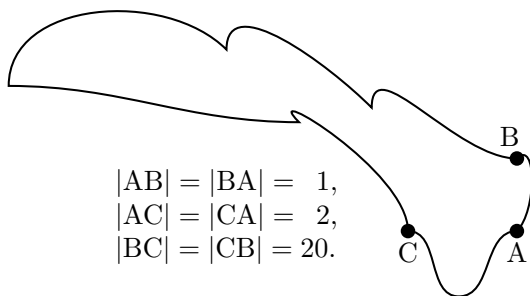
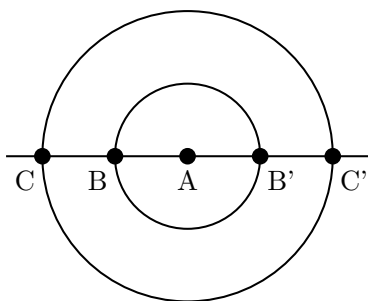
5. Otevřený problém?

Na chvíli jsem se zamyslel, jestli obecnou matici podobností/vzdáleností lze vykreslit do 2D ve formě bodů. Mohu si rovnou odpovědět, že nikoliv. Zde je ukázka pro tři body A, B a C. Máme matici vzdáleností, symetrickou podle hlavní diagonály, s nulami na hlavní diagonále:

$$\begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 20 \\ 2 & 20 & 0 \end{pmatrix}.$$

Pokud si zvolíme bod A a dokreslíme bod B, kružnice o poloměru 1 a bod C, kružnice o poloměru 2, obě se středem v A. Zjistíme, že mezi body B a C nelze získat vzdálenost 20, tedy kružnici o poloměru 20, bez ohledu na to, kde si body B a C na příslušné kružnici vybereme. Nejvíce lze získat tři jednotky, viz $|BC'|$ nebo $|B'C|$. Nejméně by šla jedna jednotka, viz $|BC|$ nebo $|B'C'|$. Nakreslit to neumím, ale řešit TSP lze.

Jistá možnost by byla neužít úsečky, ale nějakou křivku ve stylu hranic na reálné mapě. To by byla skutečná výzva, jak to zautomatizovat, kdyby to bylo užitečné! Dává to možnost užít různé typy křivek, fraktálů apod. Mohlo by to být i esteticky zajímavé, co zvolit. V pravém obrázku naznačím náčrtek, který ukázkovou matici vzdáleností splňuje, vlevo tomu tak není. Těžko odhadovat, jestli by to pomohlo s vizualizací map vah koeficientů u umělých neuronových sítí, což je aktuální téma (AI).



6. Otevřená výzva hackerům!

Nedávno mě kolega poprosil, jestli bych neudělal automatického řešitele na <https://www.jigidi.com/>, že u řady kešek se musí odehrát puzzle, a že to není pro něj. S tímto jsem zatím nepohnul, ale osobně se rád podělím, jak si na tom stojím. Přesněji se tedy chce, aby program ve webovém prohlížeči poskládal puzzle. Nebo to jakkoliv obešel. Po složení puzzle se zpravidla objeví zpráva, která pomůže s keškou.

Buď hackerem a zkus to nějak vymyslet! Kdyby se podařilo, napište mi, kačérům ze svého okolí bych výsledek předal. Pro testování lze použít např. <https://www.jigidi.com/solve/92zdkwnh/1-10/> (GCAQJ2E).

Za inspirativní považuji <https://danq.me/2021/08/26/jigidi-solver/>, byť by to v současnosti neměla být použitelná cesta, dle slov autora. Také existují databáze řešení takových puzzle ve vztahu ke keškám, viz např. <https://solvedjigidi.com/>, kde si lidé sdílejí řešení.

7. Závěrem

Je $\mathcal{P} = \mathcal{NP}$? To já zatím nevím, dodávám s úsměvem, vždycky jsem si myslel, že v žádném případě, ale TSP a vývoj heuristických algoritmů plus rozepsání libovolného problému v třídě $\mathcal{NP}$ pro SAT řešitele dává jistou naději, že jsou si rovny. A jestli k důkazu pomůže nějaká keška, proč ne, že?

Kontaktní adresa

Ing. Pavel Stríž, Ph.D., Nakladatelství Martin Stríž, U Škol 940, Bučovice, okres Vyškov, 685 01, Česká republika,
E-mailová adresa: pavel@striz.cz

MAXIMUM BLOKŮ V KLASICKÉM SUDOKU

PAVEL STRŽÍŽ (CZ) A ŠTEFAN PEŠKO (SK)

Abstrakt. Článek představuje klíčové výsledky úlohy maxima bloků v klasickém sudoku 9×9 , ať už bloky vnitřní nebo všechny (vnitřní + plovoucí). Analyzovali jsme všechna neizomorfní sudoku, užili jsme v průměru 300 procesorů po dobu 7 týdnů.

Klíčová slova. Výpočty přes CPU, klasické sudoku, maximum bloků v sudoku.

MAXIMUM BLOCKS IN CLASSIC SUDOKU

Abstract. The paper presents the key findings of maximum number of inner and all (inner + floating) blocks in classic sudoku 9×9 . We analyzed all non-isomorphic sudokus, on average we used 300 CPUs, they computed the task for 7 weeks.

Keywords. CPU computing, classic sudoku, maximum blocks in sudoku.

1. Úloha maxima bloků v sudoku

Na konferenci OSSConf 2024 jsem prezentoval úlohy a mé pokroky kolem sudoku a generování kombinovaného sudoku. Během diskuze po přednášce docent Štefan Peško zmínil neřešenou úlohu nalezení maxima bloků v klasickém sudoku. Situaci si můžeme ilustrovat následujícím schématem.

<table><tr><td>3</td><td>6</td><td>3</td><td>9</td><td>2</td><td>9</td><td>4</td><td>5</td><td>5</td></tr><tr><td>7</td><td>1</td><td>2</td><td>5</td><td>1</td><td>3</td><td>6</td><td>8</td><td>5</td></tr><tr><td>2</td><td>4</td><td>7</td><td>2</td><td>8</td><td>7</td><td>8</td><td>1</td><td>8</td></tr><tr><td>6</td><td>1</td><td>6</td><td>9</td><td>9</td><td>2</td><td>7</td><td>4</td><td>9</td></tr><tr><td>3</td><td>9</td><td>1</td><td>8</td><td>1</td><td>2</td><td>9</td><td>4</td><td>7</td></tr><tr><td>9</td><td>2</td><td>5</td><td>5</td><td>3</td><td>8</td><td>3</td><td>9</td><td>4</td></tr><tr><td>4</td><td>3</td><td>5</td><td>7</td><td>7</td><td>1</td><td>6</td><td>1</td><td>3</td></tr><tr><td>3</td><td>4</td><td>2</td><td>8</td><td>5</td><td>2</td><td>8</td><td>4</td><td>6</td></tr><tr><td>7</td><td>5</td><td>6</td><td>7</td><td>8</td><td>4</td><td>1</td><td>6</td><td>6</td></tr></table>	3	6	3	9	2	9	4	5	5	7	1	2	5	1	3	6	8	5	2	4	7	2	8	7	8	1	8	6	1	6	9	9	2	7	4	9	3	9	1	8	1	2	9	4	7	9	2	5	5	3	8	3	9	4	4	3	5	7	7	1	6	1	3	3	4	2	8	5	2	8	4	6	7	5	6	7	8	4	1	6	6	shuffle	<table><tr><td>8</td><td>9</td><td>5</td><td>4</td><td>7</td><td>3</td><td>6</td><td>2</td><td>1</td></tr><tr><td>4</td><td>3</td><td>2</td><td>6</td><td>9</td><td>8</td><td>5</td><td>1</td><td>7</td></tr><tr><td>1</td><td>5</td><td>3</td><td>8</td><td>4</td><td>2</td><td>9</td><td>7</td><td>6</td></tr><tr><td>3</td><td>1</td><td>7</td><td>9</td><td>5</td><td>6</td><td>4</td><td>8</td><td>2</td></tr><tr><td>6</td><td>2</td><td>4</td><td>5</td><td>3</td><td>1</td><td>7</td><td>9</td><td>8</td></tr><tr><td>5</td><td>7</td><td>1</td><td>3</td><td>2</td><td>4</td><td>8</td><td>6</td><td>9</td></tr><tr><td>9</td><td>6</td><td>8</td><td>2</td><td>1</td><td>7</td><td>3</td><td>4</td><td>5</td></tr><tr><td>7</td><td>8</td><td>9</td><td>1</td><td>6</td><td>5</td><td>2</td><td>3</td><td>4</td></tr><tr><td>2</td><td>4</td><td>6</td><td>7</td><td>8</td><td>9</td><td>1</td><td>5</td><td>3</td></tr></table>	8	9	5	4	7	3	6	2	1	4	3	2	6	9	8	5	1	7	1	5	3	8	4	2	9	7	6	3	1	7	9	5	6	4	8	2	6	2	4	5	3	1	7	9	8	5	7	1	3	2	4	8	6	9	9	6	8	2	1	7	3	4	5	7	8	9	1	6	5	2	3	4	2	4	6	7	8	9	1	5	3	shuffle	<table><tr><td>3</td><td>7</td><td>9</td><td>8</td><td>4</td><td>1</td><td>2</td><td>6</td><td>5</td></tr><tr><td>1</td><td>6</td><td>2</td><td>7</td><td>9</td><td>5</td><td>4</td><td>8</td><td>3</td></tr><tr><td>5</td><td>8</td><td>4</td><td>6</td><td>2</td><td>3</td><td>7</td><td>1</td><td>9</td></tr><tr><td>7</td><td>9</td><td>6</td><td>3</td><td>8</td><td>4</td><td>1</td><td>5</td><td>2</td></tr><tr><td>4</td><td>1</td><td>8</td><td>5</td><td>7</td><td>2</td><td>3</td><td>9</td><td>6</td></tr><tr><td>2</td><td>5</td><td>3</td><td>9</td><td>1</td><td>6</td><td>8</td><td>4</td><td>7</td></tr><tr><td>8</td><td>2</td><td>5</td><td>4</td><td>6</td><td>7</td><td>9</td><td>3</td><td>1</td></tr><tr><td>6</td><td>4</td><td>1</td><td>2</td><td>3</td><td>9</td><td>5</td><td>7</td><td>8</td></tr><tr><td>9</td><td>3</td><td>7</td><td>1</td><td>5</td><td>8</td><td>6</td><td>2</td><td>4</td></tr></table>	3	7	9	8	4	1	2	6	5	1	6	2	7	9	5	4	8	3	5	8	4	6	2	3	7	1	9	7	9	6	3	8	4	1	5	2	4	1	8	5	7	2	3	9	6	2	5	3	9	1	6	8	4	7	8	2	5	4	6	7	9	3	1	6	4	1	2	3	9	5	7	8	9	3	7	1	5	8	6	2	4	shuffle	<table><tr><td>3</td><td>1</td><td>9</td><td>8</td><td>4</td><td>7</td><td>5</td><td>6</td><td>2</td></tr><tr><td>6</td><td>7</td><td>5</td><td>3</td><td>9</td><td>2</td><td>4</td><td>1</td><td>8</td></tr><tr><td>2</td><td>4</td><td>8</td><td>1</td><td>5</td><td>6</td><td>9</td><td>3</td><td>7</td></tr><tr><td>1</td><td>9</td><td>6</td><td>2</td><td>3</td><td>5</td><td>7</td><td>8</td><td>4</td></tr><tr><td>8</td><td>5</td><td>2</td><td>7</td><td>6</td><td>4</td><td>3</td><td>9</td><td>1</td></tr><tr><td>7</td><td>3</td><td>4</td><td>9</td><td>1</td><td>8</td><td>2</td><td>5</td><td>6</td></tr><tr><td>4</td><td>8</td><td>1</td><td>5</td><td>2</td><td>9</td><td>6</td><td>7</td><td>3</td></tr><tr><td>9</td><td>2</td><td>7</td><td>6</td><td>8</td><td>3</td><td>1</td><td>4</td><td>5</td></tr><tr><td>5</td><td>6</td><td>3</td><td>4</td><td>7</td><td>1</td><td>8</td><td>2</td><td>9</td></tr></table>	3	1	9	8	4	7	5	6	2	6	7	5	3	9	2	4	1	8	2	4	8	1	5	6	9	3	7	1	9	6	2	3	5	7	8	4	8	5	2	7	6	4	3	9	1	7	3	4	9	1	8	2	5	6	4	8	1	5	2	9	6	7	3	9	2	7	6	8	3	1	4	5	5	6	3	4	7	1	8	2	9
3	6	3	9	2	9	4	5	5																																																																																																																																																																																																																																																																																																																																		
7	1	2	5	1	3	6	8	5																																																																																																																																																																																																																																																																																																																																		
2	4	7	2	8	7	8	1	8																																																																																																																																																																																																																																																																																																																																		
6	1	6	9	9	2	7	4	9																																																																																																																																																																																																																																																																																																																																		
3	9	1	8	1	2	9	4	7																																																																																																																																																																																																																																																																																																																																		
9	2	5	5	3	8	3	9	4																																																																																																																																																																																																																																																																																																																																		
4	3	5	7	7	1	6	1	3																																																																																																																																																																																																																																																																																																																																		
3	4	2	8	5	2	8	4	6																																																																																																																																																																																																																																																																																																																																		
7	5	6	7	8	4	1	6	6																																																																																																																																																																																																																																																																																																																																		
8	9	5	4	7	3	6	2	1																																																																																																																																																																																																																																																																																																																																		
4	3	2	6	9	8	5	1	7																																																																																																																																																																																																																																																																																																																																		
1	5	3	8	4	2	9	7	6																																																																																																																																																																																																																																																																																																																																		
3	1	7	9	5	6	4	8	2																																																																																																																																																																																																																																																																																																																																		
6	2	4	5	3	1	7	9	8																																																																																																																																																																																																																																																																																																																																		
5	7	1	3	2	4	8	6	9																																																																																																																																																																																																																																																																																																																																		
9	6	8	2	1	7	3	4	5																																																																																																																																																																																																																																																																																																																																		
7	8	9	1	6	5	2	3	4																																																																																																																																																																																																																																																																																																																																		
2	4	6	7	8	9	1	5	3																																																																																																																																																																																																																																																																																																																																		
3	7	9	8	4	1	2	6	5																																																																																																																																																																																																																																																																																																																																		
1	6	2	7	9	5	4	8	3																																																																																																																																																																																																																																																																																																																																		
5	8	4	6	2	3	7	1	9																																																																																																																																																																																																																																																																																																																																		
7	9	6	3	8	4	1	5	2																																																																																																																																																																																																																																																																																																																																		
4	1	8	5	7	2	3	9	6																																																																																																																																																																																																																																																																																																																																		
2	5	3	9	1	6	8	4	7																																																																																																																																																																																																																																																																																																																																		
8	2	5	4	6	7	9	3	1																																																																																																																																																																																																																																																																																																																																		
6	4	1	2	3	9	5	7	8																																																																																																																																																																																																																																																																																																																																		
9	3	7	1	5	8	6	2	4																																																																																																																																																																																																																																																																																																																																		
3	1	9	8	4	7	5	6	2																																																																																																																																																																																																																																																																																																																																		
6	7	5	3	9	2	4	1	8																																																																																																																																																																																																																																																																																																																																		
2	4	8	1	5	6	9	3	7																																																																																																																																																																																																																																																																																																																																		
1	9	6	2	3	5	7	8	4																																																																																																																																																																																																																																																																																																																																		
8	5	2	7	6	4	3	9	1																																																																																																																																																																																																																																																																																																																																		
7	3	4	9	1	8	2	5	6																																																																																																																																																																																																																																																																																																																																		
4	8	1	5	2	9	6	7	3																																																																																																																																																																																																																																																																																																																																		
9	2	7	6	8	3	1	4	5																																																																																																																																																																																																																																																																																																																																		
5	6	3	4	7	1	8	2	9																																																																																																																																																																																																																																																																																																																																		
I'm random square!		I'm Latin square!		I'm classic sudoku! I've got 9 blocks!		I'm windoku! I've got 13 blocks!																																																																																																																																																																																																																																																																																																																																				

