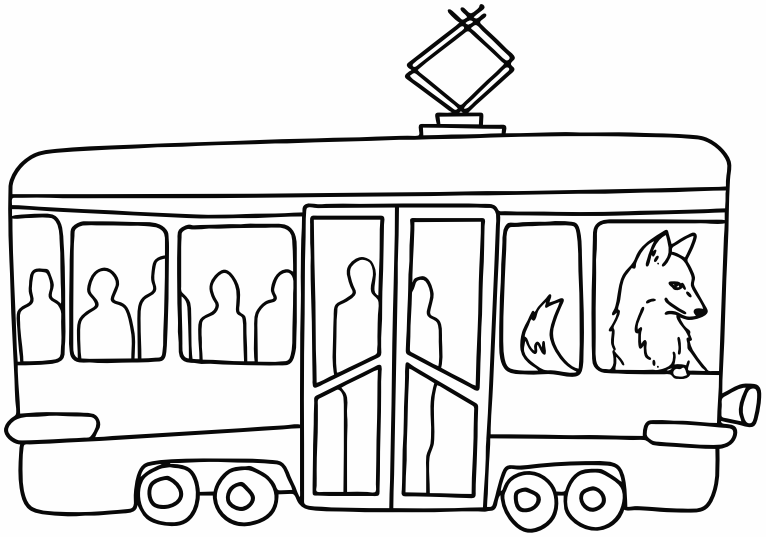
STUDENTSKÝ ČASOPIS A KORESPONDENČNÍ SEMINÁŘ

Ročník XXV
Číslo 6

M&M

MATEMATIKA FYZIKA INFORMATIKA



Uvnitř najdete několik témat a s nimi souvisejících úloh. Zamyslete se nad nimi a pošlete nám svá řešení. My vám je opravíme, pošleme zpět s dalším číslem a ta nejzajímavější z nich otiskneme. Nejlepší řešitele zveme na podzim a na jaře na soustředění.

Milí řešitelé,

v rukou držíte poslední číslo 25. ročníku. Kromě vzorových řešení všech témátek si v něm můžete přečíst článek od Mgr.^{MM} Martina Zimena o Lichtenbergových figurách nebo příspěvek k tramvajovému tématku od Mgr.^{MM} Tomáše Flídra. Na posledních stránkách naleznete výsledkovou listinu 25. ročníku. Již zde vám prozradíme, že nejvíce bodů získal Mgr.^{MM} Tomáš Sourada, kterému tímto gratulujeme. Zároveň máme tu čest udělit lahodný dort Mgr.^{MM} Martinu Zimenovi za nejlepší příspěvek tohoto ročníku. Děkujeme vám za vaši řešitelskou přízeň a těšíme se opět brzy na shledanou na stránkách našeho časopisu. Příjemné počtení a hezký začátek školního roku vám přejí

Vaši organizátoři

Řešení témat

Téma 1 – Paradoxní výsledky

Úvodem bychom chtěli poděkovat Viktoru Maternovi, který nám poslal velmi propracované řešení[1]. Těší nás, že do nich přidal vlastní nápady, počítal s netriviálním množstvím faktorů a celé téma velmi podrobně zpracoval.

Celé toto témátko pojednává o jednom objevu, který v roce 1963 učinil tehdy třináctiletý kluk z Tanzánie, Erasto Bartoloměj Mpemba. Od něj se odvíjí název celého tohoto jevu – Mpembův jev. Pokud se o něm chcete dozvědět další podrobnosti, doporučuji přečíst si diplomovou práci Pavla Böhma *Mpembův jev – skutečnost nebo fikce? Vliv historie na mrznutí vody*[2]. Je tam skvěle popsáno, co se děje, když si pohrajete s různými vlastnostmi ať už vody, nebo okolního prostředí.

My se jako první podíváme na to, jak se chová voda, když v ní budeme rozpouštět nějaké množství soli. Musíme si dát pozor, abychom nepřekročili rozpustnost soli, která je závislá na teplotě. Maximální množství soli, která se rozpustí ve vodě o teplotě blízké nule, je 35,63 g na 100 ml vody při 0 °C a toto množství nesmíme překročit[3].

Množství soli se projeví na teplotě tuhnutí solného roztoku. Proto si musíme stanovit nějaký moment, kdy ukončíme měření, i když roztok nezmrzne. Jako takovýto bod jsme si určili 0 °C. Nás zajímá převážně průběh mrznutí, povahu konečného stavu zmrznuté vody se solí známe a víme, co očekávat: slaný led.

V tabulce 1 vidíte v prvním sloupečku množství rozpouštěné soli na 100 g vody a ve druhém dobu, kterou trvalo roztoku dosáhnout 0 °C z počáteční teploty 75 °C. Čas tuhnutí není nijak výrazně odlišný.

[1] Text jeho příspěvku si můžete přečíst na `https://mam.mff.cuni.cz/media/tema/25/t1_Viktor_Materna.pdf`

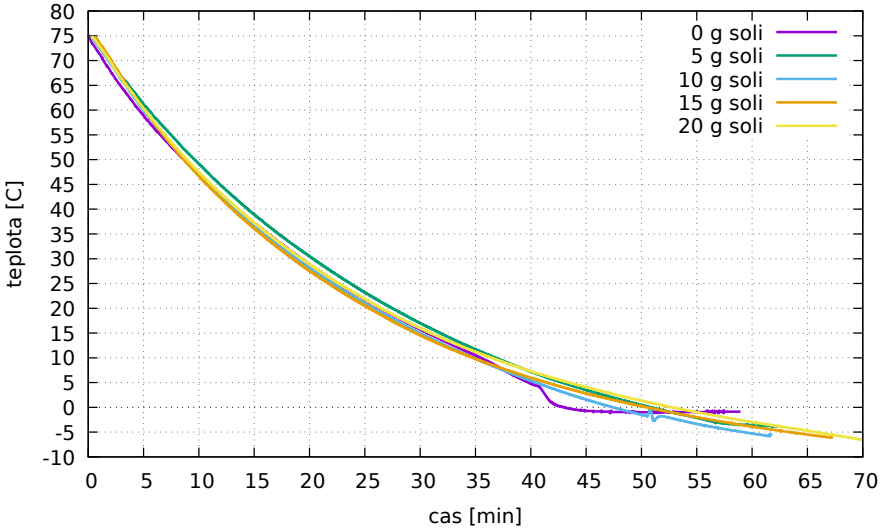
[2] Najdete ji na `http://mpemba.pavelbohm.cz/diplomka.pdf`

[3] viz `https://cs.wikipedia.org/wiki/Chlorid_sodný`

Množství soli	Doba do překročení 0 °C
0 g	43,1 min
5 g	50,3 min
10 g	47,2 min
15 g	50,0 min
20 g	52,6 min

Tabulka 1: Závislost doby tuhnutí 100 g vody o počáteční teplotě 75 °C. Koncentrace je uvedena jako množství soli na 100 g vody.

Průběh tuhnutí je zobrazen na obrázku 1. Vidíme, že po přidání soli do vody nedochází k rychlejšímu poklesu teploty kolem 5 °C. Proč? Jedním z možných vysvětlení je posun hodnoty teploty, kdy voda dosahuje maximální hustoty. V grafech bez soli v předchozím vzorovém řešení můžeme pozorovat takováto zalomení ve většině případů lehce pod 5 °C (anomálie vody se v čisté vodě pohybuje kolem 4 °C).



Obrázek 1: Průběh tuhnutí vody s různou relativní koncentrací soli

Pokud bychom měli navrhnout ideální kapalinu, na které se projevuje Mpembův efekt, tak potřebujeme, aby v kapalině lokálně vznikala místa, která se hodně liší svojí teplotou, čímž by docházelo k proudění. Teplota vody a její hustota spolu totiž souvisí. Voda se bude chovat podle Archimédova zákona a její část s menší hustotou (tedy s vyšší teplotou[4]) bude v objemu stoupat nahoru a naopak. Proto

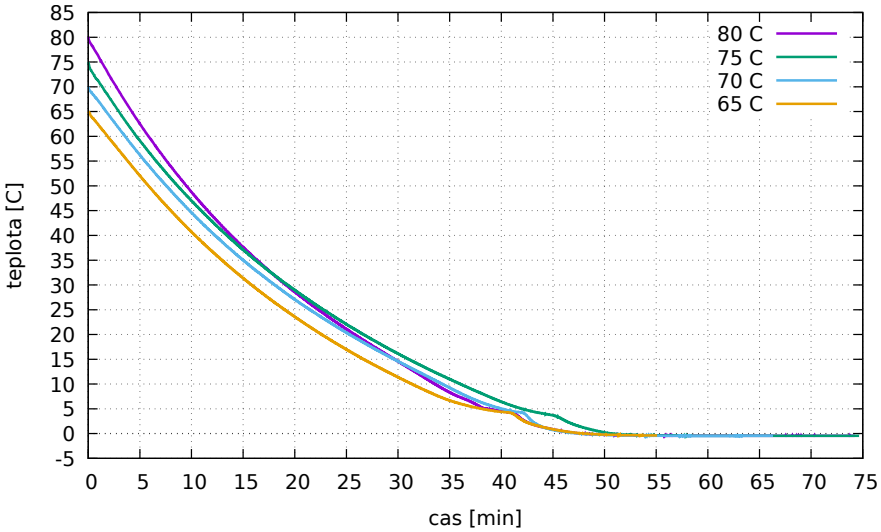
[4]Nesmíme zapomenout na anomální chování vody kolem 4°C.

potřebujeme kapalinu s malou tepelnou vodivostí, aby se rozdíly teplot ve dvou místech kapaliny nevyrovnávaly vedením a tím vznikaly proudy. Hustota tedy bude mít velmi rozdílné hodnoty v různých místech objemu a Archimédův zákon se bude projevovat výrazněji (oproti kapalině s vyšší tepelnou vodivostí). Tím pádem se u teplot bližších k nule jedná o opačný případ: teplejší voda klesá ke dnu nádoby a je chlazena od poličky mrazáku, kterou vede chladící systém. Odběr energie z vody je takto účinnější, což může mít za následek rychlejší pokles teploty v grafu pod hodnotou 5 °C.

Dalším způsobem odvodu tepla může být kromě chlazení přítomnost jiného tělesa v kapalině. Jaké vlastnosti by mělo toto těleso mít? Potřebujeme, aby odvádělo teplo z vody do okolí, tudíž by mělo být dobře tepelně vodivé a mělo by být v kontaktu s vodou a okolím co největší plochou. Například kovová lžička splňuje naše požadavky celkem obstojně. Data, která jsme naměřili po vložení lžičky do 100 g vody s různou počáteční teplotou, můžete vidět v tabulce 2 a grafu 2.

Počáteční teplota vody	Doba do překročení 0 °C
65 °C	48,3 min
70 °C	74,8 min
75 °C	51,0 min
80 °C	48,1 min

Tabulka 2: Závislost doby tuhnutí 100 g vody s vloženou lžičkou jako vodičem tepla



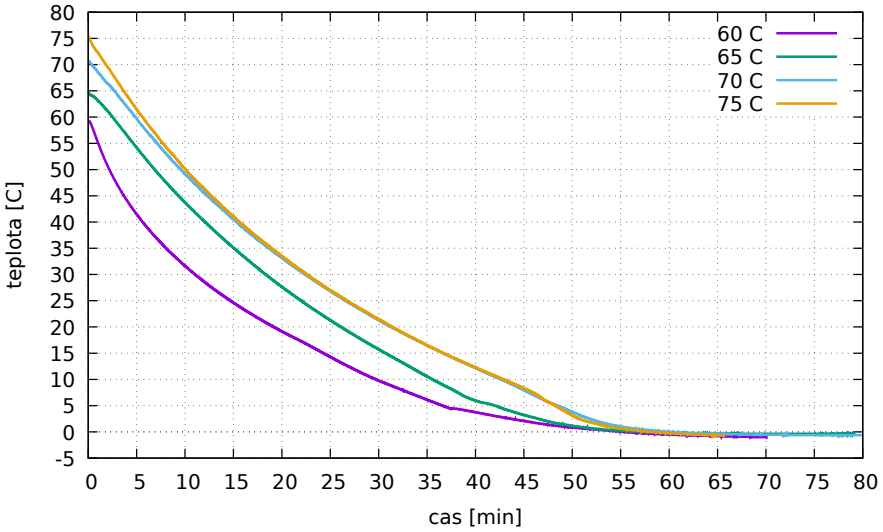
Obrázek 2: Průběh tuhnutí vody s vloženou lžičkou jako vodičem tepla

Ložička je z vodivého materiálu, vyrovnávala tedy teplotní rozdíly v různých místech objemu vody, což vedlo k rovnoměrnějšímu rozložení tepla v kapalině.

Na zamezení proudění v kapalině (a tím i zamezení rychlému vyrovnávání teplotního rozdílu a pomalejšímu chladnutí) potřebujeme něco nevodivého, co rozdělí objem vody na části, aby se voda nemíchala. V tomto experimentu byla použita papírová dvojitá spirála svisle vložená do skleněné kádinky s vodou. Výsledky měření zobrazují tabulka 3 a graf 3.

Počáteční teplota vody	Doba do překročení 0 °C
60 °C	55,4 min
65 °C	57,9 min
70 °C	60,4 min
75 °C	58,2 min

Tabulka 3: Doba tuhnutí 100 g vody s bariérou vloženou v jejím objemu



Obrázek 3: Průběh tuhnutí vody s vloženou papírovou spirálou

Při porovnání doby tuhnutí z předchozího vzorového řešení (viz tabulka dat 4) vidíme, že bariéra zpomalila tuhnutí. Důvodem je to, že bariéra zabraňuje vytváření proudů v celém objektu kapaliny.