The question is: How many blocks can I have?

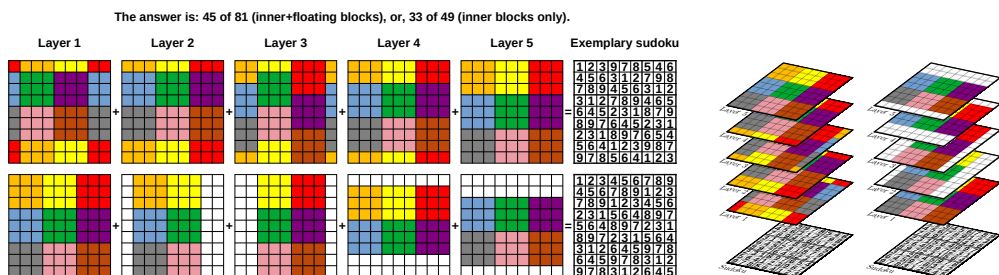
Pokud umísťujeme cifry 1–9 náhodně dostáváme první tabulku. Pokud se mohou cifry 1–9 v řádku a sloupci opakovat právě jednou, dostáváme se na pole Latinských čtverců. Žádný blok se nezvažuje. Pokud takovou podmínku nastavíme i na 9 nepřekrývajících se bloků velikosti 3×3 , dostáváme se ke klasickému sudoku. Pokud bychom dodali další čtyři bloky, dostáváme se k variantě známé jako Windoku. Počet bloků je 13, všechny jsou vnitřní. Takových vnitřních bloků, které se mohou překrývat, existuje $7 \times 7 = 49$.

Pokud bychom dovolili, aby blok přebíhal ze začátku řádku na konec, či podobně u sloupců, pak takové bloky nazýváme plovoucí. Celkový počet možných bloků je $9 \times 9 = 81$, 49 vnitřních a 32 plovoucích.

Naši úlohou je zjistit, kolik bloků ze 49 vnitřních lze užít, aby sudoku bylo platné. Podobně, kolik bloků ze všech 81 lze použít, aby sudoku bylo stále platné.

2. Přístup k řešení

Na tomto místě přeskočíme logické úvahy k vyřešení. Za pomoci lineárního programování brzy zjistíme, že řešení je 33 bloků ze 49 u první varianty úlohy, a 45 z 81 bloků u druhé varianty úlohy. Zde jsou ukázky, od každé varianty jedna. Zkusili jsme vysázet i obrázek s použitím vrstvení pro lepší názornost.



3. Výpočetní část

Zjistit maximum bloků bylo relativně snadné, zjistit která neizomorfní sudoku tomu odpovídají je trochu jiná liga. Zkusíme shrnout naše postřehy. Upravili jsme program `sudoku2.cc` z <http://www.afjarvis.org.uk/sudoku/bertram.html>, a to především tak, že prohazování řádků (2. a 3. superřádek), sloupců (2. a 3. supersloupec), 2. a 3. superřádků a 2. a 3. supersloupců aplikujeme na každé vygenerované sudoku. Generované sudoku jsme nazývali matkami, z prohazování vznikalo $3!^4 \cdot 2 \cdot 2 = 5184$ dětí. Jinými slovy: ohodnotíme každé existující neizomorfní sudoku a zjistíme u něj, kolik různých bloků obsahuje. Dosáhneme-li na maximum bloků (33, resp. 45), je to pro nás žádaný nále. Izomorfismus (prohazení cifer 1–9) zachovává počet bloků i platnost sudoku.

Na jednom jádru vycházely výpočty kolem 900 let. Bylo tedy potřeba výpočty rozhodit na víc počítačů, distribuovat. Honza Šustek z Ostravy nám poradil, ať výpočet spustíme dle počtu jader výpočetního stroje. Další tip nám poslal Miroslav Červenka z Brna, že je dobré si hlídat vlákna na jádro, a spustit úlohu jen na jednom jádru, že si časově moc nepomůžeme. Další jeho postřeh byl, že je potřeba spouštěnou úlohu navázat na jádro, to umí program `taskset` z linuxového balíku `util-linux`.

Ani s pomocí půjčených výpočetních strojů od kamarádů se nám nepodařilo výpočty stáhnout pod roky počítání. Vyzkoušeli jsme několik platforem, bohužel častokrát nám administrátoři či majitelé serverů omezili či úplně zastavili výpočty. Některé platformy byly tak drahé, že ty jsme ani nezkoušeli. Koupili jsme několik 12jádrových serverů, ale klíčové bylo až zakoupení virtuálních serverů na <https://compute.cudo.org/>, kde jsme se dostali až na 250 koupených jader.

U fyzických serverů jsme párování vláken hledali metodou pokus-omyl, až po bitvě jsme našli řešení, že tyto vztahy lze vyčíst z `/sys/devices/system/cpu/cpuN/topology/thread_siblings_list`, kde za N patří číslo jádra. Další možnost byla užít program `cpu-info` z linuxového balíku `cpuinfo`. Zde jsou ještě některé jiné možnosti.

```
lscpu -e # knihovna cpuinfo
lscpu -p # knihovna cpuinfo
lstopo --of console # knihovna hwloc
cat /sys/devices/system/cpu/cpu*/topology/thread_siblings_list|sort|uniq
```

4. Optimalizace kódu

Pořád to nebylo ono. Michal Kaukič z Žiliny poradil, ať se podíváme na optimalizaci kódu. Některé věci v C++ byly zřejmé, namátkou: vyhnout se třídění malého počtu dat, vypnout průběžné výpisy na terminál, pokusit se podmínky `If` převést na práci s poli apod.

Klíčové však bylo jiné zjištění. Jak zjišťujeme počet bloků, dochází ke sčítání čísel, které se častokrát opakují. Uvedeme školní příklad.

```
1+2+3+4+5
1+2+3+6+7
1+2+3+8+9
```

V této ukázce použijeme operaci sčítání celkem dvanáctkrát (tři řádky po čtyřech). To však můžeme přepsat takto:

```
X=1+2+3
X +4+5
X +6+7
X +8+9
```

Nyní operaci sčítání použijeme jen osmkrát, s tím, že `X` si ukládáme do paměti, která nebyla pro nás problém.

Podrobnější zkoumání ukázalo, že se u sudoku dá shlukovat po devíti z možných 81 mezivýpočtů, a následně dvoustupňově, ještě dále tyto mezisoučty shlukovat. Tato banální úvaha eliminovat počet operací sčítání výrazně časově pomohla a dostala nás z let na měsíce počítání. Jinými slovy: počítače jsou výborné v počítání, ale ještě lepší možnost je nepočítat vůbec.

5. Otevřené otázky

Ačkoliv se nám úlohu podařilo spočítat, máme stále několik úvah.

- Zda-li použité shlukování skrz úsporu operací sčítání by se nedalo využít ještě efektivněji s ohledem na zkoumaný problém. Jako optimalizační úloha shlukování se to ukázalo jako nepočítatelné pro všechny možné shluky. Je ale možné, že zde prostor byl.
- Úlohu vnitřních a všech bloků jsme měli ve dvou programech, podařilo se nám je spojit do jednoho. K větší úspoře už nedošlo, ale je stále otázka, jestli jsme něco neopomenuli.
- Úloha maxima bloků v sudoku se dá přepsat do maticového počtu, ale nedokázali jsme toho využít.
- Je také otázka, jestli by tato konkrétní úloha nebyla vhodná pro GPU. K tomuto experimentu jsme se též nedostali.
- Jako u každého výpočtu, hodilo by se nezávislé ověření jinou skupinou lidí. Neměli jsme koho oslovit.

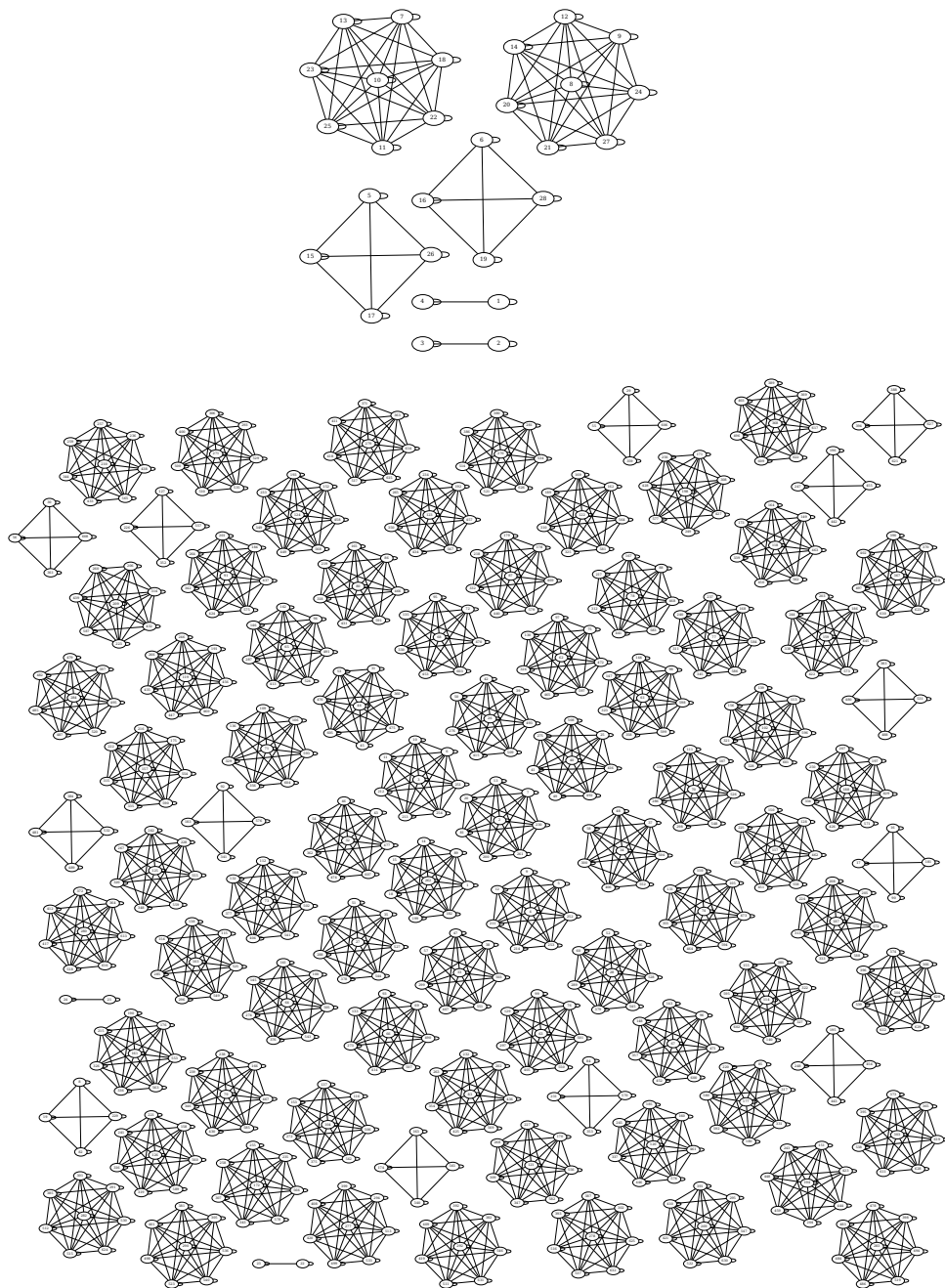
6. Nalezená sudika

Neizomorfních sudoků, která splňují podmínku maxima vnitřních bloků, jsme našli 28. Jsou to tato:

1. 12345678945657891237891234562315648975648972318972315643126458975648978312978312645
12. 12345678945657891237891234563126458975648978312978312645312645897564897231897231564
13. 1237894566123312978546523189756489756423131297854652318975648975642313129785465231
14. 1237894566123312978546523189756489756423131297854652318975648975642313129785465231
15. 1237894566123312978546523189756489756423131297854652318975648975642313129785465231
16. 1237894566123312978546523189756489756423131297854652318975648975642313129785465231
17. 1237894566123312978546523189756489756423131297854652318975648975642313129785465231
18. 1237894566123312978546523189756489756423131297854652318975648975642313129785465231
19. 1237894566123312978546523189756489756423131297854652318975648975642313129785465231
20. 1237894566123312978546523189756489756423131297854652318975648975642313129785465231
21. 1237894566123312978546523189756489756423131297854652318975648975642313129785465231
22. 1237894566123312978546523189756489756423131297854652318975648975642313129785465231
23. 1237894566123312978546523189756489756423131297854652318975648975642313129785465231
24. 1237894566123312978546523189756489756423131297854652318975648975642313129785465231
25. 1237894566123312978546523189756489756423131297854652318975648975642313129785465231
26. 1237894566123312978546523189756489756423131297854652318975648975642313129785465231
27. 1237894566123312978546523189756489756423131297854652318975648975642313129785465231
28. 1237894566123312978546523189756489756423131297854652318975648975642313129785465231

V online verzi sborníku a na adrese <https://invimath.fri.uniza.sk/images/slides/OSSConf/2025-ps01> je výpis neizomorfních sudoků, která splňují podmínku maxima všech bloků. Je jich 632, zde jsou vypsané pouze některé pro ilustraci.





Obrázek 1. Shlukování nalezených sudoku

7. Zahrnutí rotace a transpozice

Na nalezená sudoku aplikujeme 8 symetrií čtverce (rotace, překlápění). Po jejich kanonizaci se dostáváme do jiného nalezeného sudoku. Tímto způsobem se dají shlukovat, viz obrázek ???. Za pozornost stojí, že se shluky neskládají jen z 8 sudoku.

8. Závěrem postřeh

Nalezených 28 neizomorfních sudoku pro vnitřní bloky je součástí 632 neizomorfních sudoku pro všechny bloky. Teoreticky tedy bylo možné počítat jen variantu pro všechny bloky a z nalezených 632 sudoku dopočítat těch 28. Tím by se uspořila cca polovina výpočetního času. Tedy řádově mluvíme o polovině ze 7 týdnů počítání na 300 jádrech.

To jsme bohužel na začátku prací nevěděli a ani jako pracovní hypotéza by nám to nejspíš nepomohlo. Zaujatý čtenář si jistě ověří, že oněch 28 sudoku je k nalezení mezi těmi 632.

Celkový počet sudoku, bez ohledu na izomorfismus, je tedy 28, resp. 632, sudoku krát $9!$ (362 880), tedy permutace 9 cifer.

Ačkoliv máme rádi vše open a nejlépe zadarmo, u této úlohy to bez dokoupených a zapůjčených serverů nešlo. Byla to zajímavá zkušenost, mít na stole takto výpočetně náročnou úlohu. Doporučujeme a už hledáme další! Je to návykové...

Poděkování. Za pomoc s výpočetní silou děkujeme Michalu Kaukičovi, Aleši Kozubíkovi a Rudovi Blaškovi (Žilina), Miroslavu Červenkovi (Brno), Tomáši Výmolovi (Zlín), Petru Nevřivovi (Brno) a Jozefu Říhovi (Bratislava). V neposlední řadě Marku Richmanovi z <https://www.cudocompute.com/> za asistenci při využití jejich virtuálních CPU.

Kontaktní adresy

Ing. Pavel Stríž, Ph.D., Nakladatelství Martin Stríž, U Škol 940, Bučovice, okres Vyškov, 685 01, Česká republika,

E-mailová adresa: pavel@striz.cz

doc. RNDr. Štefan Peško, CSc., Ústecká 1/23, 010 08 Žilina, Slovensko,

E-mailová adresa: pesko@frcatel.fri.uniza.sk

HISTORIE T_EXU A KLÍČOVÝCH NÁSTROJŮ NA GRAFIKU

PAVEL STRÍŽ (CZ)

Abstrakt. Článek je úvodem do historie T_EXu a jeho nástrojů na grafiku.

Klíčová slova. T_EX, METAFONT, METAPOST, TikZ, Asymptote, PSTricks.

THE HISTORY OF T_EX AND ITS MAIN TOOLS FOR GRAPHICS

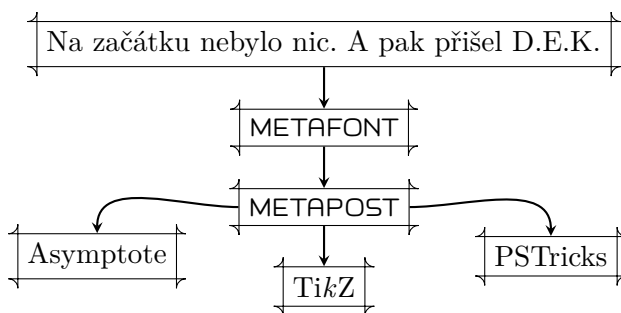
Abstract. The paper introduces very brief history of T_EX and its graphics engines.

Keywords. T_EX, METAFONT, METAPOST, TikZ, Asymptote, PSTricks.

1. Za grafiku

V roce 2009 jsme s Michalem Mádrem napsali článek o historii T_EXu, <https://eudml.org/doc/298866>, a dostali se tak i ke „žhavé“ současnosti, to byl příchod Lua do T_EXového světa.

V tomto článku se pokusím o totéž, 16 let poté (píše se rok 2025). Začnu něčím novým a „méně“ důležitým, a tím je grafika.



METAFONT pracuje na úrovni rastru. V 80. a 90. letech jiných alternativ moc nebylo. To bylo dlouhodobě neúnosné s vývojem výpočetní techniky.

John D. Hobby přišel s nápadem, syntaxi METAFONTu do velké míry zachovat, ale přejít s výstupem do vektorového formátu (PostScript). Vzniká **METAPOST**. Na to navázal Hans Hagen a Taco Hoekwater, který pročistil staré zdrojové kódy (např. vyházel Pascal, sjednotil přesnost výpočtů) a zajistil výstup do SVG. Problém METAPOSTu je, že je omezen na relativně malé hodnoty.

I tak se z něj stává knihovna pro $\text{\TeX}$ (`mplib` a `luamplib`) a stále si drží oblibu. Z výraznějších uživatelů zmiňme Honzu Šustka, Roberta Maříka či zesnulého experta Karla Horáka. Doporučuji nahlédnout na MetaFun Hanse Hageny, <https://www.pragma-ade.nl/general/manuals/metafun-s.pdf>.