Svojí povahou a vlastnostmi ovlivňuje průběh tuhnutí také podloží. V grafu 4 a tabulce 5 můžete vidět naměřená data. Pokud je podložím led, je průběh teploty vody naprosto odlišný od případů, kdy je jako podloží zvoleno dřevo. Dřevo má malou tepelnou vodivost, tudíž nezvyšuje tok tepla z nádoby ven, ale ovliv-

Počáteční teplota vody	Čas do překročení 0 °C	Odpar vody Δm
82,6 °C	39,28 min	7 %
69,9 °C	37,00 min	6 %
65,3 °C	42,75 min	1 %
60,0 °C	40,28 min	2 %
41,9 °C	57,43 min	3 %
30,9 °C	54,50 min	2 %
25,0 °C	51,92 min	2 %
22,1 °C	65,05 min	1 %
21,5 °C	62,48 min	1 %

Tabulka 4: Data z experimentů pro závislost doby tuhnutí na počáteční teplotě kapaliny, kde τ_0 je počáteční teplota vody, t_z je čas překročení 0 °C a Δm je odpar vody, poslední sloupec je záznam vody v hmotnostních procentech

ňuje způsob distribuce tepla do okolí. Izolační vlastnosti dřeva ovlivňují interakci kapaliny s podložím přes stěny nádoby.

Materiál podloží	Počáteční teplota vody	Doba do překročení 0 °C
dřevo	80 °C	45,1 min
	75 °C	52,4 min
	70 °C	37,0 min[5]
led	80 °C	71,3 min
	75 °C	65,4 min
	70 °C	51,0 min

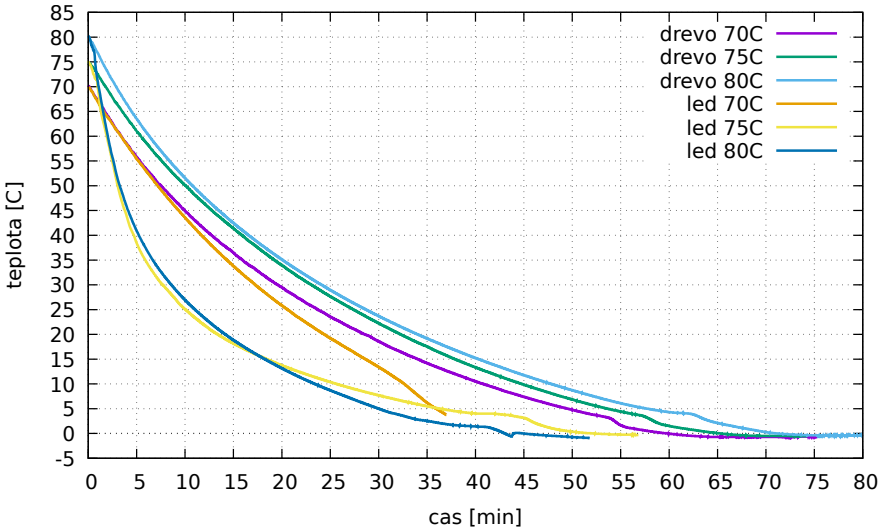
Tabulka 5: Doba tuhnutí 100 g vody na podložích z různých materiálů

Nádoba sama o sobě ovlivňuje chladnutí kapaliny. Její vliv na tuhnutí popisuje veličina zvaná tepelný tok a závisí na materiálu, rozměrech nádoby a rozdílu teplot vody a okolí, ve kterém je nádoba umístěna. Materiál nádoby charakterizuje součinitel teplotní vodivosti, který je pro každý materiál jiný. Jde jej dohledat v tabulkách či na internetu. Pokud máte válcovitou nádobu, obecně můžete vyjádřit tepelný tok pomocí tohoto vzorce:

$$Q = \frac{\lambda S \Delta\tau}{d} + \pi \frac{r_1 + r_2}{r_2 - r_1} \lambda L \Delta\tau$$

První sčítanec reprezentuje tepelný tok dnem a druhý reprezentuje tepelný tok stěnou nádoby, tedy válcem. Vzorce, které byly použity, najdete ve studijním textu Fyzikální olympiády: `http://fyzikalniolympiada.cz/texty/texttz.pdf`. Rozměry a vysvětlivky veličin jsou vypsány v tabulce 6 a po jejich dosazení vychází

[5] Za tento čas dosáhla voda teploty 3,75 °C



Obrázek 4: Průběh tuhnutí vody umístěné na různých podložích

číselně $Q = 283{,}97\Delta\tau + 6{,}97\Delta\tau$. První sčítanec je větší než druhý, větší vliv tedy má tepelný tok dnem nádoby.

Značka	Název	Hodnota
λ	součinitel tepelné vodivosti (sklo)	$0{,}6\,\mathrm{W\ m^{-1}\ K^{-1}}$
r_1	vnitřní poloměr nádoby	$5{,}48\,\mathrm{cm}$
r_2	vnější poloměr nádoby	$5{,}95\,\mathrm{cm}$
S	plocha dna nádoby odpovídající r_2	$\pi r_2^2 = 111{,}22\,\mathrm{cm}^2$
d	tloušťka stěny nádoby	$(r_2 - r_1)/2 = 0{,}24\,\mathrm{cm}$
τ_1	teplota okolí (mrazáku)	$-22\,^\circ\mathrm{C}$ (konstantní)
L	výška hladiny v nádobě	$3{,}77\,\mathrm{mm}$

Tabulka 6: Konstanty a rozměry nádoby

Musíme uvážit i vliv mrazáku. Ten ovlivňuje především dobu tuhnutí, což se odvíjí od jeho výkonu. Pokud máme na začátku 100 g vody, k jejímu ochlazení na 0 °C je třeba odebrat vodě energii o velikosti $Q = mc\Delta\tau$, kde c je měrná tepelná kapacita vody a $\Delta\tau$ je proměnná závisející na počáteční teplotě kapaliny a teplotě, při které jsme měření ukončili, resp. rozdíl teplot mezi okamžiky reprezentujícími rozdíl energie odebrané mrazákem. Pokud bychom chtěli do tepelné bilance mezi vodou a mrazákem uvažovat i přeměnu vody na led, je třeba odhadnout, kolik procent vody zmrzlo. Pokud zmrzlo k hmotnostních procent vody ($0 \leq k \leq 1$), tak teplo potřebné na přeměnu oné části vody v led lze spočítat $L_\mathrm{t} = l_\mathrm{t} mk$, kde

l_t je měrné skupenské teplo tání vody. Celková energie odebraná vodě mrazákem se tedy rovná $E = Q + L_t$ a za pomoci zjištěného výkonu mrazáku můžeme odhadnout čas, za který voda dosáhne 0 °C a k hmotnostních procent vody se přemění na led: $t = E/P = kmc\Delta\tau/P$. Pokud považujeme $k = 1$, tak se pro konkrétní podmínky experimentu je čas teoreticky lineárně závislý na teplotě. Zamyslete se, zda předchozí tvrzení platí či ne.

Pája a Matej; pavla.trembulakova@seznam.cz

e-mailová konference: paradoxni@mam.mff.cuni.cz

Téma 2 – Principy kryptografie

Řešení

V průběhu celého roku byla v rámci tématu zadána celá řada úloh, z nichž některé se zatím nedočkaly vzorového řešení. To se v tomto čísle změní.

Zároveň nás těší, že pro některé řešitele bylo toto téma motivací se hlouběji ponořit do programování. Proto jsme připravili interaktivní možnost si s programátorskými úlohami z tématu trochu pohrát.

Úlohy z prvního a z třetího čísla, které se týkají šifry OCTA, si lze prakticky ozkoušet ve sdíleném dokumentu `https://1url.cz/@octa`. S úlohami z druhého čísla týkajícími se hashovací funkce RIKSHA pomůže sdílený dokument `https://1url.cz/@riksha`.

Problémy zadané ve 2. čísle

Úloha 4

Zadání:

Dokážete najít dvě vstupní hodnoty funkce RIKSHA, pro které je výsledný hash stejný?

Řešení:

Tato úloha vyžaduje pochopit, co se vlastně ve funkci RIKSHA při výpočtu děje. Na rozdíl od běžných hashovacích funkcí můžeme v jejím výpočtu vidět místo rotací mnoho shiftů (vlevo i vpravo). Z toho důvodu některé vstupní bity nemají na výstup vůbec žádný vliv. K shiftům dochází jak při výpočtu pomocných proměnných a_0 až a_4, tak při úpravě vnitřního stavu funkce.

Funkce vždy pracuje s 8 byty vstupního textu. Označme je zleva 0 až 7. Ve skutečnosti je využívána dokonce jen malá část vstupního textu:

Pozice bytu	Počet využitých bitů
0	6
1	1
2	0
3	1
4	1
5	1
6	0
7	6

Vidíme tedy, že například třetí znak zleva v řetězci, ze kterého počítáme hash, nemá na výsledek žádný vliv. A skutečně: Hash slov „Ahoj“ i „Ahej“ je stejný a to 8D754901F101F15D.

Pro lepší představu, jak funkce pracuje, doporučujeme využít sdílený dokument odkazovaný výše a do funkce RIKSHA si přidat průběžné výpisy stavu.

Úloha 5

Zadání:
Víte, že pomocí funkce RIKSHA byl zahashován text, který není delší než 8 bytů. Výsledný hash je `02365E1E061E62BA`*. Dokážete najít nějaký (ne nutně stejný) vstup, který má stejný hash?*

Řešení:
Tato úloha navazuje na předchozí. Předně si můžeme všimnout, že pokud využijeme pouze jeden průchod updatu hashovací funkce, což nastane pro řetězce dlouhé maximálně 7 bytů, bude hash vždy začínat na 8D. Pomocná proměnná a_0 totiž vždy končí třemi nulami, takže pokud na ni při úpravě h_0 použijeme shift vlevo o 5, vždy přičítáme 0.

Hledaný text tedy bude dlouhý právě 8 bytů a doplněný 8 byty (známého) paddingu. Z každých 8 bytů se ve funkci ale podle tabulky u předchozí úlohy využije pouze 16 bitů. To znamená, že stačí vyzkoušet $2^{16} = 65536$ možností, což hrubou silou není problém.

Původním zahashovaným řetězcem bylo slovo „KraLovNa“. Pokud při hledání hrubou silou necháme na všech nepodstatných pozicích nuly, tak ho nedostaneme. Ale dostaneme jiný řetězec se stejným hashem.

Úloha 6

Zadání:
Existuje nějaká hodnota hashe, kterou funkce RIKSHA nevrátí pro žádný vstup? Nezapomeňte své tvrzení důkladně zdůvodnit.

Řešení:
Předpokládejme, že máme na mysli hash s délkou 8 bytů (to sice zadání explicitně nespecifikuje, ale naštěstí se nenašel nikdo, kdo by to chápal jinak).

Tato úloha je mnohem obtížnější než úlohy předchozí. Vstup do hashovací funkce můžeme prodlužovat a tím postupně ovlivnit všechny bity výstupu. Intuitivně se tedy zdá, že hashovací funkce může vrátit každou hodnotu. Precizní důkaz by ale pravděpodobně vyžadoval rozebrat velké množství možností, a proto ho neuvádíme.

Problémy zadané ve 3. čísle

První dvě úlohy jsou vysloveně programátorské. Všechny zde uveřejněné zdrojové kódy jsou v jazyku Python 3 a je možné je on-line spustit ve sdíleném dokumentu odkazovaném v úvodu tématu. V tomto dokumentu jsou i další kódy, které nejsou pro správné řešení nezbytné (např. pro dešifrování), ale mohou být užitečné pro pochopení, jak fungují módy blokových šifer.

Problém 1

Zadání:
Vyberte si libovolný bitmapový obrázek (například formát PPM) a zašifrujte jeho obrazová data (tedy vše kromě hlaviček) pomocí šifry OCTA s klíčem MANDM (t.j. 4D 41 4E 44 4D) v módu ECB. Měl by opět vzniknout zobrazitelný obrázek. Jak se od toho původního liší? Původní obrázek i jeho zašifrovanou podobu nám zašlete.

Pokud si nechcete hrát s obrázkovými formáty, tak si vzorový obrázek včetně popisu, jakou oblast je záhodno zašifrovat, můžete stáhnout z našeho webu ze stránky věnované tématku[6].

Řešení:
Pokud to ještě nemáme hotové z prvního čísla, musíme si nejdříve napsat šifrování pomocí funkce OCTA. To je dle popisu v prvním čísle poměrně přímočaré:

```
def octa_encrypt(block, key):
    ''' Zasifruj blok dat pomoci OCTA.
        Ocekava parametry
            block: plaintext, cislo odpovidajici ASCII kodu znaku
            key: klic, retezec ASCII znaku
        Vraci ciphertext, cislo odpovidajici ASCII kodu znaku.
    '''
    def substitution(state):
        sbox = (2, 0, 1, 3)
        new_state = 0
        for i in range(4):
            new_state += sbox[state >> 2*i & 0x03] * 4**i
        return new_state
    def permutation(state):
```

[6] https://mam.mff.cuni.cz/problem/2203

```
        return (state >> 3 | state << 5) & 0xff
    def add_key(state, round_key):
        return state ^ round_key
    state = block
    for r in range(5):
        state = add_key(state, ord(key[r % len(key)]))
        state = substitution(state)
        state = permutation(state)
    state = add_key(state, ord(key[5 % len(key)]))
    return state
```

Nyní stačí tuto funkci využít pro jednotlivé byty obrázku. Jde tedy jen o to se s problémem vypořádat technicky:

```
from struct import pack, unpack

def encrypt_file_ecb(input_file, output_file, key, header_lenght):
  ''' Zasifruj input_file pomoci OCTA v ECB modu s klicem key
      a vysledek zapis do output_file. Prvnich header_leght bytu
      opis bez sifrovani.
      Parametry
          input_file: cesta ke vstupnimu souboru
          output_file: cesta k vystupnimu souboru (bude prepsan)
          key: klic jako retezec ASCII znaku
          header_lenght: pocet bytu, ktere se maji preskocit
  '''
  with open(input_file,'rb') as inf, open(output_file,'wb') as ouf:
      # prepis hlavicku
      ouf.write(inf.read(header_lenght))
      # cti byty a sifruj
      pt_byte = inf.read(1)
      while pt_byte:
          ct_int = octa_encrypt(unpack('B', pt_byte)[0], key)
          ouf.write(pack('B', ct_int))
          pt_byte = inf.read(1)

encrypt_file_ecb('riki.ppm', 'riki_ecb.ppm', 'MANDM', 15)
```

Pro úplnost dodejme, že číst soubor po jednotlivých bytech určitě není nejrychlejší metoda. Cílem bylo především vytvořit co nejsrozumitelnější kód.