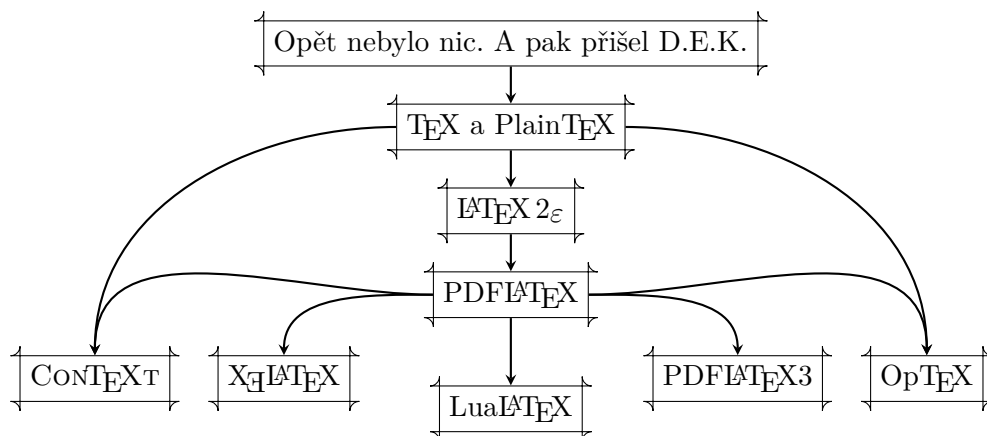
Ať je METAPOST jakkoliv oblíbený a stálíci, vývoj jde dál. Hlavní alternativy jsou tři.

Projekt **PSTricks** pracuje přímo s PostScriptem (jazyk tiskáren). V posledních letech vývoj řídí Herbert Voss z Německa, je to neuvěřitelně činný vývojář a autor mnoha knih. Viz <https://tug.org/PSTricks>.

Druhý směr je program **Asymptote**, <https://asymptote.sourceforge.io/>. Ten se snaží o syntaxi typu C++. Velkou výhodou je možnost generování 3D modelů (WebGL, V3D, PRC). Mezi aktivní uživatele patří například Rudolf Blaško z Žiliny, Roman Plch a Petra Šarmanová. Na toto téma několikrát publikovali.

Jistý přirozený vývoj vedle METAPOSTu je balíček **TikZ**. Till Tantau, mj. i autor balíčku Beamer na přípravu prezentací, šel tou cestou, že zdrojové kódy obrázků jsou součástí $\text{\TeX}$ ového kódu. Můžeme říci, že TikZ se stal standardem v $\text{\TeX}$ ovém světě. Statistický balík R umí obrázky do TikZu exportovat. Existuje celá řada balíčků na TikZu založených. Zmiňme alespoň `pgfplots` na tvorbu grafů, `tikzlings` je balíček na vykreslení zvířátek a mnoho dalších. TikZ spolupracuje s jazykem Lua. Někteří uživatelé si začínají stěžovat na stále rostoucí velikost základního manuálu (nyní 1300+ stran).

2. Za $\text{\TeX}$ ové formáty zjednodušeně (angl. `formats` či `engines`)



$\text{\TeX}$ vyvinul profesor Donald E. Knuth v roce 1978. Na vývoji již nepracuje, jen opravuje chyby. Na této základní úrovni pracuje především Knuth, užívá základní příkazy plus několik vlastních. Přirovnal bych to k jazyku Assembler. Dal však možnost pomocí tzv. formátů rozšířit funkčnost jádra $\text{\TeX}$ u.

Nejstarší formát je tzv. **PlainT_EX**. Je to drobně rozšířené jádro T_EXových příkazů. Mezi oblíbence tohoto formátu patří například Petr Olšák z Prahy. Viz jeho knihy *T_EXbook naruby* a *T_EX pro pragmatiky*. Přirovnal bych to obtížností k C. Toto jádro Petr Olšák rozšiřuje pod formátem **OpT_EX**.

Posun základních příkazů T_EXu do lidsky přijatelnější podoby zrealizovat Leslie Lamport, formát **L^AT_EX** se stal velmi populární. Přirovnal bych to k C versus C++. Většina knih a manuálů se zaměřuje právě na tento formát. Standard, který všichni používáme, je L^AT_EX 2_ε, přechází se na L^AT_EX3, který má atypickou strukturu příkazů. Velmi volně by se to dalo přirovnat k Python2 versus Python3.

L^AT_EX jako takový má výstup do PostScriptu. Díky disertační práci Hàn Thé Thànha na FI MUNI v Brně vzniká formát **PDFL^AT_EX**, který dále rozvíjel Hans Hagen. Z jazyku tiskáren se přešlo do formátu PDF firmy Adobe. To byl na dlouhé roky standard v T_EXovém světě.

Rozšířením pro exotické jazyky je formát **X_qL^AT_EX**. Mluvíme především o jazycích ČJKV, arabštině, perštině ap. U nás je aktivním uživatelem například Zdeněk Wagner z Prahy (hindština).

Zvláštní kategorii tvoří produkční a vývojářské prostředí **ConT_EXt** od Hanse Hageny. Nevýhodou je, že ne všechny L^AT_EXové balíčky lze v ConT_EXtu načíst. To je asi hlavní důvod, proč nemá tolik uživatelů. Je to nejkomplexnější nástroj, chceme-li formát, který T_EXový svět v současnosti má! Aktivní uživatel je např. Tomáš Hála z Brna.

Za současný standard lze považovat **LuaL^AT_EX**, který PDFL^AT_EX rozšiřuje o možnost užít a načítat balíčky pracující s jazykem Lua. Můžeme rozlišit tři kategorie uživatelů:

- běžný uživatel nepozná rozdíl, ale v pozadí mu Lua běží,
- druhou skupinu tvoří T_EXisté, kteří si občas Lua zavolají a pracují s ní aktivně, a,
- poslední skupinu tvoří vývojáři, kteří L^AT_EX3 nebo Lua mají za základ vývoje v T_EXu.

3. O vývojářích

Petr Olšák má svůj formát OpT_EX (Olšák Petr T_EX), který je programátorskou raritou mezi zmíněnými formáty. Je na vývoj sám, nemůže dosáhnout rozvoje, který zajišťuje v T_EXu několik tuctů vývojářů po celém světě.

Mezi „starší“ vývojáře lze považovat Honzu Šustka, Tomáše Hálu, Zdenka Wagnera, Karla Skoupého, Roberta Maříka, Víta Zýku, Karla Píšku, Karla Horáka, Petra Sojku a určitě jsem na mnohé zapomněl.

Nespočet českých a slovenských uživatelů diskutuje a pomáhá na T_EX.SE.

Mezi aktivní popularizátory T_EXu a tvůrce knih lze považovat Jiřího Rybičku z Brna a Pavla Satrapy z Liberce. Překlad LShort do slovenštiny zrealizovali Ján

Buša ml. a st. z Košic, korekturu realizovali Ladislav Ševčovič a Marcel Takáč. Za zvláštní zmínku stojí dnes již neaktivní T_EXonWeb (předchůdce OverLeaf) Honzy Přichystala z Brna. Pustil jim to. :-)

Mezi hledače „bugů“, tedy chyb v programech, zmiňme minimálně Rudolfa Blaška a Aleše Kozubíka, oba z Žiliny, a Miroslava Biňase z Košic. Ale mnoho dalších, které ani neznám!

Na mezinárodním poli T_EXu si vydobili místo Vít Starý Novotný z Brna (současný editor Zpravodaje $\LaTeX$ TUGu po Zdeňku Wagnerovi, balíčky `markdown` a `expltools`), ale hlavně skromný a nenápadný Michal Hoftich z Prahy. Namátkou zmiňme jeho balíčky `tex4ebook`, `Lua-UCA`, `luavlna`, `biblatex-iso690` či `responsive`, více viz <https://ctan.org/author/hoftich>. Pomohl zorganizovat TUG 2024 v Praze! Neuvěřitelný přínos jednoho člověka! Smekáme klobouk!

Díky Vám všem, uživatelům T_EXu, popularizátorům a vývojářům! OverLeaf zhlásil nad 20 miliónů uživatelů T_EXu na své platformě (6/2025)... Možná ten T_EX opravdu někdo používá!

~~~

Profesor Donald E. Knuth byl v Brně v roce 2019, kdo se chce zasmát a dozvědět víc, zde jsou videa a audiopřepisy jejich Q+A, tedy sekce otázek a odpovědí. <https://www.fi.muni.cz/events/2019-10-08-donald-knuth-question-answer-session-computer-programming-as-an-art-brno.html>

## Kontaktní adresa

**Ing. Pavel Stríž, Ph.D.**, Nakladatelství Martin Stríž, U Škol 940, Bučovice, okres Vyškov, 685 01, Česká republika,  
*E-mailová adresa:* [pavel@striz.cz](mailto:pavel@striz.cz)

## MAPA MÍST KONFERENCÍ ROBUST V R

ONDŘEJ VENCÁLEK (CZ) A PAVEL STRÍŽ (CZ)

**Abstrakt.** Článek je úvodem do práce s mapami s dodatečnými popisky ve výpočetním prostředí R. Během naší práce používáme data z Eurostatu plus interní informace o konferencích ROBUST. Vytvořená mapa je zobrazena v R i přes Rscript, vyexportována do png a pdf určeném pro Informační bulletin na tisk, navíc zobrazena ve webovém prohlížeči offline i online za pomoci knihovny `plotly` a jejich prostředí Chart Studio určeném pro práci v týmech a zaměřeném na zveřejňování grafů.

**Klíčová slova.** programovací jazyk R, Eurostat, `sf`, `dplyr`, `ggplot2`, `ggrepel`, `plotly`, Chart Studio, `tccltk`.

### A MAP OF ROBUST CONFERENCE VENUES IN R

**Abstract.** The article is a brief introduction to a map creation in R programming language with additional information. We primarily use data from the Eurostat servers and internal information about the ROBUST conference series. A preview of the map is shown in R and Rscript environments, exported to the png image and the pdf file for printing purposes of the Information Bulletin as well as previewed offline and online using the `plotly` library and their Chart Studio designed for collaborative work and chart publishing.

**Keywords.** R programming language, Eurostat, `sf`, `dplyr`, `ggplot2`, `ggrepel`, `plotly`, Chart Studio, `tccltk`.

## 1. Úvod

Konferenční řada ROBUST má v roce 2025 20+25. výročí a proto jsme se rozhodli vytvořit tzv. „robust(n)í mapu“, tj. mapu Česko-Slovenska se zvýrazněním míst. V neformálním duchu jako jsou tyto konference, možná jako dárek s cílem si zkusit pár fint v R.

## 2. Data z Eurostatu

Z webové stránky <https://ec.europa.eu/eurostat/web/gisco/geodata/reference-data/administrative-units-statistical-units/nuts> stáhneme si soubor zip v řádku NUTS 2016, 14/03/2019, 1:1 Million, sloupec SHP. NUTS neboli Nomenklatura územních statistických jednotek jsou územní celky vytvořené pro statistické účely Eurostatu. NUTS 0 odpovídá státu, NUTS 1 území (ČR), NUTS 2 regionům, NUTS 3 krajům, LAU 1 (dříve NUTS 4) okresům a LAU 2 (dříve NUTS 5) obcím.

Užijeme adresář, kde budeme mít zdrojový kód v R. Konkrétně:

```
$ wget https://ec.europa.eu/eurostat/cache/GISCO/distribution/\
> v2/nuts/download/ref-nuts-2016-01m.shp.zip
```

Ze staženého souboru potřebujeme extrahovat jen jediný. Na vykreslení mapy nám stačí.

```
$ unzip ref-nuts-2016-01m.shp.zip NUTS_RG_01M_2016_4326.shp.zip
```

RG značí regiony (angl. regions; vedle toho najdeme BN/boundaries pro hranice a LB/labels pro popisky), 01M je vybrané měřítko, 2016 je rok regulace statistické jednotky (aktualizace 14. 3. 2019) a 4326 značí EPSG:4326, také známé jako geografická projekce WGS84 v desetinách stupňů. LEVL odpovídá NUTS.

Ostatní soubory jsou v dalších geografických projekcích (EPSG:3035 pro Evropskou projekci LAEA a EPSG:3857 pro WGS84 / Pseudo-Mercator), menších územních celcích a rozlišené dle regionů, hranic a popisků.

Tento soubor kompletně rozbalíme:

```
$ unzip NUTS_RG_01M_2016_4326.shp.zip
```

Získáme tak hlavní soubor vektorových dat (shp), indexový soubor (shx) a databázový soubor (dbf). Tyto soubory jsou povinné, vedle toho získáme dva doplňkové soubory: zaznačení kódové stránky pro dbf (cpg) a přesné informace o projekci (prj).

### 3. Softwarová příprava

Pokud nemáme, nainstalujeme si R a potřebné knihovny, v našem případě minimálně `libssl-dev`, bude jej potřebovat `plotly` u interaktivní online verze:

```
$ sudo apt install r-base-core
$ sudo apt install libssl-dev
```

Ve výpočetním prostředí R si doinstalujeme knihovny:

```
$ R
> install.packages(c("sf","dplyr","ggplot2","ggrepel","plotly","tcltk"))
> q()
```

Ideální je instalovat knihovny jednu za druhou, skrz možné chybové zprávy. Dále si vytvoříme soubor `robust-celek.R`, který si postupně okomentujeme.

Může se vám hodit `TeXLive`, `ImageMagick` nebo `GraphicsMagick`, prohlížeč rastrových obrázků (např. `ristretto` či `GIMP`) a prohlížeč pdf souborů (např. `Okular` či `XpdfReader`). Necháváme na laskavém čtenáři, jak hluboce se chce do článku ponořit.

## 4. Stati(sti)cká mapa

Načteme si potřebné knihovny:

```
1 library(sf); library(dplyr); library(ggplot2)
2 library(ggrepel); library(tcltk); library(plotly)
```

Připravíme si data o konferenční řadě v proměnné `rob(ust)`.

```
3 ### Data o ROBUSTech ###
4 rob = data.frame(
5   location = c("Načetín", "Kost", "Slavonice", "Adršpach", "Plasy",
6   "Liblice", "Herbertov", "Malenovice", "Lednice", "Radešín",
7   "Nečtiny", "Hejnice", "Třešť", "Lhota nad Rohanovem", "Pribylina",
8   "Králíky", "Němčičky", "Jetřichovice", "Kurzovní", "Rybník", "Bardějov"),
9   longitude = c(13.2634230, 15.1350694, 15.3195072, 16.1025647, 13.3838904,
10  14.5853101, 14.3358220, 18.4153645, 16.8023564, 16.0841326,
11  13.1450920, 15.2190330, 15.4885601, 13.6832665, 19.7801583,
12  16.7519537, 16.8280667, 14.3837522, 17.2096600, 12.6724633, 21.2713128),
13  latitude = c(50.5489359, 50.4902238, 49.0149020, 50.6203202, 49.9340679,
14  50.3237900, 48.6200863, 49.5678849, 48.8082447, 49.4666355,
15  49.9541085, 50.8502057, 49.3102300, 49.1399375, 49.1334503,
16  50.1007519, 48.9331628, 50.8655207, 50.0785817, 49.5053812, 49.2926572),
17  Termín = c("letní", "zimní", "letní", "zimní", "letní",
18  "zimní", "letní", "zimní", "letní", "zimní",
19  "letní", "zimní", "letní", "zimní", "letní",
20  "zimní", "letní", "zimní", "letní", "zimní", "2020"),
21  roky = seq(1980, 2020, by=2)
22 )
```

Načteme si mapu a vytvoříme si základ v proměnné `p(lotly)`.

```
23 ### Základ mapy ###
24 map = st_read("NUTS_RG_01M_2016_4326.shp", stringsAsFactors=FALSE)
25 CZO = map %>% filter(NUTS_ID %in% c("CZ", "SK"))
26 p = ggplot(rob) +
27   geom_sf(data = CZO, show.legend = FALSE, color="black") +
28   xlab("") + ylab("") +
29   geom_point(data=rob,
30     aes(x=longitude, y=latitude, fill=Termín),
31     pch=21, size=4, alpha=I(0.7)) +
32   scale_fill_manual(values=c("green", "red", "blue")) +
33   theme_void() +
34   theme(
35     legend.position="bottom",
36     legend.title = element_blank(),
```

```

37     panel.grid.major = element_line(linetype="blank"),
38     panel.grid.minor = element_line(linetype="blank")
39 )

```

Náš poslední krok je přidat popisky.

```

40 ### Přidání popisku, statická verze ###
41 pfinal = p + geom_label_repel(data=rob, aes(x=longitude, y=latitude,
42     label=location), cex=3)

```

Nyní se můžeme podívat na výsledek:

```

43 ### Zobrazení v prostředí R ###
44 pfinal

```

Pokud bychom rádi mapu viděli přes `Rscript`, můžeme užít `X11()` pro operační systém Linux, resp. `windows()` či `quartz()` pro Microsoft Windows a Mac OS X. Pozastavení zrealizujeme přes knihovnu `tcltk`.

```

45 ### Zobrazení přes Rscript ###
46 X11() # windows() nebo quartz()
47 pfinal
48 prompt <- "Mezerníkem se zavře graf"
49 capture <- tk_messageBox(message = prompt)
50 dev.off()

```

Dále si ukážeme uložení do pdf.

```

51 ### Uložení do PDF, verze tisková ###
52 cairo_pdf("Robust.pdf", width=10, height=7)
53 pfinal
54 dev.off()

```

Pokud bychom si přáli ořezat ochrannou bílou zónu, lze to udělat přes příkaz

```
$ pdfcrop --hires --margins 1 Robust.pdf
```

Nástroj `pdfcrop` je součástí  $\text{T}_{\text{E}}\text{X}$ ové distribuce  $\text{T}_{\text{E}}\text{XLive}$ . Nula se zpravidla nepoužívá kvůli přesahům Béziových křivek v písmech.

Jiná možnost je uložit si mapu do rastrového obrázku a ten si zobrazit přes prohlížeč obrázků.

```

55 ### Uložení do PNG, verze na web ###
56 png("Robust.png")
57 pfinal
58 dev.off()
59 browseURL("Robust.png")

```

Pokud chceme ořezat ochrannou bílou zónu i zde, lze i to:

```

$ gm convert Robust.png -flatten -fuzz 1% -trim +repage Robust-crop.png
$ # convert Robust.png -flatten -fuzz 1% -trim +repage Robust-crop.png

```

## 5. Interaktivní mapa offline

Náš další krok je připravit si verzi pro internet. Inspiraci lze hledat na serveru <https://plot.ly/r/>.

Prvně zasáhneme do vzhledu mapy:

```

60  ### Přidání popisků, interaktivní verze ###
61  p = ggplot(rob) +
62    geom_sf(data = CZ0, show.legend = FALSE) +
63    xlab("") + ylab("") +
64    geom_point(data=rob,
65      aes(x=longitude, y=latitude, fill=Termín,
66        text=paste(roky,'<br>',location)),
67      pch=21, size=4, alpha=I(0.7)) +
68    scale_fill_manual(values=c("green","red","blue")) +
69    theme_void() +
70    theme(
71      legend.position="bottom",
72      panel.grid.major = element_line(linetype="blank"),
73      panel.grid.minor = element_line(linetype="blank")
74    )
75  p <- ggplotly(p, tooltip= c("text")) # c("location"))
76  p <- p %>%
77    layout(legend = list(
78      orientation = "h", x = 0.4, y =0
79    ))

```

Zobrazit si mapu v R v offline režimu je jednoduché.

```

79  ### Zobrazení mapy bez nutnosti účtu na plot.ly ###
80  p

```

Na první dobrou se nám nepodařilo mapu zobrazit přes `Rscript`. Jeden ze zajímavých tipů na internetových fórech bylo místo `p` použít jednu z možností (namátkou zde <https://stackoverflow.com/questions/26643852/ggplot-plots-in-scripts-do-not-display-in-rstudio>):

```

p %>% print
print(p)
show(p)

```

To sice začalo fungovat, pokud jsme blok kódu volali v R přes příkaz `source` a v RStudio, ale nikoliv přes `Rscript`.

Řešení jsme objevili na <https://stackoverflow.com/questions/44048347/r-open-plotly-in-standalone-window>.

Nemáme-li, doinstalujeme si webové prohlížeče:

```
$ sudo apt install chromium-browser
$ sudo apt install firefox
```

Návod pro chromium-browser funguje bezvadně.

```
81 print_app <- function(widget) {
82   temp <- paste(tempfile('plotly'), 'html', sep = '.')
83   htmlwidgets::saveWidget(widget, temp, selfcontained = FALSE)
84   system(sprintf("chromium-browser -app=file://%s", temp))
85   # system(sprintf("firefox file://%s", temp))
86   # system(sprintf("sleep 5"))
87   temp
88 }
89 print_app(p)
```

Ovšem u prohlížeče **firefox** se dočasný soubor smaže dřív, než jej prohlížeč dokáže otevřít. Řešení je pozastavit funkci **print\_app** a jsou to zakomentované řádky 85 a 86.

## 6. Interaktivní mapa online

Náš poslední robustní úkol je mapa v online režimu.

- Otevřeme si <https://chart-studio.plot.ly/>.
- Naklikneme Sign Up.
- Vyplníme potřebné registrační údaje. Dáme SIGN UP.
- Potvrdíme údaje přes odkaz v došlém emailu.
- Po najetí na vytvořené uživatelské jméno (vpravo nahoře) dáme Settings.
- Vlevo v menu zvolíme API Keys.
- Přes Generate Key si necháme vytvořit API klíč.
- Někam si uživatele a vytvořený API klíč uložíme. Hned je použijeme. Heslo je dobré si též uložit, ale v R jej potřebovat nebudeme.

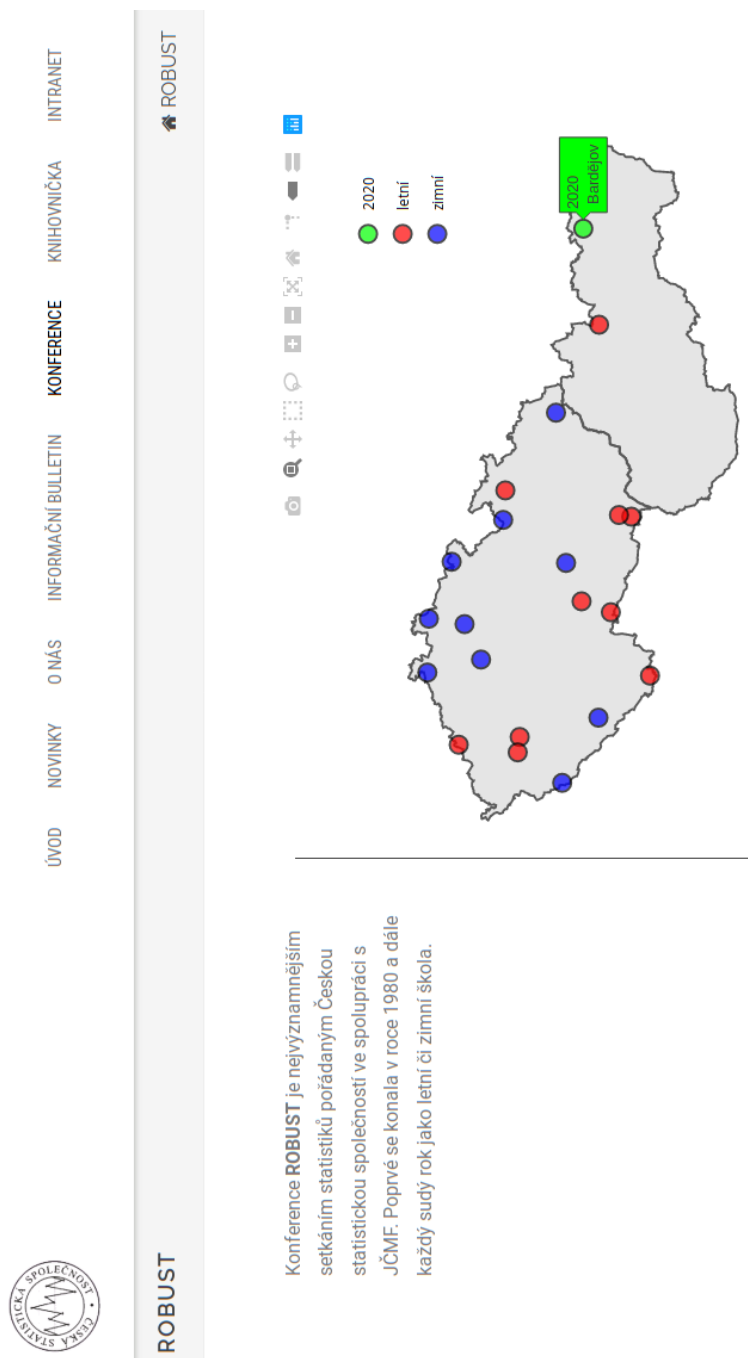
Vrátíme se zpět do zdrojového kódu a zapišme:

```
90 ### Zobrazení mapy na serveru plot.ly ###
91 Sys.setenv("plotly_username"="VÁŠ UŽIVATEL") # <-- zasáhněte
92 Sys.setenv("plotly_api_key" ="VÁŠ API KLÍČ") # <-- zasáhněte
93 chart_link = api_create(p, filename="Robust-2020")
94 chart_link
```

Proběhlo-li vše v pořádku, otevře se webový prohlížeč a vznikly (nebo se aktualizovaly) dvě záložky: Robust-2020 Grid a Robust-2020.

Pracovní verze mapy je dostupná přes stránku [https://chart-studio.plot.ly/create/?fid=vencalek:5#](https://chart-studio.plot.ly/create/?fid=vencalek:5#/)/, zkrácený odkaz je [plot.ly/~vencalek/5](https://plot.ly/~vencalek/5).





Obrázek 1. Výsledná mapa

Nezávislý test je uložen na [https://chart-studio.plot.ly/create/?fid=malipivo:3#/,](https://chart-studio.plot.ly/create/?fid=malipivo:3#/) resp. [plot.ly/~malipivo/3](https://plot.ly/~malipivo/3).  
Gratulujeme! Jsme hotoví.

## 7. Mapa na produkčním serveru

Webové stránky ČStS – [statspol.cz](http://statspol.cz) – jsou přepracované. Pokud z menu vybereme KONFERENCE a dále ROBUST, mapa se otevře v pravé horní části. Přímý odkaz na výsledek našich snah je <http://www.statspol.cz/konference/robust/>.

Knih a materiálů je nespočet, zájemce o grafiku v R s knihovnou `plotly` odkazujeme na knižní přírůstek z ledna 2020 [1].

## Reference

- [1] SIEVERT, C.: *Interactive Web-Based Data Visualization With R, plotly, and shiny*, 1<sup>st</sup> edition, Chapman & Hall/CRC The R Series, January 21, 2020, pp. 448, ISBN 978-1138331457. on-line version available at: <https://plotly-r.com/>.

## Kontaktní adresy

**doc. Mgr. Ondřej Vencálek, Ph.D.**, Katedra matematické analýzy a aplikací matematiky, Přírodovědecká fakulta Univerzity Palackého, 17. listopadu 12, 771 46 Olomouc,  
*E-mailová adresa:* [ondrej.vencalek@upol.cz](mailto:ondrej.vencalek@upol.cz)

**Ing. Pavel Stríž, Ph.D.**, U Škol 940, Bučovice, okres Vyškov, 685 01, Česká republika,  
*E-mailová adresa:* [pavel@striz.cz](mailto:pavel@striz.cz)

## RECENZE KNIHY OPT ART OD ROBERTA BOSCHE

PAVEL STRÍŽ (CZ)

**Abstrakt.** Tohle je knižní recenze na knihu *Opt Art – From Mathematical Optimization to Visual Design* od Roberta Bosche z roku 2019.

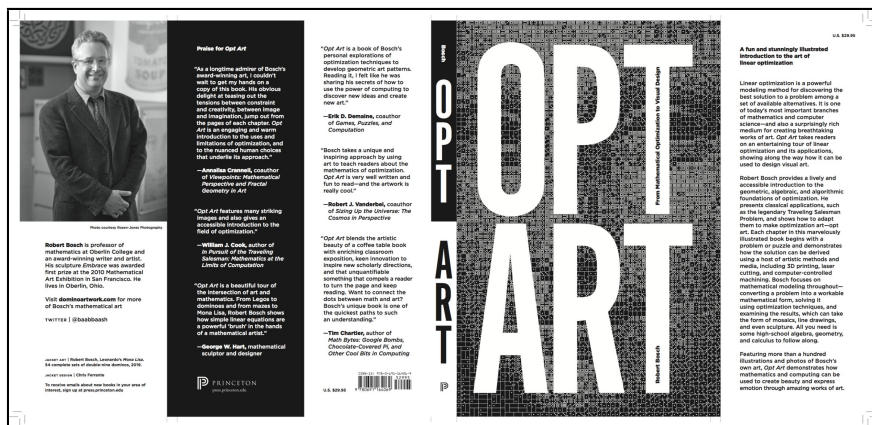
**Klíčová slova.** Optimalizace, umění, problém obchodního cestujícího.

### BOOK REVIEW OPT ART

**Abstract.** This is a book review on *Opt Art – From Mathematical Optimization to Visual Design* by Robert Bosché, 2019.

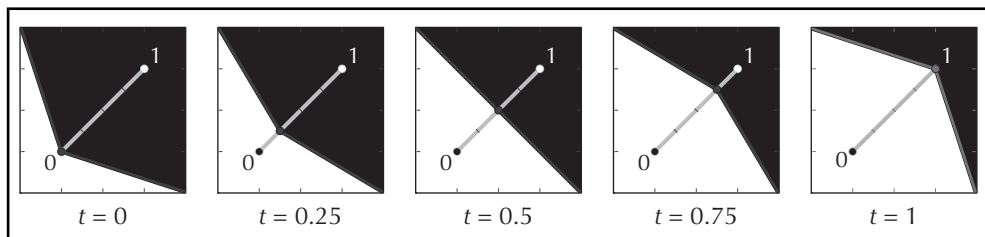
**Keywords.** Optimization, Art, TSP.

V roce 2019 vyšla v nakladatelství Princeton University Press v USA kniha Roberta Bosche: *Opt Art – From Mathematical Optimization to Visual Design*, xii+188 stran, ISBN 978-0-691-16406-9.



Kdybychom měli knihu shrnout do jedné věty, tak je to ukázka matematické optimalizace s výstupy do světa vizualizace a umění. Tedy místo Op-art (optické umění) přechází do Opt-art (optimalizační umění).

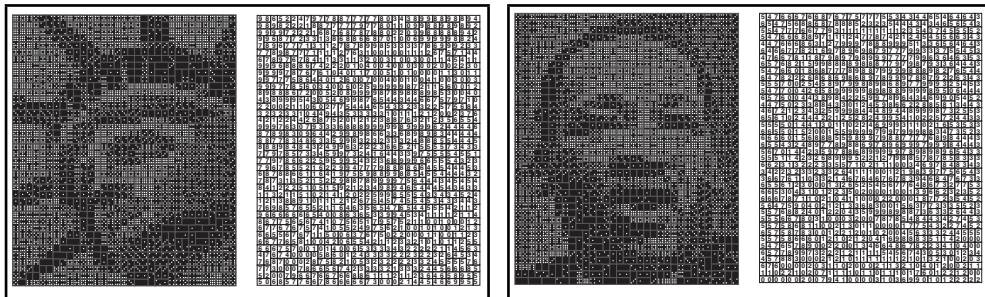
**První dvě kapitoly** uvádí do problematiky příkladem flexibilních Truchetových dlaždic. Stupeň šedi rastrových obrázků pasuje na vhodnou dlaždici. Optický klam pak odvede svou práci: zblízka člověk vnímá jednotlivé dlaždice, z větší dálky člověk vnímá původní obrázek. V druhé kapitole knihy pak zkouší některé další flexibilní dlaždice.



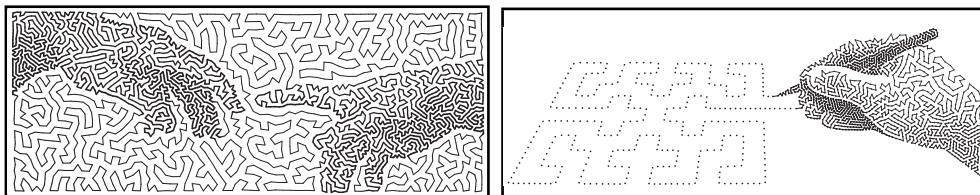
Pro méně znalé uvádí problematiku lineárního programování (principy, Simplexův algoritmus, Branch-and-Bound) v **třetí kapitole**. Na pomoc si bere program Gurobi, <https://www.gurobi.com>. Zaujalo mě, že si bez větších problémů nechává vypsat více/všechna řešení, to není u lineárního programování úplně obvyklé. Všiml jsem si, že v názvech proměnných používá čárku, to si nevybavuji, že by jiný program uměl. Tedy doslova, např. místo neznámé  $X_{A,j}$ ,  $j = 1, \dots, 5$ , používá  $XA,1$  až  $XA,5$ .

V **kapitole čtvrté** zmiňuje přiřazovací úlohu, tedy úvod do binárního lineárního programování. Pixels rastrových obrázků nahrazuje různými obrázky či dílky. Ať už se jedná o kreslené obrázky, cifry, nebo domino kostky.

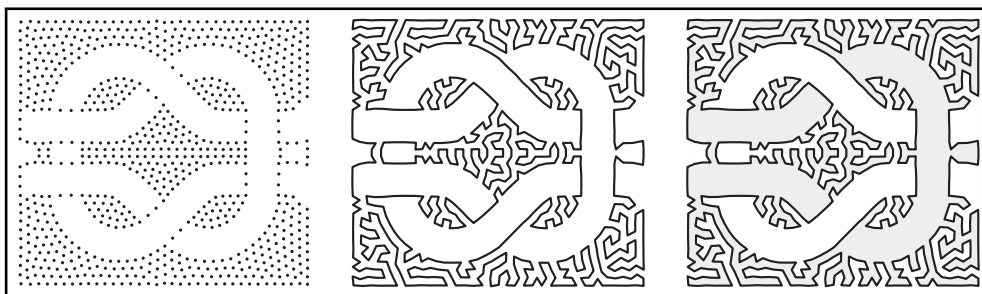
Experimenty s nimi pak provází celá **pátá kapitola**. Kde je to potřeba, přidává celé sady domino kostek. Zmiňuje, že obrázky Martina Luthera Kinga Jr., či bývalého prezidenta Baracka Obamy jsou dětmi složeny a vystaveny po celých Spojených státech. Autor na svých webových stránkách má rozkresy.



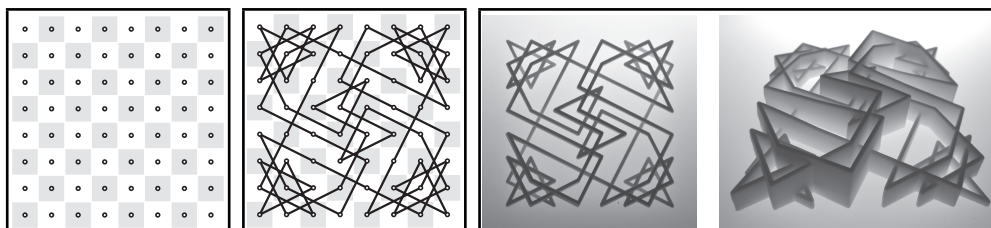
**Kapitola šestá** potěší příznivce TSP, tedy problém obchodního cestujícího. Z trasy vzniká původní obrázek. Uvádí metodu Branch-and-Cut a heuristiku Lin-Kerninghana, vedle lineárního programování v Gurobi si bere na pomoc program Concorde, viz <https://www.math.uwaterloo.ca/tsp/concorde.html>. Našince potěší, že tento nyní již legendární program spoluvytváří Vašek Chvátal (<https://www.math.uwaterloo.ca/tsp/concorde/contact/team.htm>), absolvent pražské MFF. Za pozornost stojí metoda stippling (algoritmus MacQueena), tedy převod obrázku do série bodů. Seriózní badatele by měla zaujmout soutěž *The Mona Lisa TSP Challenge*, tedy snaha najít globální optimum TSP pro 100 tisíc bodů. Zatím vede Yuichi Nagata se svou optimalizací z roku 2009.



Autor v kapitole sedmé dále zkoumá možnosti optimalizace TSP, konkrétně užití více tras a práci se symetriemi. Inspiraci našel v díle Rinuse Roelofse. On se totiž cyklus dá vymalovat barvou. Kapitola obsahuje nespočet obrázků, včetně ukázek laserového a vodního řezání. Vznikají tedy unikátní projekty ze dřeva a kovů. Typografy potěší zmínka konstrukce Beziérový křivky jako metoda vyhlazování dvou či více bodů.



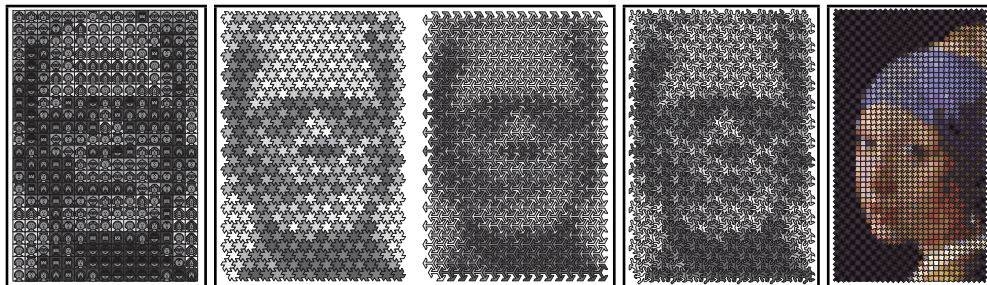
**Kapitola osmá** je lahůdka. Autor si bere do parády cestu šachového koníka na šachovnici  $8 \times 8$ , a i když ví, že neexistuje rotační či zrcadlová symetrie takové cesty, hledá cestu, která se ideální symetrické nejvíc blíží. Pracuje se slovíčkem *nearly*, např. *nearly 4-fold symmetry*, volně přeloženo jako *téměř rotačně symetrické*. Vedle obrázků potkáme fotky 3D tisků.



V kapitole deváté se vrací k mozaice, tentokrát na konstrukci bludišť. Za stavební prvky si bere možnosti dílku bludiště (jak vypadá trasa a stěny) a bludiště si nechává vygenerovat za nejrůznějších omezení.

V tomto experimentu pokračuje i v kapitole deváté, tentokrát však transformuje stavební prvky mozaik. Aby barvy netvořily velké shluky, nastavuje omezení na oblast. Došlo i na mozaiku inspirovanou Escherem, včetně méně známé mozaiky Cyrila Stanleyho Smitha, která měla tvořit alternativu Truchetově mozaice. Gurobi už nepočítá vteřiny, jako v úvodu knihy, ale hodiny a dny. Kdo

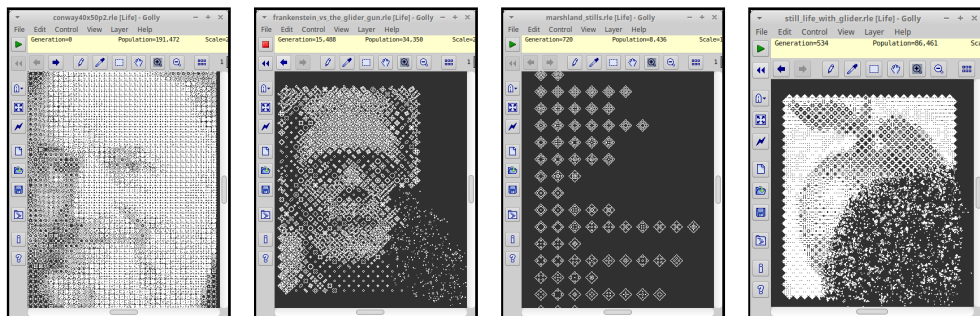
by si přál komerční program Gurobi vyzkoušet, zkuste si jej přes NEOS server, <https://neos-server.org>.



V poslední **kapitole jedenácté** zkouší vizualizaci přes Game of Life Johna Hortona Conwaye. Ať už nalezením symetrických dílků či zajímavý pokus optimalizovat dva snímky tohoto celulárního automatu.

Knize nelze nic vytknout. Jedinečný projekt s mnoha inspirativními nápady podpořena řadou obrázků a fotek. Knihu může číst člověk neznalý matematiky, ale i matematik, neb závěrečné kapitoly knihy jsou výzvou. Jako příznivec volně dostupných programů bych místo Gurobi uvítal Scip či Symphony, ale jak by řekli počítačová hackeři: Musíme si umět poradit.