Pro šifrování můžeme využít obrázek Rikiho umístěný na našem webu, viz Obrázek 5.

Výsledek po zašifrování zachycuje Obrázek 6.

Můžeme vidět, že byty jsou šifrovány nezávisle, a tedy všechny pixely, které mají na původním obrázku stejnou barvu, mají stejnou barvu i na zašifrovaném

Obrázek 5: Původní obrázek

obrázku. I v zašifrované zprávě tak zůstala řada informací – například, které byty jsou stejné.

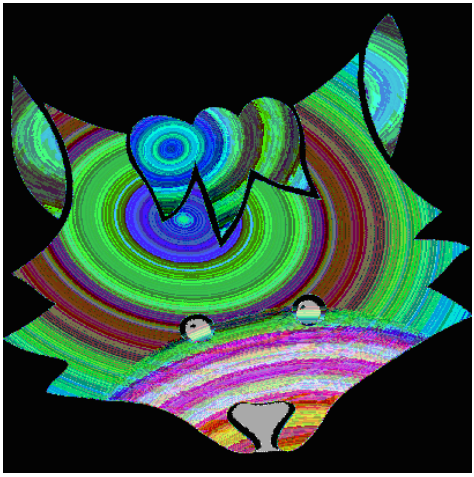
Problém 2

Zadání:
Zašifrujete obrázek z Problému 1 také pomocí šifry OCTA se stejným klíčem v CBC módu. Jaký je výsledek tentokrát? Opět nám oba obrázky pošlete.

Řešení:
K problému přistoupíme podobně jako u ECB módu. Jen při šifrování musíme ještě využívat i předchozí zašifrovaný blok.

```
from struct import pack, unpack
import random

def encrypt_file_cbc(input_file, output_file, key, header_lenght):
  ''' Zasifruj input_file pomoci OCTA v CBC modu s klicem key
      a vysledek zapis do output_file. Prvnich header_leght bytu
      opis bez sifrovani.
      Parametry
          input_file: cesta ke vstupnimu souboru
          output_file: cesta k vystupnimu souboru (bude prepsan)
          key: klic jako retezec ASCII znaku
          header_lenght: pocet bytu, ktere se maji preskocit
  '''
  with open(input_file,'rb') as inf, open(output_file,'wb') as ouf:
```



Obrázek 6: Obrázek zašifrovaný pomocí ECB módu

```
    # prepis hlavicku
    ouf.write(inf.read(header_lenght))
    # minuly ciphertext, na zacatku nahodny inicializacni vektor
    last_ct_int = random.randint(0,255)
    print("Inicializacni vektor: %s" % last_ct_int)
    # cti bloky a sifruj (pt = plaintext, ct = ciphertext)
    pt_byte = inf.read(1)
    while pt_byte:
        pt_int = unpack('B', pt_byte)[0]
        ct_int = octa_encrypt(pt_int ^ last_ct_int, key)
        last_ct_int = ct_int
        ouf.write(pack('B', ct_int))
        pt_byte = inf.read(1)

encrypt_file_cbc('riki.ppm', 'riki_cbc.ppm', 'MANDM', 15)
```

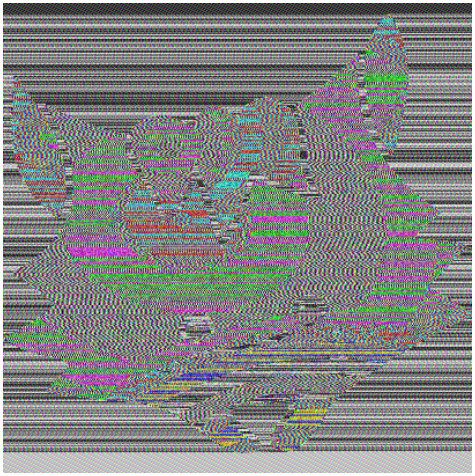
Výsledek si můžeme prohlédnout na Obrázku 7.

Při použití kvalitní šifry by z obrázku nemělo být vůbec zřejmé, co na něm je. Bohužel OCTA není (kvůli jiným úlohám) příliš dokonalá šifra. Pokud bychom ale šifrovali obrázek s proměnlivým pozadím, byl by výsledek mnohem přesvědčivější. Můžete to vyzkoušet.

Problém 3

Zadání:

Pokud zašifrujeme hodně dat pomocí šifry s malou velikostí bloku, musí se jednou stát, že dostaneme dva stejné bloky šifrového textu.

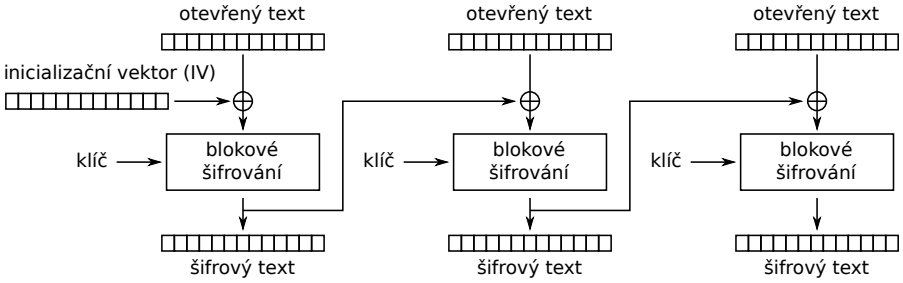


Obrázek 7: Obrázek zašifrovaný pomocí CBC módu

Plynou z toho v případě využití CBC módu pro šifrování nějaká bezpečnostní rizika?

Kolik textu musíme průměrně zašifrovat blokovou šifrou s bloky délky 8 (to je například OCTA), než dostaneme dva stejné bloky šifrového textu? Předpokládejte, že máte náhodný vstupní text a ideální blokovou šifru.

Řešení:
Otázka je tak návodná, že odpověď musí být ano. Na Obrázku 8 si můžeme připomenout, jak funguje CBC mód.



Obrázek 8: Šifrování pomocí CBC módu

Předpokládejme, že jsme našli dva stejné bloky šifrového textu $c_i = c_j$, kterým přísluší pro nás neznámé bloky otevřeného textu m_i a m_j. Protože blo-

ková šifra je permutace (zašifrovaný blok lze vždy jednoznačně dešifrovat na původní otevřený text), znamená to, že musí platit $m_i \oplus c_{i-1} = m_j \oplus c_{j-1}$, neboli $m_i \oplus m_j = c_{i-1} \oplus c_{j-1}$. Všechny bloky šifrového textu známe, dokážeme tedy spočítat XOR dvou bloků otevřeného textu. Pokud bychom dokázali odhadnout hodnotu jednoho z nich (to je občas možné například u hodnoty v hlavičce zašifrovaného souboru), mohli bychom spočítat hodnotu druhého. To určitě není žádoucí.

Tento útok v případě šifer s velikostí bloku 64 bitů je znám jako Sweet32.

Otázku, kolik dat potřebujeme na to, aby nastala kolize, jsme už řešili v první úloze druhého čísla. Z narozeninového paradoxu lze odvodit, že budeme potřebovat přibližně $2^{n/2}$ bloků, kde n je velikost bloku. Pokud máme velikost bloku 8 bitů, budeme tedy potřebovat průměrně asi $2^4 = 16$ bloků dat, což v tomto případě znamená zašifrovat 16 znaků. Pokud zašifrujeme znaků více, pravděpodobně nalezneme i více kolizí. Na 8 bitové šifry je tedy tento útok velmi reálný.

Problém 4

Zadání:
Nevýhodou CBC módu je, že k zašifrování bloku potřebujeme znát hodnotu předchozího bloku šifrového textu. Nemůžeme tedy výpočty provádět paralelně. Dokážete dohledat nebo vymyslet mód, který bude zachovávat výhody CBC módu a přitom bude umožňovat výpočty provádět (alespoň částečně) paralelně?

Řešení:
Tento problém je ryze praktický, a proto má řadu řešení využívaných v reálném světě. Asi nejrozšířenější a nejjednodušší možnost je využít tzv. *Counter (CTR)* mód.

V případě CTR módu si na začátku vygenerujeme náhodné číslo zvané *nonce*. První blok vytvoříme tak, že nonce zašifrujeme pomocí blokové šifry a k výsledku přixorujeme otevřený text. Pro každý další blok zvýšíme nonce o 1. Můžeme také říct, že k nonce vždy přičítáme *counter*, který pro každý blok zvýšíme o 1.

Vzhledem k tomu, že blok šifrového textu je závislý pouze na odpovídajícím bloku otevřeného textu a counteru, lze šifrovat i dešifrovat data paralelně. Dokonce je možné například dešifrovat pouze vybraný úsek dat.

Problémy zadané ve 4. čísle

Problém 1

Zadání:
Zdůvodněte, že vzorec pro výpočet Eulerovy funkce je správný.

Řešení:
Připomeňme, že Eulerova funkce $\varphi(n)$ počítá počet přirozených čísel menších nebo rovných n, která jsou s n nesoudělná.

U prvočísel je situace jednoduchá. Každé číslo menší než prvočíslo p je s p nesoudělné, tedy $\varphi(p) = p - 1$.

Pro mocniny prvočísel není výpočet o moc obtížnější. Pro p^k, kde p je prvočíslo, jsou nesoudělná právě ta čísla, která nejsou dělitelná p. Počet čísel od 1 do p^k, která jsou dělitelná p, je $p^k/p = p^{k-1}$. Proto $\varphi(p^k) = p^k - p^{k-1}$.

Abychom mohli důkaz dokončit, pro libovolné n, dokážeme multiplikativitu Eulerovy funkce: pro nesoudělná čísla a a b platí $\varphi(ab) = \varphi(a)\varphi(b)$. Důkaz tohoto tvrzení je o něco obtížnější[7].

Napišme si všechna čísla $0, 1, \ldots, ab-1$ do tabulky po řádcích zleva doprava.

0	1	2	...	$a-1$
a	$a+1$	$a+2$	...	$2a-1$
...	...	...	...	...
$a(b-1)$	$a(b-1)+1$	$a(b-1)+2$	...	$ab-1$

Koukněme se na číslo v řádku i a sloupci j, přičemž řádky a sloupce značíme od nuly. Pak je na tomto místě napsané číslo $ia+j$. Zajímá nás, zda je soudělné s ab. Jelikož jsou ale čísla a a b nesoudělná, tak stačí zjistit, jestli je $ia+j$ nesoudělné jak s a, tak s b. Aby bylo číslo nesoudělné s a, tak musí být $\mathrm{NSD}(ia+j, a) = 1$, tedy $\mathrm{NSD}(j, a) = 1$. To ale znamená, že čísla nesoudělná s ab mohou být jen ve sloupcích označených čísly, která jsou nesoudělná s a. Těchto sloupců je $\varphi(a)$.

Podívejme se na čísla v jednom z těchto sloupců. Jsou to čísla $j, a+j, 2a+j$, $3a+j, \ldots, (b-1)\,a+j$. Tato čísla dávají navzájem různé zbytky modulo b. Kdyby totiž dvě z těchto čísel dávala stejný zbytek modulo b, byl by jejich rozdíl nenulový a dělitelný b. Zároveň je jejich rozdíl jistě dělitelný a. Dostali bychom tedy spor s $\mathrm{NSD}(a,b) = 1$.

Čísla tedy dávají v nějakém pořadí zbytky $0, 1, \ldots, b-1$ modulo b. Právě $\varphi(b)$ z nich je nesoudělných s b, a tedy i s ab. V každém z uvažovaných $\varphi(a)$ sloupců máme $\varphi(b)$ čísel nesoudělných s ab, dohromady je tedy čísel nesoudělných s ab přesně $\varphi(a)\varphi(b)$, což jsme chtěli dokázat.

Díky multiplikativitě dostáváme pro $n = p_1^{\alpha_1} p_2^{\alpha_2} \ldots p_k^{\alpha_k}$ vztah

$$\varphi(n) = \varphi(p_1^{\alpha_1})\varphi(p_2^{\alpha_2}) \ldots \varphi(p_k^{\alpha_k}) = (p_1-1)p_1^{\alpha_1-1}(p_2-1)p_2^{\alpha_2-1} \ldots (p_k-1)p_k^{\alpha_k-1}.$$

Problém 2

Zadání:
Platí Eulerova věta i pro přirozená čísla a a n, jejichž největší společný dělitel je 2? Pokud ano, zdůvodněte. Pokud ne, najděte protipříklad.

Řešení:
Najít protipříklad není obtížné. Zvolme $a = 2$, $n = 4$. Poté $\varphi(n) = 2$, ale $a^{\varphi(n)} = 2^2 = 4$ určitě nedává zbytek 1 po dělení 4.

[7] Důkaz je s drobnými úpravami převzatý ze seriálu 33. ročníku Matematického korespondenčního semináře, `https://mks.mff.cuni.cz/archive/33/serial.pdf`. Tento seriál můžeme všem zájemcům o teorii čísel vřele doporučit.

Problém 3

Zadání:
Mějme RSA s veřejným klíčem (493, 33) a soukromým klíčem (493, 353). Přišla vám zašifrovaná zpráva 93. Jaká je hodnota původního otevřeného textu?

Řešení:
Stačí využít vzorec ze zadání tématu.

$$m \equiv c^d \pmod{n}, \qquad \text{neboli} \qquad m \equiv 93^{353} \pmod{493}$$

Pro výpočet je nejjednodušší využít počítač. Nepracujeme s tak velkými čísly, takže může dobře posloužit třeba i Python. Případně můžeme použít vhodnou webovou službu jako například Wolfram Alpha[8].

Získáme tak původní zprávu $m = 42$.

Problém 4

Zadání:
Jaké jsou zvyklosti pro volby velikosti modulu a veřejného a privátního exponentu u RSA? Můžete vycházet z veřejně dostupných doporučení. Bonusové body jsou za vlastní průzkum. Například téměř každý server umožňující šifrovaný přístup přes HTTPS využívá certifikát s RSA klíčem. Podívat se, jaké hodnoty jsou využívány v praxi, tedy není vůbec obtížné.