Myslím si, že pokud ze svých kolekcí vytáhneme domino kostky a začneme experimentovat, tak kniha splnila svůj účel. Za pozornost stojí přednáška autora knihy na *Talks at Google*, <https://www.youtube.com/watch?v=g3CiVrN-BnY>. Knihu podporují autorovy webové stránky, <http://dominoartwork.com/>, a několik animací na <https://www.youtube.com/user/baabbaash>. V sekci *Downloads* najdeme RLE soubory (Game of Life Patterns), spustitelné například přes program Golly, <https://golly.sourceforge.io>.



## Kontaktní adresa

**Ing. Pavel Stríž, Ph.D.**, Nakladatelství Martin Stríž, U Škol 940, Bučovice, okres Vyškov, 685 01, Česká republika,

*E-mailová adresa:* [pavel@striz.cz](mailto:pavel@striz.cz)



## OPEN SOURCE METEOROLOGICKÁ STANICA

BARTA Dávid (SK)

Open Source softvér a Open Source hardvér, špeciálne RaspberryPi, v akcii. Prezentácia výsledkov čerstvo obhájenej bakalárskej práce na FRI UNIZA.

Cieľom tejto Open Source aplikácie bolo vytvoriť systém na zber, spracovanie a vizualizáciu environmentálnych dát. Riešenie pozostávalo z hardvérovej časti zabezpečujúcej snímanie údajov s pomocou senzorov a zberného modulu Raspberry Pi 3B a zo softvérovej časti, ktorá umožňovala ich uchovanie a prehľadné zobrazovanie cez webové rozhranie. Aplikácia podporovala zobrazenie údajov v reálnom čase aj prácu s historickými dátami. Na prenos dát medzi komponentmi boli využité technológie MQTT a back-endové REST API. Dôraz sa kládol najmä na jednoduchosť používania, čisté rozhranie a možnosť nasadenia v rôznych prostrediach — od domácich zostáv až po školské alebo výskumné projekty.

## RASPBERRY PI AKO KIOSK ALEBO „MISSION: KIOSKPOSSIBLE“

BIŇAS Miroslav (SK)

Minipočítač Raspberry Pi je obľúbeným hrdinom mnohých domácich kutilských projektov. Jedným z nich je aj (webový) kiosk. Ale nie hocijaký myslíme tým kiosk, ktorý beží spoľahlivo 24/7. Od prvého zapnutia. Taký, ktorý sa dá spravovať na diaľku. A keď nie na diaľku, tak aspoň na blízko. Ktorý v prípade potreby vypnete a zapnete ručne – a on bude aj tak fungovať. Rovnako. Aj zajtra.

Presne o tom bude tento príspevok – ako vytiahnuť Raspberry Pi zo šuflíka a spraviť z neho spoľahlivý kiosk. Bude to trochu technické, no veľmi praktické. Pripravte sa na príručku plnú tipov, chýb a víťazných „Aha!“ momentov, ktoré vám pomôžu zachrániť vaše duševné zdravie.

## AKO NAŠI ŠTUDENTI PÍŠU ZÁVEREČNÉ PRÁCE V $\text{\LaTeX} 2_{\epsilon}$

BIŇAS Miroslav (SK)

Naši študenti odovzdávajú záverečné práce v typografickom systéme  $\text{\LaTeX}$  povinne už viac než sedem rokov. Ročne tak vznikajú stovky prác a s nimi aj množstvo drobných nástrojov a overených postupov, ktoré sme v priebehu rokov zapracovali do celého

procesu. Tento príspevok ponúka prehľad aktuálneho stavu šablóny v roku 2025, náčrt jej ďalšieho potenciálu a zároveň predstavuje náš agilný prístup k písaniu a hodnoteniu záverečných prác.

Ak uvažujete nad tým, že by aj vaši študenti mohli pri tvorbe záverečných prác používať L<sup>A</sup>T<sub>E</sub>X, pridte sa inšpirovať – ukážeme vám, ako to môže fungovať v praxi.

## O BEZPEČNOSTI V INTERNETE OVLÁDANOM TOASTERMI

BIŇAS Miroslav (SK)

Legenda hovorí, že písmeno „S“ v skratke IoT znamená „bezpečnosť“. Aa tak, ako v skratke chýba písmeno S, tak sa bezpečnosti venujeme nedostatočne – ak vôbec – aj vo výučbe príslušných technológií. Dopadom tejto nedbanlivosti môže byť práve Internet ovládaný ovládnutými toastermi.

V tomto príspevku vám predstavím, ako sa tejto hrozbe snažíme predísť vo výučbe predmetu IoT na TUKE, a aký obrovský popularizačný potenciál táto téma má. a ako bonus si môžete sami vyskúšať zabezpečenie jedného z našich zariadení.

## KDYŽ ŘEKA TEČE DO KOPCE: ZKUŠENOST ČESKÉ KOMUNITY OPENSTREETMAP S OTEVŘENÝMI STÁTNÍMI DATY

KAŠÍK Daniel (CZ)

V roce 2023 Český úřad zeměměřický a katastrální (ČÚZK) otevřel své datové sady, což umožnilo české komunitě OpenStreetMap získat oficiální souhlas k jejich využití pro mapování. Zejména vysoce kvalitní ortofoto výrazně usnadnilo a zpřesnilo proces tvorby mapy.

Otevření těchto dat však zároveň opět dostalo do popředí otázky spojené s kvalitou dat v OpenStreetMap. S využíváním detailních podkladů se ukázalo, že v datech stále přetrvávají různé typy chyb – například řeky tečou do kopce, landuse není správně zmapován, chybí budovy, silnice a lesní cesty jsou o několik metrů posunuty vůči skutečnosti. Část problémů lze řešit pomocí nástrojů dostupných v QGIS, ať už je to detekce a eliminace sliverů (mezery mezi polygony) či vyhodnocování polohové přesnosti prvků. Větším problémem jsou však zcela chybějící data v OpenStreetMap, a to zejména na venkově, kde se nacházejí nejhůře zmapované oblasti. Navíc, srovnatelná datová sada, která by sloužila jako referenční dataset, často neexistuje. Nezbyvá nic jiného než si referenční dataset vytvořit vlastní, např. za použití machine learningu.

V přednášce si ukážeme, jak je možné pomocí objektové detekce hodnotit kvalitu dat, jak vyhledávat nejméně zmapovaná místa v České republice a jak si při mapování uchovat zdravý rozum. v závěru budou prezentovány aktuální i plánované projekty české komunity OpenStreetMap.



## MARIMO NOTEBOOK: DÁTA O NEZAMESTNANOSTI V KRAJINÁCH V4

KAUKIČ Michal (SK)

Poukážeme na široké možnosti otvoreného softvéru, ktoré umožňujú spracovanie a vizualizáciu tých istých dát s použitím rôznych nástrojov. V našom prípade využijeme datasety Michala Páleníka (<https://doi.org/10.5281/zenodo.6165135>) o nezamestnanosti a vývoji populácie v krajinách V4. Budeme pracovať v marimo notebooku (dokumentácia je na <https://docs.marimo.io/>).

Porovnáme možnosti marimo a Jupyter notebookov, dôvody, prečo sme sa rozhodli pre marimo. Ukážeme, ako môžeme vyvíjať a zverejniť webovú aplikáciu s názornou vizualizáciou týchto dát. Využijeme tieto Python moduly:

- polars a geopandas pre spracovanie geografických a štatistických údajov ako dátových rámcov (DataFrames),
- plotly pre kreslenie máp a čiarových grafov s údajmi o nezamestnanosti a vývoji populácie.

Aplikácia je (a bude) prístupná na <https://unemp.feelmath.eu/>. Zdrojové kódy a použité datasety sa dajú nájsť na <https://github.com/misolietavec/OSSConf25>.

## AKO UROBIŤ BATÉRIOVÉ ÚLOŽISKÁ PROFITABILNÉ: Z ČOHO BOLÍ HLAVA AI INŽINIEROV V ENERGETIKE (INO-HUB Energy)

STRAKA Milan (SK)

S rastúcim podielom obnoviteľných zdrojov energie, ako je fotovoltika či veterné elektrárne, čelí energetická sieť výzve v podobe kolísavej a ťažko predvídateľnej výroby elektriny. Batériové úložiská ponúkajú riešenie – dokážu stabilizovať dodávku energie, zvyšovať jej využiteľnosť a zároveň poskytovať služby pre sieť. Ich nasadenie je však finančne náročné, a preto je nevyhnutné optimalizovať ich prevádzku s ohľadom na ekonomickú návratnosť.

V tomto príspevku predstavujeme praktickú aplikáciu, ktorá kombinuje batériové úložisko s fotovoltickým zdrojom a zameriava sa na znižovanie nákladov prostredníctvom protiodchýlkovania – teda minimalizácie rozdielov medzi plánovanou a skutočnou dodávkou energie. Ide o problém, ktorý spája predikciu, optimalizáciu a rozhodovanie v reálnom čase.