Řešení:
V současné době jsou pro RSA nejrozšířenější moduly dlouhé 2048 bitů, což je všeobecně považováno za minimální bezpečnou hodnotu. Výjimkou nejsou ani moduly o velikosti 3072 nebo 4096 bitů.

Webové certifikáty typicky využívají moduly s délkou 2048 bitů, delší moduly se využívají spíše třeba pro SSH, kde by narušení bezpečnosti mohlo mít často fatálnější důsledky.

Ačkoli není veřejně známa žádná faktorizace modulu délky 1024 bitů, je tato délka považována za nedostatečnou a například moderní prohlížeče certifikáty s takto krátkým klíčem nepovažují za důvěryhodné.

Při volbě veřejného exponentu je téměř výhradně využíváno číslo 65537, neboli $(10000000000000001)_2$ ve dvojkové soustavě. Pro příliš krátké nebo dlouhé veřejné exponenty jsou známy různé útoky. Tato hodnota je považována optimální z hlediska bezpečnosti i snadnosti výpočtu.

Problém 5

Zadání:
Protože problém faktorizace je sice obtížný, ale nikoli neřešitelný, doporučuje se RSA klíče po nějaké době (třeba po několika letech) vyměnit. To vyžaduje vygenerovat dvě nová velká prvočísla. To je výpočetně náročné, a tak jsme se rozhodli

[8] `https://www.wolframalpha.com/`

vždy generovat jenom jedno velké prvočíslo a druhé nechat z minulého klíče. Je takový přístup bezpečný?

Řešení:
Popsaný postup je jednoznačně špatně, ale byl kupodivu v praxi pozorován.

Zatímco faktorizace je výpočetně náročný problém, hledání největšího společného dělitele dvou čísel je problém řešitelný snadno pomocí Eukleidova algoritmu.

Pokud tedy útočník získá dva moduly (starý a nový), které obsahují právě jedno stejné prvočíslo, může toto prvočíslo snadno zjistit. Tím se mu podařilo faktorizovat starý i nový modul.

Problém 6

Zadání:
Představte si, že znáte dvě zašifrované zprávy c_1 a c_2, které vznikly zašifrováním neznámých textů m_1 a m_2. Dokážete z nich spočítat zašifrovanou zprávu, která odpovídá $m_1 \cdot m_2$ nebo $m_1^2 \cdot m_2^3$? Jak by tomu bylo možné zamezit?

Řešení:
Můžeme si rozmyslet, že RSA je multiplikativní. Označme c_s zašifrovanou zprávu, která odpovídá $m_1 \cdot m_2$. Potom platí

$$c_s = (m_1 \cdot m_2)^e = m_1^e \cdot m_2^e = c_1 \cdot c_2.$$

Tedy znalost c_1 a c_2 mi stačí k dopočítání c_s. Podobně můžeme postupovat pro $m_1^2 \cdot m_2^3$.

Řešením tohoto problému je zavedení hlaviček a paddingu pro všechny zprávy určené k šifrování. Regulérní dešifrovaná zpráva pak musí začínat i končit přesně popsanými sekvencemi bytů. Ty jsou navrženy tak, aby nešlo multiplikativity využít pro zkonstruování validní zašifrované zprávy.

Problémy zadané v 5. čísle

Problém 1

Zadání:
Představme si, že v elektronickém bankovnictví chceme podepsat příkaz k úhradě. Nejjednodušší přístup je vzít jednotlivé položky platby a naskládat je za sebe.

Výsledný řetězec by mohl mít například tvar UCET_ODESILATELE ‖ UCET_ADRESATA ‖ ZPRAVA_PRO_ADRESATA ‖ CASTKA ‖ KONSTANTNI_SYMBOL ‖ VARIABILNI_SYMBOL ‖ SPECIFICKY_SYMBOL, kde ‖ značí spojení řetězců.

Je takový přístup pro reálné situace dostatečný, nebo přináší nějaké riziko? Dokážete navrhnout, jak vytvořit řetězec lépe?

Řešení:
Nedostatek navrženého řešení spočívá v tom, že nejsou jasně oddělené položky hashe. Podpis pro platbu s údaji `ZPRAVA_PRO_ADRESATA` „Platim 100“ a `CASTKA`

„100“ tak je úplně stejný jako podpis platby s položkami `ZPRAVA_PRO_ADRESATA` „Platim “ a `CASTKA` „100100“. Pokud tedy uživatel v dobré víře odsouhlasí první variantu platby, může jeho podpis být zneužit i pro druhou variantu platby.

Možností řešení je celá řada. Za nejjednodušší považujeme pole oddělit pomocí nějakého znaku (obvykle nazývaného *oddělovač*), který se v žádném z polí nemůže vyskytnout. Alternativně by bylo možné všechna pole zarovnat na pevnou délku (každé může mít jinou, ale pevně stanovenou), ale takové řešení se v praxi příliš nepoužívá. Dalším řešením by bylo počítat hash z nějakého samopopisného formátu dat jako třeba XML nebo ASN.1.

Problém 2

Zadání:
Pokud chceme uzavírat elektronicky smlouvy, objevuje se nám ještě jeden problém: často je důležité prokázat, kdy byla smlouva uzavřena. Napadá vás nějaké řešení, jak do podpisu přidat nezpochybnitelnou informaci o času, kdy k němu došlo?

Řešení:
K tomuto účelu slouží tzv. *časová razítka*. V podstatě jde o popisy od uznávané certifikační autority, které obsahují informaci o čase, kdy k podpisu došlo. Časovým razítkem tedy lze doložit, že v daný čas dokument včetně podpisů existoval. Stejně jako všude u elektronického podpisu předpokládáme, že nedokážeme vytvořit dva dokumenty se stejným hashem.

Velkým problémem u elektronických podpisů je též expirace certifikátů. Když certifikační autorita vydá certifikát, mimo jiné tím potvrzuje, že využívaný formát podpisu považuje za dostatečně silný pro několik následujících let. Po tuto dobu je elektronický podpis důvěryhodný. Ve chvíli kdy certifikát expiruje ale už podpis jím vytvořený za důvěryhodný považovaný není. To má dobré důvody – i RSA klíč lze během pár let výpočtů teoreticky prolomit.

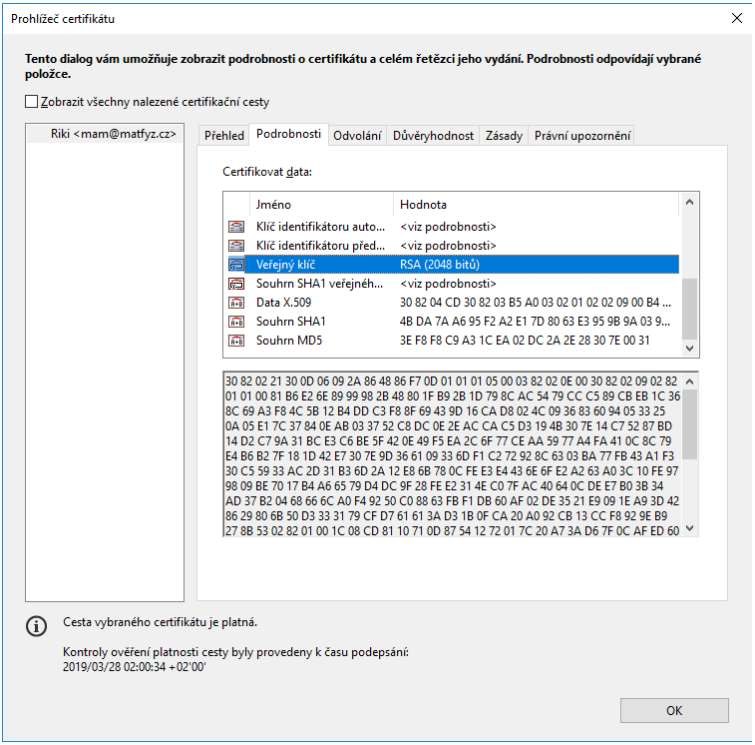
Časová razítka řeší problémy i tady. Pomocí nich je možné dokázat, že dokument byl podepsán ještě v době platnosti certifikátu. Ale protože i časová razítka jsou podpisy svázané s expirujícími certifikáty, je nutné je pravidelně obnovovat a dokumenty přerazítkovávat. Uchovávat elektronicky podepsané dokumenty tedy není vůbec jednoduché.

Problém 3

Zadání:
Ze stránek tématu `https://mam.mff.cuni.cz/problem/2203/` *si můžete stáhnout PDF podepsané pomocí RSA. Zkuste zjistit co nejvíce o použitém klíči. Jeden bod je za modul a veřejný exponent. Za privátní klíč je speciální odměna 10 bodů. Pokud ho budete chtít získat, asi se budete muset porozhlédnout po nějakých známých útocích na RSA.*

Řešení:
Informace o podepsaných PDF dokumentech umí zobrazit například Adobe Reader. Stačí v panelu podpisu, který se automaticky otevře, zvolit položku „Podrob-

nosti certifikátu“. Zde jsou informace o veřejném klíči dostupné v kartě „Podrobnosti“ jako položka „Veřejný klíč“. Příslušné okno je zachyceno na Obrázku 9.



Obrázek 9: Veřejný klíč využitý v certifikátu

Data zde jsou zde zapsána jakožto ASN.1 struktura. Pro jejich dekódování můžeme využít program OpenSSL nebo třeba webovou službu `http://lapo.it/asn1js/`. Stačí do příslušné stránky zkopírovat data a zobrazí se nám struktura se dvěma položkami. První z nich je modul o délce 2048 bitů a druhá veřejný exponent s délkou 2045 bitů.

Pokud vás zaráží, že je veřejný exponent tak dlouhý (obvykle to bývá 65537), jste na stopě řešení druhé části problému.

Trocha hledání vás může přivézt na stopu Wienerova útoku. Ten využívá vztahu pro privátní a veřejný exponent RSA klíče, pomocí kterého lze dopočítat privátní exponent v případě, že $d < \frac{1}{3}n^{\frac{1}{4}}$. To vzhledem k velikosti veřejného exponentu zde bude splněno. Podrobný popis útoku včetně příkladu lze nalézt například na anglické Wikipedii `https://en.wikipedia.org/wiki/Wiener%27s_attack`.

Nyní již jen zbývá útok naprogramovat. Trocha dalšího hledání ale ukáže, že tento úkol je jednodušší, než jsme si mysleli. Připravený skript, kam stačí zadat

hodnoty, totiž lze stáhnout z `https://github.com/orisano/owiener`. Spuštění pak je přímočaré, jak je zachyceno na Obrázku 10, kde si můžete též ověřit vyčtená data z první části problému.

```
>>> import owiener
>>> n = 0x81b6e26e8999982b48801fb92b1d798cac5479ccc589cbeb1c368c69a3f84c5b12b4dd
c3f88f69439d16cad8024c09368360940533250a05e17c37840eab033752c8dc0e2eaccac5d3194b
307e14c75287bd14d2c79a31bce3c6be5f420e49f5ea2c6f77ceaa5977a4fa410c8c79e4b6b27f18
1d42e7307e9d366109336df1c272928c6303ba77fb43a1f330c55933ac2d31b36d2a12e86b780cfe
e3e4436e6fe2a263a03c10fe979809be7017b4a66579d4dc9f28fee2314ec07fac40640cdee7b03b
34ad37b20468666ca0f49250c08863fbf1db60af02de3521e9091ea93d428629806b50d3333179cf
d761613ad31b0fca20a092cb13ccf8929eb9278b53
>>> e = 0x1c08cd8110710d87541272017c20a73ad67f0cafed60c673505cc9f15260d49175d1ce
a8d26a03a19c4491d8c48a3377dfdc6abf624b107ba63fbec5a5062b98fe85aef69940c85897a751
606595e0cef5ddaeb384e937e978ea2170d4e81c9aa847493940b09c4655fbe2b35cad8d9c9b7509
e6c47adeb14eda7c17b39229452b0897500ee66bfbdf278eb4be52d91d42d40cd5118d11f66296db
d3dc24793254ee75d78513f3ad0156927afd0907f2df343ae65eabb68ce8bb027e2e95d012337d36
913c5cf7f037b0c641d5d2e7d65f77aaf6c2967f64e5b471b5d6cd301d27475f039fe5b3bd74b6f2
8f6f7d02d12ed10d6a0e25be5d35715867e9ceee2f
>>> d = owiener.attack(e, n)
>>> print(hex(d))
0xd7fc9895546190a726e52c0c6fe9438d7be3f86c9b27ab4eeab0c716a8479c625b887d254d4417
9be7edef9ed0d6a88d3b8102d489466612d4faca7997eb49ff
>>>
```

Obrázek 10: Spuštění skriptu pro Wienerův útok

Problém 4

Zadání:
Chcete poslat řešení úlohy obsahující libovolné nesmysly a získat plný počet bodů? Právě máte šanci. Za správné bude pokládáno každé řešení, které bude zasláno elektronicky v PDF podepsaném klíčem, který najdete na webových stránkách tématu.

Řešení:
Úlohu lze řešit téměř libovolným softwarem pro podepisování PDF dokumentů. Osobně používám jednoduchou utilitu PortableSigner dostupnou na adrese `http://portablesigner.sourceforge.net/`.

Ve všech případech po nás program pro tvorbu podpisu bude chtít nejen privátní klíč, ale i certifikát. Protože nemáme žádnou certifikační autoritu, podepíšeme si ho sami (tzv. *self-signed* certifikát). Ten sice nebude nikdo další považovat za důvěryhodný, ale to nám nevadí. Na vytvoření self-signed certifikátu může navézt též předchozí úloha, kde je prezentováno vzorově podepsané PDF.

Self-signed certifikát si můžeme vytvořit libovolný například pomocí knihovny OpenSSL příkazem:

```
openssl req -x509 -key private_key.pem -out cert.pem -days 365
```

PortableSigner vyžaduje privátní klíč a certifikát zabalený do jednoho souboru ve formátu PKCS#12. K tomu opět pomůže OpenSSL:

```
openssl pkcs12 -export -out cert.pfx \
  -inkey private_key.pem -in cert.pem
```

Nyní již stačí vše doplnit do grafického okénka PortableSigneru.

Problém 5

Zadání:
Nejprimitivnějším přístupem pro tvorbu HMAC by bylo počítat Hash(klíč ‖ ‖ zpráva), kde ‖ značí spojení textových řetězců.

Takový přístup by ale při použití běžných hashovacích funkcí (SHA1, SHA256, RIKSHA, ...) umožňoval při odposlechu podepsaných zpráv podepisovat další zprávy i bez znalosti klíče. Za teoretický popis slabiny nabízíme 2 body.

Další 4 body lze získat za praktickou demonstraci. Zachytili jste zprávu „Mam rad jablka.“ a její podpis 50BA1E0A5660AC90. Víte, že podpis je spočítán jako RIKSHA(klíč ‖ zpráva). Dokážete získat podpis zprávy, která obsahuje mimo jiné text „Nemam rad hrusky“?

Popis hashovací funkce RIKSHA včetně její kompletní implementace lze najít ve 2. čísle.

Řešení:
Pamatujete si na řešení třetí úlohy z druhého čísla? Pokud ne, najdete ho ve čtvrtém čísle[9]. Píše se tam, že za zahashovaný text je přidáván padding, aby se částečně zabránilo podpisu prodloužené zprávy. Proč částečně? Protože hashovací funkce jako SHA1, SHA2 nebo RIKSHA jsou navrženy tak, že vůči *length extension attack* zcela odolné nejsou.

Vždy je podepsána zpráva včetně paddingu, ale za tento padding je možné přidat cokoli dalšího a hodnotu hashe dopočítat. Dokážeme tedy spočítat podpis pro zprávu „Mam rad jablka.<padding>Nemam rad hrusky<padding>“. Když se s tímto vědomím podíváme na konstrukci HMAC, hned je trochu jasnější, proč je na první pohled tak zbytečně složitá.

Pro praktickou realizaci můžeme využít vzorovou implementaci funkce RIKSHA. Stačí nastavit na jejím počátku interní stav (proměnné $h0$ až $h7$) na poskytnutou hodnotu hashe a pak zahashovat zprávu „Nemam rad hrusky“.

Na co nezbylo místo

V tématu jsme se vám snažili představit myšlenky a konstrukty, které se objevují všude kolem vás, často aniž byste o tom věděli, například při běžné práci na internetu. Ze základních algoritmů jsme se nedostali k Diffie-Hellmanově výměně klíče, tedy k algoritmu, jak si dvě strany mohou dohodnout tajný klíč po nechráněném kanálu. Ani na tomto algoritmu ale není z matematického pohledu nic složitého.

[9] `https://mam.mff.cuni.cz/media/cislo/pdf/25/25-4.pdf`

Dalo by se říct, že ještě nedávno moderní kryptografie nebyla postavena na žádné příliš komplikované matematice. Ale situace se poslední dobou začíná měnit. Čím dál více se rozšiřují algoritmy využívající eliptické křivky, což je velmi obtížné matematické odvětví. Navíc se intenzivně zkoumají šifry, které dokáží odolat kvantovému počítači, a mnoho návrhů též využívá nějakou komplikovanou matematickou teorii.

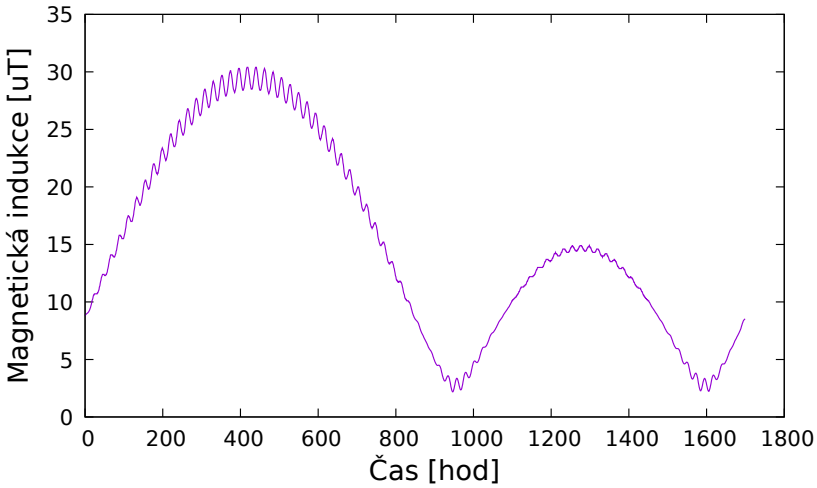
Kuba a Káťa; jakub.topfer@matfyz.cz
e-mailová konference: krypto@mam.mff.cuni.cz

Téma 3 – Neznámý měsíc

Téma 3.3 řešení

Problém 1

Zadání:
Vysvětlete, čím je způsobený tvar grafu na obrázku 11.



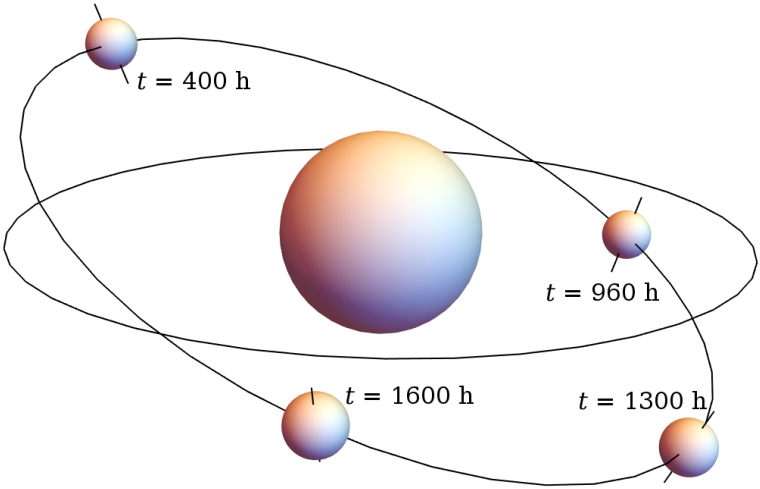
Obrázek 11: Měření magnetické indukce po dobu jednoho oběhu kolem planety

Řešení:
Rozeberme graf ze zadání: V úsecích od 0 h do 960 h, od 960 h do 1600 h a od 1600 h do konce vidíme přibližně části sinusoidy; pokud v úseku od 960 h do 1600 h změníme znaménko, dostaneme celou jednu periodu sinusoidy.

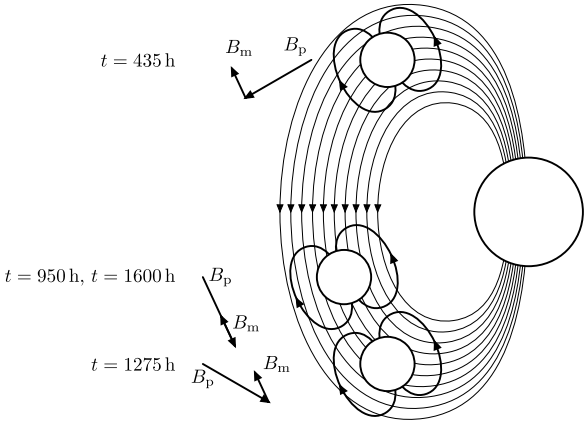
To interpretujeme tak, že na vrcholcích je maximální vychýlení od rovníkové roviny planety. Minima zase interpretujeme tak, že vektor magnetické indukce planety je v opačném směru než vektor magnetické indukce měsíce, a proto se celkové pole při pohybu v obou směrech bude zvětšovat. Z toho plyne, že máme

docela zvláštní měsíc, jelikož aby se takto měnila orientace těchto dvou vektorů magnetické indukce, tak se musí měnit směr rotační osy měsíce, viz obrázek 12. Ze symetrie dostáváme, že perioda precese je rovna periodě oběhu.

Za takto zvláštní situaci se omlouváme, přestože může nastat, je to velmi nepravděpodobné. V tomto případě vznikla chybou v kódu generujícím graf.



Obrázek 12: Směr rotační osy měsíce během oběhu



Obrázek 13: Znázornění siločar magnetického pole na planetě

Problém 2

Zadání:

Určete amplitudu magnetické indukce na pólech měsíce, pokud zanedbáte pole způ-

sobené planetou, a amplitudu magnetické indukce na pólech planety, pokud zanedbáte pole způsobené měsícem.

Řešení:
Pro výpočet síly magnetických polí na pólech planety a měsíce potřebujeme určit magnetické náboje Q_p a Q_m. Podíváme se na okamžiky $t = 960\,\mathrm{h}$ a $t = 1600\,\mathrm{h}$, kdy má magnetické pole na rovníku měsíce nejmenší hodnotu. To je způsobeno tím, že pole planety a měsíce mají opačný směr a odečtou se. Rozdíl velikosti pole na odvrácené a přivrácené straně měsíce k planetě je roven rozdílu hodnot pole generovaného planetou v těchto dvou místech, protože magnetické pole je v nich stejné (rotuje s planetou). Tento rozdíl můžeme z grafu vyčíst, je to rozdíl minim a maxim malých vlnek kolem hodnoty $t = 960\,\mathrm{h}$.

Vektorový vztah pro magnetickou indukci je

$$\mathbf{B} = \sum_Q \frac{Q}{(r - r_Q)^3}(\mathbf{r} - \mathbf{r}_Q),$$

kde sčítáme přes všechny náboje.

Pro výpočet magnetického pole způsobeného planetou musíme zjistit polohu magnetometru vzhledem k magnetickým nábojům planety. Zavedeme si souřadnice (x,y) s počátkem ve středu planety. Osa y míří ve směru rotační osy planety a osa x rotuje s oběhem měsíce tak, že měsíc pořád leží v rovině xy. y-ová souřadnice středu měsíce je

$$y_\mathrm{m} = a_\mathrm{m} \sin\theta \sin(2\pi(t_0 + t)/T) = -6{,}685 \cdot 10^7\,\mathrm{m},$$

kde a_m je poloměr oběhu měsíce, θ je sklon oběžné dráhy vůči rovníkové rovině planety, t_0 je čas průchodu měsíce rovníkovou rovinou planety a v našem případě má hodnotu $t_0 = 0\,\mathrm{h}$, čas $t = 960\,\mathrm{h}$ a $T = 1700\,\mathrm{h}$ je perioda oběhu. Z y-ové souřadnice dopočítáme x-ovou souřadnici středu měsíce

$$x_\mathrm{m} = \sqrt{a_\mathrm{m}^2 - y_\mathrm{m}^2} = 3{,}944 \cdot 10^8\,\mathrm{m}.$$

Dále potřebujeme spočítat směr rotační osy měsíce. Víme, že magnetické pole měsíce, které je rovnoběžné s jeho rotační osou, je v čase t rovnoběžné i s magnetickým polem planety (viz obrázek 14). To má hodnotu

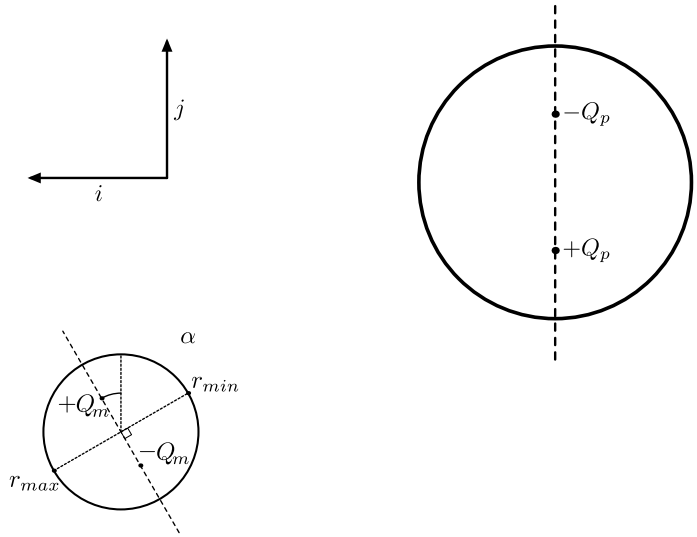
$$\mathbf{B}_\mathrm{p} = \frac{+Q_\mathrm{p}\left(x_\mathrm{m}\mathbf{i} + \left(y_\mathrm{m} + \frac{R_\mathrm{p}}{2}\right)\mathbf{j}\right)}{\left(x_\mathrm{m}^2 + \left(y_\mathrm{m} + \frac{R_\mathrm{p}}{2}\right)^2\right)^{3/2}} + \frac{-Q_\mathrm{p}\left(x_\mathrm{m}\mathbf{i} + \left(y_\mathrm{m} - \frac{R_\mathrm{p}}{2}\right)\mathbf{j}\right)}{\left(x_\mathrm{m}^2 + \left(y_\mathrm{m} - \frac{R_\mathrm{p}}{2}\right)^2\right)^{3/2}} =$$

$$= Q_\mathrm{p}(8{,}129 \cdot 10^{-19}\mathbf{i} + 1{,}539 \cdot 10^{-18}\mathbf{j})\,\mathrm{m}^{-2},$$

kde $\mathbf{i}$, resp. $\mathbf{j}$, je jednotkový vektor ve směru osy x, resp. y. Úhel mezi vektorem magnetické indukce $\mathbf{B}_\mathrm{p}$ a kladnou poloosou y je

$$\alpha = \arctan\frac{B_{\mathrm{p}x}}{B_{\mathrm{p}y}} = \arctan\frac{8{,}129 \cdot 10^{-19}}{1{,}539 \cdot 10^{-18}} = 27{,}8^\circ.$$

Nyní můžeme spočítat polohu rovníkových bodů na přivrácené a odvrácené straně měsíce (viz obrázek 14).