Riešenie je postavené na open source nástrojoch – najmä na Pyomo, modelovacím jazyku pre optimalizačné úlohy v jazyku Python. Ukazujeme, ako možno formulovať optimalizačný model, ako ho integrovať do systémov riadenia energetických zdrojov a ako zvládnuť praktické aspekty jeho nasadenia a údržby.

Príspevok cieli na odborníkov aj širšiu komunitu záujemcov o open source a dátovo riadené rozhodovanie – poukazuje na to, že aj v oblasti energetiky existujú zaujímavé a aktuálne výzvy, ktoré sú technicky prístupné, spoločensky významné a otvorené pre vývoj v open source ekosystéme.

## POČET OBARVENÍ N BARVAMI (část 2)

STRÍŽ Pavel (CZ)

Příspěvek počítá počet  $n$ -obarvení v daném grafu, chceme-li mapě krajů či okresů. Vedle běžných 4 barev užitých v teorému čtyř barev jsem dopočítal  $n$ -obarvení pro přípustné barvy. Dvacetidvouuzlový graf byl stále ještě výpočetně přijatelný, tedy pro cca 200 jader a několik dní počítání.

## VYBARVENÍ MAPY JAKO HRA (část 3)

STRÍŽ Pavel (CZ)

Příspěvek představuje variantu obarvovací hry v mapě/grafu. Barvy reprezentují cifry a ty se sčítají na hranicích a společných bodech. Idea pochází z teorému čtyř barev a generování zadání je principiálně stejné jako u sudoku. V pozadí se nejvíc užívá Python a Picat. V článku čtenář najde ukázky takové hry, ta není pro osm a víc barev tak jednoduchá, jak by se mohlo zdát. A hra by byla ještě těžší, kdyby se užila fiktivní mapa s průsečíky s mnoha oblastmi. Většina reálných map má hranici složenou jen ze dvou či tří oblastí. Je to výzva to zkusit.

## HLEDÁNÍ OPTICKÉHO STŘEDU HRANIC (část 4)

STRÍŽ Pavel (CZ)

Příspěvek prochází možností, jak zobrazit optický střed na hranici dvou oblastí, např. států, krajů či okresů. Zvolená možnost je naprogramovaná v Pythonu a výstup ukázán na reálné mapě České a Slovenské republiky. Užití je v obarvovací hře, která se diskutuje v 3. části o mapách.

## ROHOVÉ RAZÍTKO V T<sub>E</sub>XU (bonusová část)

STRÍŽ Pavel (CZ)

Příspěvek naznačuje, jak by se dalo vysázet rohové razítko v T<sub>E</sub>Xu. Úsvit TikZu.

## AI OČAMI ŠTUDENTOV ALEBO JE AI IBA ML A LLM?

RAKOVSKÁ Eva (SK)

Fenomén doby ChatGPT ovplyvňuje aj učenie, aj výkon a schopnosti študentov. V rámci APVV projektu BICISEDU sme realizovali spoločne s Bifröst University na Islande prieskum ako vnímajú študenti umelú inteligenciu, zisťovali sme, či majú spoločné pohľady slovenskí, islandskí a neskôr aj srbskí študenti. Ukázali sa nezrovnalosti vo

vnímaní AI, ale aj v praktickom využívaní AI. V rámci pedagogickej praxi učím umelú inteligenciu a ukazuje sa, že s rozvojom strojového učenia (ML) študenti nevnímajú ostatné paradigmy AI a nedokážu rozoznávať ani základné atribúty AI.

Korektné a etické využívanie AI vo výučbe sa stáva veľmi zložitou otázkou a zisťovanie nekorektného využívania veľkých jazykových modelov (LLM) v písomných prácach, odpovediach a záverečných prácach je pre učiteľov veľmi náročné. V spolupráci s FAJ EUBA a Trnavskou univerzitou sme sa pokúsili v roku 2024 urobiť detekciu využívania LLMs v písomných prácach študentov z angličtiny, kde sme získali prekvapivé zistenia. V príspevku sa snažím zhrnúť poznania ako rozvoj LLMs a ML ovplyvňuje myslenie študentov a na čo sa zamerať pri výučbe AI a pri osвете aj v študijných programoch, ktoré zdanlivo nemajú nič spoločné s AI.

## ANALÝZA A ŠTATISTIKA MIEN A PRIEZVISK Z OTVORENÝCH ÚDAJOV SLOVENSKÝCH MIEST

TOTH Štefan (SK)

Aké sú najčastejšie mužské a ženské mená v Žiline a v iných slovenských mestách? Ktoré mená a priezviská patria medzi najkratšie a najdlhšie? Ako často sa vyskytujú v jednotlivých lokalitách vzhľadom na počet obyvateľov?

Prednáška predstaví spôsob získavania a spracovania takýchto údajov z otvorených databáz miest a obcí, ako aj návrh webovej aplikácie umožňujúcej ich vizualizáciu a analýzu.

Mená a priezviská nie sú len identifikačné znaky, ale zároveň odrážajú migračné vzorce, kultúrne tradície a jazykové variácie. Ich štatistické spracovanie patrí do oblasti onomastiky, ktorá skúma výskyt mien v rôznych regiónoch a ich vzťah k demografickým trendom. V praxi je však dostupnosť týchto údajov obmedzená – štatistiky zverejňuje len časť samospráv, zatiaľ čo mnohé mestá takéto údaje vôbec neposkytujú.

Výsledkom práce je aplikácia, ktorá ponúka prehľadné a interaktívne štatistiky mien a priezvisk obyvateľov slovenských miest, vrátane ich geografického zobrazenia na mape s využitím open-source wrappera nad knižnicou Leaflet pre Blazor.

## **PICO NA CITRÓNE** **workshop**

**BIŇAS Miroslav (SK)**

Alebo kyslo-sladký úvod do programovania mikrokontrolérov v jazyku MicroPython.

Chceš vedieť, ako z miniatúrnej dosky vyžmýkať maximum? Príď na workshop, na ktorom ti ukážeme, že aj s malým mikrokontrolérom Raspberry Pi Pico WH sa dajú robiť veľké veci! Najprv cez protokol HTTP stiahneme informácie o aktuálnom počasi (lebo kto chce pracovať v zime alebo daždi, že?). Potom ich upravíme a odošleme cez protokol MQTT do služby Adafruit IO, kde ich budeme ukladať a vizualizovať. Nebude to žiadna nuda, len čistý digitálny zážitok!

Workshop skúma možnosti a potenciál Raspberry Pi Pico 101 v jazyku MicroPython.

## **AKÝ JE TO POKÉMON?** **workshop**

**BIŇAS Miroslav (SK)**

Alebo vytvor si svoj vlastný Pokédex v jazyku Python ako webovú aplikáciu.

Pokédex si vytvoríme pomocou populárneho mikro-webového rámca FastAPI. Pokémonov získame z SQLite databázy pomocou balíka SQLAlchemy. Pomocou šablónovacieho systému Jinja vytvoríme jednoduchú dynamickú webovú stránku, ale rovnako aj REST API pre poskytovanie údajov o Pokémonoch ďalším aplikáciám. A čo nestihneme spolu si môžete podľa návodu dorobiť doma.

Počas workshopu sa vytvorí nevyhnutná pomôcka pre každého zberateľa Pokémonov.

## **KREATÍVNE PROGRAMOVANIE** **S p5.js (... a ChatGPT)** **workshop**

**TKÁČOVÁ Zuzana (SK)**

Workshop pre učiteľov, ktorí chcú rozvíjať kreatívne myslenie žiakov prostredníctvom live coding-u.

Objavte silu kreatívneho kódovania, ktoré prepája umenie, dizajn a programovanie. V tomto workshope sa naučíte pracovať s open knižnicou p5.js – prístupným a vizuálne atraktívnym nástrojom pre tvorbu interaktívnych vizualizácií. Spoločne preskúmame, ako môže ChatGPT podporiť proces tvorby – od generovania nápadov cez ladenie kódu až po návrhy na zlepšenie. Získate praktické tipy, ako creative coding zapojiť do vyučovania informatiky aj medzipredmetových projektov. Odídete s vlastným mini projektom a jasnou predstavou, ako túto inovatívnu formu programovania priniesť do triedy.

Vydanie tohto zborníka bolo podporené grantom slovenskej kultúrno-edukačnej agentúry KEGA 019ŽU-4/2023 „Inovatívne učenie matematiky s podporou Open Source“.

**OTVORENÝ SOFTVÉR VO VZDELÁVANÍ, VÝSKUME A V IT RIEŠENIACH**  
Zborník príspevkov medzinárodnej konferencie OSSConf 2025,  
konanej 1. júla – 3. júla 2025 v Žiline

Prvé vydanie 2025  
Elektronická sadzba programom pdfL<sup>A</sup>T<sub>E</sub>X Rudolf Blaško  
Vydala Žilinská univerzita v Žiline  
Tlač EDIS – vydavateľstvo UNIZA  
Náklad 100 ks  
Rozsah 180 strán

ISBN 978-80-554-2209-1

ISBN 978-80-554-2209-1