Obrázek 14: Pozice magnetických pólů a bodů, kde je největší $r_{\max}$ a nejmenší $r_{\min}$

$$\mathbf{r}_{\min} = (x_{\mathrm{m}} - \cos(\alpha)R_{\mathrm{m}})\mathbf{i} + (y_{\mathrm{m}} + \sin(\alpha)R_{\mathrm{m}})\mathbf{j} = (3{,}900 \cdot 10^8\mathbf{i} - 6{,}252 \cdot 10^7\mathbf{j})\,\mathrm{m}$$

$$\mathbf{r}_{\max} = (x_{\mathrm{m}} + \cos(\alpha)R_{\mathrm{m}})\mathbf{i} + (y_{\mathrm{m}} - \sin(\alpha)R_{\mathrm{m}})\mathbf{j} = (3{,}988 \cdot 10^8\mathbf{i} - 6{,}919 \cdot 10^7\mathbf{j})\,\mathrm{m}$$

Spočítáme rozdíl magnetických polí generovaných planetou v těchto bodech

$$\Delta B = |\mathbf{B}_{\mathrm{p}}(\mathbf{r}_{\min}) - \mathbf{B}_{\mathrm{p}}(\mathbf{r}_{\max})|$$

$$\Delta B = \left| \left(\frac{+Q_{\mathrm{p}}(\mathbf{r}_{\min} + \frac{R_{\mathrm{p}}}{2}\mathbf{j})}{\left|\mathbf{r}_{\min} + \frac{R_{\mathrm{p}}}{2}\mathbf{j}\right|^3} + \frac{-Q_{\mathrm{p}}(\mathbf{r}_{\min} - \frac{R_{\mathrm{p}}}{2}\mathbf{j})}{\left|\mathbf{r}_{\min} - \frac{R_{\mathrm{p}}}{2}\mathbf{j}\right|^3} \right) - \right.$$

$$\left. - \left(\frac{+Q_{\mathrm{p}}(\mathbf{r}_{\max} + \frac{R_{\mathrm{p}}}{2}\mathbf{j})}{\left|\mathbf{r}_{\max} + \frac{R_{\mathrm{p}}}{2}\mathbf{j}\right|^3} + \frac{-Q_{\mathrm{p}}(\mathbf{r}_{\max} - \frac{R_{\mathrm{p}}}{2}\mathbf{j})}{\left|\mathbf{r}_{\max} - \frac{R_{\mathrm{p}}}{2}\mathbf{j}\right|^3} \right) \right| =$$

$$= Q_{\mathrm{p}} \cdot 1{,}314 \cdot 10^{-19}\,\mathrm{m}^{-2}.$$

Z grafu můžeme vyčíst, že kolem času $t = 960\,\mathrm{h}$ mají malé vlnky způsobené rotací měsíce amplitudu $\Delta B = 1{,}3\,\mu\mathrm{T}$. Z toho plyne

$$\Delta B = 1{,}3\,\mu\mathrm{T} = Q_{\mathrm{p}} \cdot 1{,}314 \cdot 10^{-19}\,\mathrm{m}^{-2} \Rightarrow Q_{\mathrm{p}} = 9{,}89 \cdot 10^{12}\,\mathrm{T} \cdot \mathrm{m}^2.$$

Dále spočítáme velikost magnetických nábojů měsíce. Magnetické pole způsobené měsícem bude pouze v severojižním směru vzhledem k měsíci, jelikož jsme na rovníku. Velikost pole je

$$B_\mathrm{m} = \frac{Q_\mathrm{m}\frac{R_\mathrm{m}}{2}}{\left(R_\mathrm{m}^2 + \left(\frac{R_\mathrm{m}}{2}\right)^2\right)^{3/2}} + \frac{-Q_\mathrm{m}\frac{-R_\mathrm{m}}{2}}{\left(R_\mathrm{m}^2 + \left(\frac{R_\mathrm{m}}{2}\right)^2\right)^{3/2}} =$$

$$= \frac{Q_\mathrm{m} R_\mathrm{m}}{\left(\left(R_\mathrm{m}^2 + \left(\frac{R_\mathrm{m}}{2}\right)^2\right)^{\frac{3}{2}}\right)} = Q_\mathrm{m} \cdot 2{,}862 \cdot 10^{-14}\,\mathrm{m}^{-2}.$$

V grafu vidíme, že velikost magnetického pole na přivrácené straně měsíce je $B(\mathbf{r}_\mathrm{min}) = 3{,}3\,\mu\mathrm{T}$ (maximum malých vlnek kolem $t = 960\,\mathrm{h}$). Pole generované planetou zde má velikost $B_\mathrm{p}(\mathbf{r}_\mathrm{min}) = 17{,}8\,\mu\mathrm{T}$. Náboj spočítáme z rozdílu těchto polí

$$Q_\mathrm{m} = \frac{B_\mathrm{m}}{2{,}862 \cdot 10^{-14}\,\mathrm{m}^{-2}} = \frac{B(\mathbf{r}_\mathrm{min}) - B_\mathrm{p}(\mathbf{r}_\mathrm{min})}{2{,}862 \cdot 10^{-14} m^{-2}} =$$

$$= \frac{3{,}3\,\mu\mathrm{T} - 17{,}8\,\mu\mathrm{T}}{2{,}862 \cdot 10^{-14} m^{-2}} = -5{,}07 \cdot 10^{8}\,\mathrm{T}\cdot\mathrm{m}^2.$$

Když už známe náboj magnetických pólů, můžeme snadno spočítat magnetickou indukci na rotačních pólech planety a měsíce. Děláme to kvůli tomu, že náš model magnetického náboje nemá přímou fyzikální reprezentaci a je platný jen pro vnější pole planety/měsíce.

$$B_\mathrm{p,pól} = \left(\frac{Q_\mathrm{p}}{\frac{R_\mathrm{p}}{2}^2} - \frac{Q_\mathrm{p}}{\frac{3R_\mathrm{p}}{2}^2}\right) = 2{,}91\,\mathrm{mT}$$

$$B_\mathrm{m,pól} = \left(\frac{Q_\mathrm{m}}{\frac{R_\mathrm{m}}{2}^2} - \frac{Q_\mathrm{m}}{\frac{3R_\mathrm{m}}{2}^2}\right) = 77{,}9\,\mu\mathrm{T}$$

Kuba a Kubo; `kusnir.jk@gmail.com`
e-mailová konference: `mesic@mam.mff.cuni.cz`

Téma 4 – Algoritmy od nuly (do n)

Milí řešitelé, naše tématko je u konce a už nám zbývá jen zveřejnit řešení posledních dvou sérií, která naleznete níže. Děkuji všem, kteří se do tématka zapojili a zkusili něco vyřešit. Pokud vás zde probírané problémy zaujaly, vězte, že existuje skvělá knížka *Průvodce labyrintem algoritmů* [1], která obsahuje téměř vše, čím jsme se v tématku zabývali a k tomu velkou spoustu dalších zajímavých věcí. Tímto vám ji tedy vřele doporučuji a těším se na vaši účast v dalším ročníku.

Řešení čtvrté série

Problém 1

Zadání:
Spojový seznam se v praxi používá hlavně v případech, kdy potřebujeme vkládat prvky doprostřed seznamu. Skutečné procesory totiž typicky načítají naráz velký úsek paměti, takže s daty, která jsou v paměti uložena blízko sebe, umí pracovat výrazně rychleji než se spojovým seznamem, který má data roztroušená různě po paměti. Navrhněte způsob implementace fronty pomocí pole při zachování časové i prostorové složitosti. Můžete předpokládat, že předem znáte maximální počet prvků ve frontě.

Řešení:
Maximální počet prvků fronty si označíme n. Představíme si, že pole délky n „stočíme do kruhu“ – za posledním prvkem bude opět následovat první. Nyní do něj můžeme frontu jednoduše uložit – vkládat budeme prostě vždy na následující pozici v poli. Začátek a konec fronty se nikdy nepřekryje, protože kruhové pole je vždy delší než aktuální počet prvků fronty. Můžeme si to také představit tak, že vkládáme do nekonečně dlouhého pole, ale indexy ve skutečném poli počítáme modulo n. Tak v podstatě vypadá i implementace našeho kruhového pole.

Vytvoříme si pole délky n, kde n je maximální počet prvků ve frontě. K tomu si vytvoříme dvě proměnné *prvni* a *posledni*, ve kterých si budeme pamatovat, kde v poli leží první a poslední prvek fronty. Při vkládání nového prvku nejprve přepočítáme hodnotu *posledni*: $posledni \leftarrow (posledni + 1) \bmod n$ a pak na místo v poli, kam *posledni* nově ukazuje, vložíme nový prvek. Funguje to tedy tak, jak bylo popsáno výše. V poli se posunujeme vždy o 1 doprava a pokud bychom jej měli opustit, vrátíme se zpátky na začátek. Podobným způsobem budeme z fronty mazat – po smazání prvku bude $prvni \leftarrow (prvni + 1) \bmod n$.

Vkládání i mazání evidentně umíme v $\mathcal{O}(1)$ a paměťová složitost je lineární vzhledem k maximálnímu počtu prvků fronty.

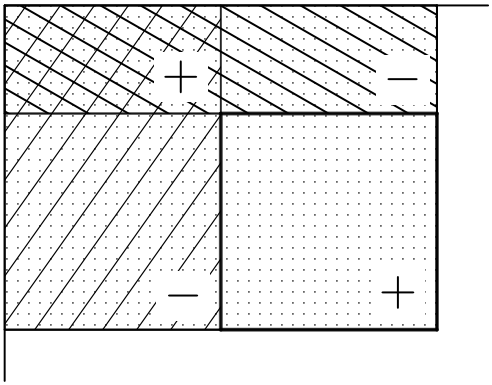
Problém 2

Zadání:
Co kdybychom místo posloupnosti měli tabulku celých čísel o rozměrech $n \times m$? Navrhněte co nejefektivnější datovou strukturu, které zadáme „podtabulku“ (souvislý obdélník v naší tabulce) a vrátí nám součet všech čísel v této podtabulce. Podtabulku budeme zadávat jako pozici levého horního a pravého dolního rohu. Nezapomeňte na časovou a prostorovou složitost.

Řešení:
Pro tabulku si předpočítáme *tabulkové součty*. Ty nám budou pro každý obdélník, jehož levý horní roh leží v levém horním rohu tabulky a pravý dolní roh na pozici $[i, j]$, udávat jeho součet. Jak rychle dokážeme tabulkové součty spočítat? Mohlo by se zdát, že spočítat jeden tabulkový součet nám zabere lineární čas vzhledem k velikosti tabulky – sčítáme při tom až $n \times m$ čísel. Problém je ale analogický

k počítání klasických prefixových součtů. Pokud bychom každý z nich počítali zvlášť, zabral by nám každý součet čas $\mathcal{O}(n)$, my ale využíváme vždy předchozího součtu. Použijeme tedy stejný trik. Součet obdélníku s pravým dolním rohem na políčku $[i, j]$ si označíme $S[i, j]$, hodnotu na tomto políčku budeme značit $P[i, j]$. Nyní můžeme $S[i, j]$ spočítat jako $S[i-1, j] + S[i, j-1] - S[i-1, j-1] + P[i, j]$. Součty tedy můžeme počítat postupně po řádcích shora dolů a zleva doprava. V každém okamžiku máme spočítané všechny součty vlevo nahoře od aktuálního součtu, umíme ho tudíž spočítat v konstantním čase a výpočet všech tabulkových součtů nám proto zabere čas $\mathcal{O}(nm)$.

Zbývá říct, jak z tabulkových součtů spočítat součet libovolného obdélníku. To uděláme podobně, jako když jsme počítali jednotlivé tabulkové součty z předchozích – stačí od velkého obdélníku s levým horním rohem v rohu tabulky odečíst dva menší a opět přičíst obdélník, který jsme odečetli dvakrát. Lépe než ze zdlouhavého popisu je to vidět z Obrázku 15. Součet libovolného obdélníku tak dokážeme zjistit v konstantním čase.



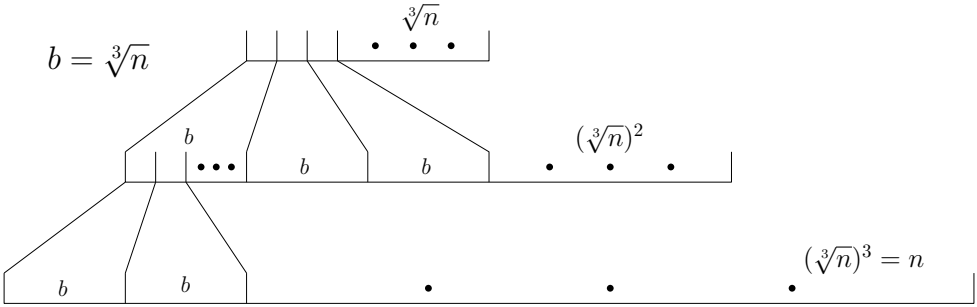
Obrázek 15: Jak z tabulkových součtů spočítat součet obdélníku

Problém 3

Zadání:
Naši datovou strukturu pro počítání intervalových minim jde ve skutečnosti ještě vylepšit. Navrhněte obdobnou blokovou datovou strukturu na počítání minim, která využívá tří úrovní místo dvou a získejte tak časovou složitost operací $\mathcal{O}(\sqrt[3]{n})$ (nezapomeňte všechny složitosti řádně zdůvodnit).

Řešení:
Strukturu vytvoříme zcela analogicky. Posloupnost rozdělíme na bloky druhé úrovně velikosti $\sqrt[3]{n}$, kterých bude evidentně $\frac{n}{\sqrt[3]{n}}$. Spočítáme posloupnost minim těchto bloků, která bude mít délku $\frac{n}{\sqrt[3]{n}}$, a tu rozdělíme na bloky první úrovně velikosti $\sqrt[3]{n}$, kterých bude $\frac{n}{\sqrt[3]{n^2}} = \sqrt[3]{n}$. Nakonec vytvoříme posloupnost minim

těchto bloků, která už bude mít délku $\sqrt[3]{n}$. Celá inicializace struktury nám zabrala lineární čas. Strukturu můžete vidět schematicky znázorněnou na obrázku 16.



Obrázek 16: Tříúrovňová bloková struktura

Dotazy na strukturu budou rovněž analogické. Dotaz na úsek posloupnosti můžeme rozdělit na počáteční a koncovou část, z nichž každá má délku nejvýše $\sqrt[3]{n}$, a na středovou část, která se skládá z celých bloků druhé úrovně. Počáteční a koncovou část celou projdeme a najdeme minimum a na středovou část se zeptáme kratší posloupnosti. Dotaz na ni ale opět stejným způsobem rozdělíme na počáteční, středovou a koncovou část (počáteční a koncová mají opět každá délku nejvýše $\sqrt[3]{n}$ a můžeme je opět lineárně projít). Minimum středové části (která se skládá z bloků první úrovně) spočteme lineárním průchodem nejkratší posloupnosti a jsme hotovi. Celá operace nám trvala čas $\mathcal{O}(\sqrt[3]{n})$, protože jsme pětkrát prošli úsek délky $\sqrt[3]{n}$ a poté jen porovnali výsledných 5 prvků.

Změnu prvku rovněž zvládneme v čase $\mathcal{O}(\sqrt[3]{n})$ – stačí znovu spočítat minimum bloku druhé úrovně, ve kterém prvek leží, a poté minimum bloku první úrovně, ve kterém leží příslušný blok druhé úrovně.

Problém 4

Zadání:
Proč bychom měli zůstat u tří úrovní? Spočítejte, jaký počet úrovní je optimální, a určete časovou složitost inicializace i jednotlivých operací a složitost prostorovou. Získáte tak velmi zajímavou a užitečnou datovou strukturu.

Řešení:
Velikost bloku si označíme b, počet úrovní k. Jedna operace na struktuře nám bude trvat $\mathcal{O}(bk)$ – při zjišťování minima úseku procházíme na každé úrovni maximálně dva bloky a při změně prvku na každé úrovni maximálně jeden blok přepočítáme. Chtěli bychom tedy minimalizovat hodnotu bk. K tomu by se nám hodilo vědět, jak na sobě b a k závisí. Na každé úrovni snížíme počet prvků b-krát, počet úrovní k nám tedy říká, kolikrát musíme n vydělit velikostí bloku b, abychom dostali číslo 1. To je ale definice logaritmu, takže z toho dostáváme, že $k = \log_b(n)$. Stačí nám tedy minimalizovat funkci $b \cdot \log_b(n)$. Zároveň víme, že b jakožto velikost bloku

musí být rovno alespoň dvěma a zároveň nemůže být větší než n. Z toho už je jasné, že chceme zvolit b rovno dvěma, protože první činitel roste s b lineárně, zatímco druhý klesá s b logaritmicky.

Naše struktura tedy bude úplný binární strom – v listech budou uloženy prvky posloupnosti a v každém vnitřním vrcholu bude uloženo minimum z jeho synů. Inicializaci stále zvládneme v lineárním čase – potřebujeme lineárně projít každou posloupnost a součet délek posloupností je $n + n/2 + n/4 + n/8 \ldots$, což je menší rovno $2n$. Dotazy nám budou trvat $\mathcal{O}(\log(n))$.

Tato struktura je v informatice známa pod názvem intervalový strom a existují k ní různá vylepšení a speciální varianty, které jsou ale nad rámec našeho tématka. Můžete se o nich dočíst v již dříve zmiňované knížce *Průvodce labyrintem algoritmů* [1].

Řešení páté série

Problém 1

Zadání:
Máte zadanou orientovanou přímku $A[x_1, y_1]$, $B[x_2, y_2]$ a bod, který na ní neleží. Vymyslete, jak v konstantním čase zjistit, jestli bod leží vpravo nebo vlevo od přímky. Nezapomeňte na speciální případy, jako je přímka rovnoběžná s osou souřadnic.

Řešení:
Nechť má náš bod ze zadání souřadnice $C[x_3, y_3]$. Dosazením obou bodů do rovnice přímky $y = ax + b$ získáme koeficienty a a b. Dále se podíváme na hodnotu výrazu $ax_3 + b$. Pokud je hodnota větší než y_3, leží leží bod „pod“ přímkou, pokud je menší, leží bod „nad“ přímkou. My chceme znát polohu bodu C vzhledem k pořadí bodů A a B, porovnáme tedy jejich x-ové souřadnice. Pokud $x_1 < x_2$, znamená „pod přímkou“ napravo a „nad přímkou“ nalevo, v opačném případě je to samozřejmě naopak. Zbývají nám vyřešit speciální případy. Pokud je přímka rovnoběžná s osou x, popsaný postup stále funguje. Pokud je rovnoběžná s osou y, porovnáme x-ovou souřadnici dvojice bodů A, B a bodu C, čímž zjistíme, zda C leží vlevo či vpravo od přímky a stranu případně změníme na opačnou, pokud bod A leží nad bodem B. Všechny operace nám zabraly konstantní čas, jsme tedy hotovi.

Problém 2

Zadání:
Máte konvexní n-úhelník zadaný výčtem bodů na obvodu a bod. Zjistěte, zda bod leží uvnitř n-úhelníku. Nezapomeňte na časovou složitost.

Řešení:
Přímočaře využijeme první úlohu. Projdeme n-úhelník po obvodu a pro každou hranu orientovanou v směru procházení zkontrolujeme, zda bod leží vlevo nebo vpravo od ní. Pokud dostaneme pro všechny hrany stejnou odpověď, leží bod

v n-úhelníku, v opačném případě nikoli. Celý algoritmus nám evidentně zabral lineární čas.

Problém 3

Zadání:
Vyřešte předchozí problém i pro nekonvexní n-úhelníky

Řešení:
Úlohu rovněž zvládneme vyřešit v lineárním čase, půjdeme na to ale tentokrát trochu jinak. Vezmeme si přímku rovnoběžnou s osou x, která prochází daným bodem. Nyní si uvědomíme, že pokud bod leží v n-úhelníku, protíná tato přímka jeho hranici napravo od něj v lichém počtu míst, protože začíná uvnitř n-úhelníku a končí venku. Naopak, pokud by bod ležel vně n-úhelníku, protínala by přímka jeho hranici v sudém počtu míst, jelikož by začínala venku a končila rovněž venku.

Zbývá nám tedy zkontrolovat počet míst napravo od daného bodu, ve kterých jím vedená přímka rovnoběžná s osou souřadnic protíná hranici n-úhelníku. To uděláme tak, že projdeme postupně všechny hrany n-úhelníku a pro každou z nich spočítáme, zda ji přímka protíná, a poté zkontrolujeme, zda průsečík leží napravo od daného bodu. Zvlášť musíme vyřešit případy, ve kterých přímka prochází vrcholem n-úhelníku. Přímka totiž může vrcholem projít, aniž by se přesunula z vnějšku dovnitř n-úhelníku či naopak. Pro každý vrchol, kterým přímka prochází, tedy navíc zkontrolujeme, že pro obě příslušné hrany leží daný bod na stejné straně. Pokud ano, vrchol započítáme, v opačném případě nikoli.

Zbývá nám ještě poslední speciální případ, a to ten, ve kterém některá z hran n-úhelníku leží na naší přímce. Můžeme si ale všimnout, že jsme nikde nevyužili toho, že je přímka rovnoběžná s osou x. Zvolíme si tedy přímku tak, aby tento případ vůbec nenastal. Projdeme si všechny hrany, zjistíme, která z nich má největší směrnici (svislé hrany přeskočíme) a směrnici naší přímky zvolíme větší, než maximum směrnic hran. Tím jsme hotovi.

Problém 4

Zadání:
Máte zadáno n bodů v rovině. Vaším úkolem je vymyslet co nejrychlejší algoritmus na nalezení nejmenšího konvexního k-úhelníku, který obsahuje všechny body. Můžete si to představit tak, že kolem hřebíků zatlučených v rovné desce natáhneme velkou kruhovou gumičku. Gumička tvoří hledaný konvexní k-úhelník. Výstupem vašeho algoritmu by měl být seznam vrcholů hledaného k-úhelníku v pořadí, v jakém leží na obvodu (každý z vrcholů bude evidentně roven některému z bodů na vstupu).

Řešení:
Nejdříve bych se chtěl omluvit za drobnou chybu v zadání – použili jsme v něm proměnnou n ve dvou různých významech, zadání výše je již opravené.

Hledaný konvexní k-úhelník se nazývá konvexní obal a existuje známý algoritmus na jeho sestrojení. Využijeme zametání roviny, které jsme si předváděli

na hledání průsečíků úseček. Budeme tedy zametat rovinu svislou přímkou zleva doprava a našimi událostmi budou jednotlivé body. V každé chvíli budeme mít sestrojený konvexní obal prvních l bodů a budeme do něj přidávat $(l+1)$-ní bod. Ten bude v novém konvexním obalu určitě ležet, protože je nejvíce napravo. Možná se ale jeho přidáním stane, že některé ze starších bodů už na konvexním obalu neleží. Projdeme tedy nový konvexní obal po i proti směru hodinových ručiček od nově přidaného bodu a všechny body, které porušují konvexitu, odstraníme. Takto budeme postupovat, dokud nebude konvexní obal kompletní.

Pro implementaci si rozdělíme konvexní obal na dvě části – takzvanou horní a dolní obálku. Horní obálka začíná v nejlevějším bodu a pokračuje vrchem až k nejpravějšímu bodu. Nyní si popíšeme, jak horní obálku sestrojit – konstrukce dolní obálky je analogická.

Body si budeme do horní obálky přidávat postupně podle jejich x-ové souřadnice. Před přidáním každého bodu ale zkontrolujeme, zda leží napravo od přímky tvořené posledními dvěma body, které již v konvexním obalu leží. Pokud není tato vlastnost splněna, odebereme z obálky poslední bod a pokusíme se opět nový bod stejným způsobem přidat. Pokud jsme během přidávání nového bodu odebrali všechny body kromě počátečního, přidáme ho v každém případě. Takto sestrojená horní obálka je zřejmě horní částí konvexního obalu, protože jsme udržovali základní invariant, že zahýbáme vždy doprava, a všechny zbylé body leží napravo od každé z hran horní obálky.

Zbývá nám rozmyslet časovou složitost. Ta závisí na tom, zda jsme na začátku dostali body již setříděné. Pokud ano, tak je složitost lineární – každý bod do každé z obálek jednou přidáme a maximálně jednou odebereme a každé přidání i odebrání nás stojí konstantní čas. Pokud jsme body setříděné nedostali, bude časová složitost $\mathcal{O}(n \log n)$ – body musíme nejdříve setřídit.

Reference

[1] M. Mareš, T. Valla: Průvodce labyrintem algoritmů, CZ.NIC, z.s.p.o., CC BY-ND 3.0 CZ, dostupné online na `http://pruvodce.ucw.cz`

Tom; `domestomas+mam@gmail.com`
e-mailová konference: `algoritmy@mam.mff.cuni.cz`

Téma 5 – Přeplněná tramvaj

Řešení 4. série

Problém 1

Zadání:
Zobecněte známý tvar křivky s rovnocennými stanicemi $S(m) = Cm(N - m)$ na situace s nerovnocennými stanicemi popsanými systémem koeficientů $\alpha_1, \ldots, \alpha_N$. Pro koeficienty má platit $\alpha_1 : \ldots : \alpha_N = D_1 : \ldots : D_N = p_1 : \ldots : p_N$, kde D_k je střední počet pasažérů vygenerovaných na k-té stanici a p_k je pravděpodobnost pro každého z těchto pasažérů, že si vybere za svůj cíl stanici k.

Řešení:
Předpokládejme, že všechny koeficienty α_k jsou přirozená čísla. Označme jejich součet $M := \alpha_1 + \cdots + \alpha_N$.

Nyní si představme, že máme linku s M rovnocennými stanicemi. Rozdělme ji do N segmentů tak, aby k-tý segment obsahoval právě α_k stanic. Ve všech stanicích k-tého segmentu dohromady bylo vygenerováno $D\alpha_k$ pasažérů. Vyberme libovolného právě vygenerovaného pasažéra. Ten si volí svůj cíl s uniformním rozdělením mezi stanicemi. Kteroukoli stanici linky si zvolí s pravděpodobností $1/M$. Stanici z k-tého segmentu si zvolí s pravděpodobností α_k/M.

Ve chvíli, kdy tramvaj opouští k-tý segment, je uvnitř ní pasažérů

$$S(m) = Cm(M - m) = C(\alpha_1 + \cdots + \alpha_k)(\alpha_{k+1} + \cdots + \alpha_N).$$

Nyní proveďme trik. Ztotožněme k-tý segment s pomyslnou k-tou stanicí. Dostáváme linku s N nerovnocennými stanicemi a pro ni přesný tvar křivky $S(m)$. Navíc z předchozího plyne, že je splněn vztah mezi koeficienty $\alpha_1 : \ldots : \alpha_N = D_1 : \ldots : D_N = p_1 : \ldots : p_N$.

Pro přehlednost nadále ustanovme normalizaci koeficientů α_k jako $\alpha_1 + \cdots + \alpha_N = N$.

Nakonec zobecněme uvedené řešení na koeficienty, které nejsou přirozené. Jsou-li α_k racionální, vždy je lze přenásobením jistou konstantou převést na problém přirozených koeficientů. Řešení ani vztah mezi koeficienty se tím nezmění, jen dostaneme jiný parametr C. Opětným vydělením stejnou konstantou tedy máme uvedený tvar $S(m)$ se stejným C pro jakékoli racionální koeficienty.

Jsou-li α_k iracionální čísla, lze je libovolně aproximovat čísly racionálními. Křivka $S(m)$ má tvar výše uvedený pro každou takovou aproximaci (mění se pouze číselná hodnota α_k) a lze očekávat mezi koeficienty α_k a křivkou $S(m)$ spojitou závislost. Proto uvedený tvar platí také pro α_k iracionální. □

Problém 2

Zadání:
Každá stanice vygeneruje v průměru D pasažérů, z nich si každý uniformně vybírá za svůj cíl některou ze zbývajících stanic s tím, že vzdálenost jím vybrané cílové stanice od jeho stanice výchozí musí být v intervalu $[n_0, n_1]$. Vypočítejte křivku $S(m)$

Řešení:
Úlohu lze řešit poměrně přímočarým způsobem, tj. zcela analogicky částem problému 2.1a a 2.1b. Přímé rozepsání by však bylo tentokrát velmi komplikované z důvodu velkého množství situací, které mohou nastat. Některé detaily proto nebudeme dotahovat do konce a nebude to ani vyžadováno od řešitelů.

Pro a, b celočíselná zaveďme[10] pro zjednodušení zápisu funkci F, která udává, kolik přirozených čísel z intervalu $[a, b]$ leží mimo interval $[1, N]$.

$$\mathsf{F}(a, b) := |\{a, \ldots, b\} \setminus \{1, \ldots, N\}|$$

Počet pasažérů v tramvaji $S(m)$ nalezneme jako rozdíl počtu pasažérů, kteří do tramvaje nastoupili, $S_+(m)$ a počtu pasažérů, kteří z tramvaje vystoupili, $S_-(m)$.

Příspěvek $S_+(m)$ je zjevně součtem přistoupivších pasažérů přes všechny stanice $1, \ldots, m$. U každé z nich pro každého na ní vygenerovaného pasažéra existuje $L := n_1 - n_0 + 1$ možných cílů, při jejichž výběru se rozhodne nastoupit. V každé stanici tedy přibude DL/N pasažérů.

Jedinou výjimkou, kdy pasažérů přibude méně, bude, pokud část jejich intervalu povolených cílů leží už za konečnou stanicí linky. Opravu na tyto případy započítáme tak, že od předchozího odhadu mDL/N odečteme pasažéry, kteří nepřistoupili. Jejich zdrojů je:

$$\mathsf{F}(1 + n_0, 1 + n_1) + \cdots + \mathsf{F}(m + n_0, m + n_1) =: \mathsf{G}(m, n_0, n_1)$$

Potom lze počet přistoupivších zapsat jako:

$$S_+(m) = [m - \mathsf{G}(m, n_0, n_1)]\,\frac{DL}{N}$$

Velmi podobně vyjádříme příspěvek $S_-(m)$. Do stanice k mířilo D/N pasažérů z každé stanice z intervalu $k - n_1, \ldots, k - n_0$. To znovu činí celkově mDL/N dosud vystoupivších. Zbývá jen provést opravu na situace, kdy stanice s číslem $k-n_1, \ldots, k-n_0$ leží ještě před výchozí stanicí linky. To provedeme odečtením:

$$\mathsf{F}(2 - n_1, 1 - n_0) + \cdots + \mathsf{F}(m - n_0, m - n_1) =: \mathsf{H}(m, n_0, n_1)$$

[10] Je časté, že pokud se něco definuje, používá se k tomu symbol := místo obyčejného rovnítka. My se tohoto zvyku budeme držet.

Příspěvek vystoupivších pasažérů tedy činí:

$$S_-(m) = [m - \mathsf{H}(m, n_0, n_1)]\,\frac{DL}{N}$$

Jako konečné vyjádření počtu pasažérů uvnitř tramvaje potom dostáváme:

$$S(m) = S_+(m) - S_-(m) = [\mathsf{H}(m, n_0, n_1) - \mathsf{G}(m, n_0, n_1)]\,\frac{DL}{N}$$

K explicitnímu vyjádření se dá dopracovat teprve při daných číselných hodnotách m, N, n_0, n_1.

Řešení 5. série

Problém 1

Zadání:
Linka B pražského metra má 24 stanic. Evžen nastoupil v první stanici, ve které se zaplnilo K procent z kapacity metra, a jede až na konečnou. Evžen nechce sedět, pokud by to mělo znamenat, že na někoho jiného nezbude místo k sezení, zároveň mu je ale kontakt s cizími lidmi natolik nepříjemný, že bude raději stát celou cestu, než aby si sedl a někomu pak případně své místo nabídl.

Při jakých hodnotách C, K si může Evžen hned po svém nástupu sednout? Předpokládejme, že nám není známa vytíženost jednotlivých stanic, proto uvažujeme stanice rovnocenné.

Řešení:
Při tvaru křivky $S(m) = Cm(N - m)$ bude metro nejzaplněnější ve stanici $m = N/2 = 12$. Podmínka v zadání bude splněna právě tehdy, pokud v této stanici bude uvnitř metra stále méně lidí, než je kapacita míst k sezení, o níž víme, že je $S(1)/K$. Dostáváme nerovnost:

$$S\left(\frac{N}{2}\right) \le \frac{S(1)}{K}$$

$$C\frac{N}{2}\frac{N}{2} \le \frac{C}{K}(N-1)$$

$$K \le \frac{4}{N^2}(N-1) \approx 0{,}16$$

Dostáváme, že Evžen si může v metru sednout právě tehdy, pokud je kapacita míst k sezení zaplněna nejvýš ze zhruba 16 %. Na hodnotě konstanty C nezáleží, tj. tento výsledek platí v jakoukoli denní hodinu i pro jakoukoli jinou linku hromadné dopravy, která má 24 stanic. □

Problém 2

Zadání:

Náš model předpokládá, že je na každé stanici vygenerován nějaký náhodný počet pasažérů. Dosud jsme uvažovali, že je nám známa střední hodnota tohoto počtu D, konkrétní povahou generování pasažérů jsme se však nezabývali. Cílem této úlohy je zamyslet se nad rozložením pravděpodobnosti pro počet vygenerovaných pasažérů.

Předpokládejme, že každá stanice linky poskytuje jediné dopravní spojení právě jednomu sídlišti, kde bydlí M lidí (M je v řádech stovek až tisíců), a každý člověk zde bydlící se může rozhodnout s pravděpodobností p někam jet (tj. být „vygenerován“ jako pasažér) nebo $1-p$ zůstat na sídlišti.

1. *Vypočítejte v takovém modelu pravděpodobnost $P_{M,p}(n)$, že bude v dané stanici vygenerováno n pasažérů při pevně zadaných parametrech M, p. Nedaří-li se vám nalézt přesný výsledek, stačí alespoň dobrá aproximace.*

2. *Vypočítejte při známých parametrech M, p pro počet vygenerovaných pasažérů střední hodnotu D a střední kvadratickou odchylku σ.*

Řešení:

1. Existuje užitečný a často využívaný matematický model z teorie pravděpodobnosti, zvaný Bernoulliho pokusy[11]. Ten se zabývá posloupností n nezávislých a identicky rozdělených pokusů, z nichž každý může nabývat právě jednoho ze dvou možných výsledků a to buďto úspěchu (s pravděpodobností p) nebo neúspěchu (s pravděpodobností $1-p$). Je známo, že pravděpodobnost, že došlo ke k úspěchům, je rovna:

$$p_k = \binom{n}{k} p^k (1-p)^k$$

Tento model se skvěle hodí na naši situaci. Každý člověk představuje náhodný experiment ve smyslu rozhodování se, zda někam pojede či nikoli. Jeho výsledek bude odpovídat úspěchu, rozhodne-li se daný člověk někam jet, a neúspěchu v opačném případě. Tím pádem přímo dostáváme, že na stanici se zadanými parametry M, p bude vygenerováno n pasažérů s pravděpodobností

$$P_{M,p}(n) = \binom{M}{n} p^k (1-p)^k.$$

[11] více se o tomto schématu můžete dočíst např. na `https://sk.wikipedia.org/wiki/Bernoulliho_sch%C3%A9ma`

Aproximativního výsledku bylo možné dosáhnout užitím centrální limitní věty[12]. Ta říká, že máme-li n nezávislých a identicky rozdělených náhodných veličin, pravděpodobnostní rozdělení jejich součtu se pro velká n blíží Gaussovu pravděpodobnostnímu rozdělení. Ztotožníme-li proces rozhodování každého potenciálního pasažéra s náhodnou veličinou, která nabývá hodnoty 1 v případě „úspěchu“ a 0 v případě „neúspěchu“, součet všech těchto veličin bude udávat celkový počet vygenerovaných pasažérů a zároveň bude gaussovsky rozložen.

Zbývá pro něj odvodit střední hodnotu a varianci. Dosazením do definice těchto veličin snadno dostáváme pro každého jednotlivého pasažéra střední hodnotu $\mu = p$ a varianci $V = p(1-p)$. Z aditivity střední hodnoty a variance a z tvaru Gaussova rozdělení dostáváme aproximativní vyjádření:

$$P_{M,p}(n) \approx \frac{1}{\sqrt{2\pi Mp(1-p)}} e^{-\frac{(n-Mp)^2}{2Mp(1-p)}}$$

Výše uvedené přesné řešení se mu blíží pro velké M. Při vysokých hodnotách M je tedy vhodnější užít vyjádření aproximativního, které je výrazně snazší pro výpočet číselné hodnoty. □

2. Tento úkol jsme splnili již výše při odvozování aproximativního tvaru $P_{M,p}(n)$. Střední hodnota a střední kvadratická odchylka vygenerovaného počtu lidí jsou rovny:

$$D := \mu = Mp$$

$$\sigma = \sqrt{Mp(1-p)}$$

□

Problém 4

Zadání:
Bez nutnosti výpočtu se zamyslete nad následujícími problémy:

1. *Může se stát, že by měla křivka $S(m)$ více než jedno lokální maximum? Za jakých podmínek?*

2. *Co se stane, když jsou na lince dvě stanice jistým způsobem propojené a lidé mezi nimi jezdí častěji, ačkoli se taková dvojice z hlediska ostatních stanic nechová nijak výrazně (např. u jedné stanice stojí Matfyz a u druhé koleje). Dala by se tomuto případu nějak přizpůsobit naše teorie?*

3. *Lze užívaný popis křivky $S(m)$ při velmi vysokých hodnotách N nějak upravit/zjednodušit?*

[12] viz příloha o Teorii pravděpodobnosti, 2. série tohoto ročníku: `https://mam.mff.cuni.cz/media/cislo/pdf/25/25-2.pdf`

Řešení:

1. Za dosud uvažovaných zjednodušených podmínek se to stát nemůže, žádné z dosud zavedených zobecnění takový jev nedovoluje. V reálné hromadné dopravě to však nastat může, proto je třeba v tomto směru naši teorii dále zobecnit. Jedno z možných takových zobecnění nastíní hned další bod.

2. Je-li mezi jistou dvojicí stanic větší tok pasažérů než mezi ostatními dvojicemi, pochopitelně se to projeví zvýšením hodnoty $S(m)$ mezi těmito dvěma stanicemi. Předpokládáme-li navíc, že dané propojené stanice jsou rovnocenné ostatním stanicím, co se týče produkce pasažérů, dá se navíc předpokládat snížení $S(m)$ na zbytku linky. Propojené stanice totiž přispívají do zbytku linky méně pasažéry, než kdyby propojené nebyly. Rozmístíme-li na linku vhodným způsobem vícero dvojic propojených stanic, na křivce $S(m)$ se to může projevit právě výskytem vícero lokálních maxim, viz bod 1.

   Uvažme extrémní případ, kdy každá ze dvou propojených stanic vygeneruje D pasažérů a ty pak všechny pošle té druhé, mezitímco každá ze zbylých stanic svých D vygenerovaných pasažérů rozdělí mezi stanice rovnoměrně. Každá z nepropojených stanic by tím pádem dostala D/N pasažérů od každé stanice krom dvou propojených, dohromady $D(N-2)/N$, tedy méně, než kolik vygenerovala. Oproti tomu každá ze dvou propojených stanic dostane pasažérů $D + D(N-2)/N$, tj. skoro dvojnásobek toho, co vygenerovala.

   V našem matematickém modelu z toho žádný spor neplyne, na reálné lince by to však intuitivně znamenalo, že pasažéři driftují do dvojice propojených stanic, které je budou nenávratně pohlcovat. Podobný efekt bychom pozorovali i mimo uvedený extrémní případ, prakticky u jakéhokoli netriviálního propojení stanic. Model linky, který umožňuje takto stanice propojit, nemůže proto vystihovat žádnou reálnou linku dlouhodobě. Vyžadoval by nejprve další zobecnění, například zahrnutí časové závislosti propojení mezi stanicemi nebo, ještě lépe, zahrnutí podmínky na zachování celkového počtu pasažérů. Druhá metoda by mohla využít k popisu linky tzv. Ehrenfestův model, známý artefakt ze statistické fyziky, jímž už se však my zabývat nebudeme.

   Co je však zajímavé, že právě tímto modelem se velmi úspěšně zabývá Tomáš Flídr v článku zde otištěném. Postuluje nikoli znalost vytíženosti stanic, jako jsme to dělali my, nýbrž znalost vytíženosti dvojic stanic a dále řeší úlohu hledání křivky $S(m)$. Pro rovnocenné dvojice dospívá ke stejnému výsledku, jaký jsme získali my pro rovnocenné stanice. Pro nerovnocenné stanice se však jeho teorie s naší dle očekávání rozchází. Objevuje se tím nová příležitost ke zkoumání, který matematický model vystihuje pozorovanou skutečnost lépe. Zadána tato úloha nebude pouze proto, že je toto poslední série tohoto ročníku.

Evžen; JanSkvara@email.cz
e-mailová konference: tramvaj@mam.mff.cuni.cz

Téma 6 – Vrcholové pokrytí

Řešení 3. 4. a 5. série

První dva grafy jsou dost malé na to, aby bylo možné jim najít minimální vrcholové pokrytí „ručně“, nebo hrubou silou na počítači.

Řešení pro graf 1:
2,3,5,7,9

Řešení pro graf 2:
0,2,4,5,7,9

Graf 3 je strom. (Konkrétně každý vrchol kromě nuly má právě jednu hranu do vrcholu s nižším číslem.) Pro stromy můžeme najít optimální řešení tím, že si všimneme, že chceme vybírat vrcholy ve vzdálenosti jedna od listu.

Řešení pro graf 3:
0,2,3,4,6,8,12,13

Graf 4 je bipartitní. (Konkrétně každá hrana spojuje vrchol s lichým číslem a vrchol se sudým číslem.) Všimneme si, že existuje osm hran nesdílejících žádný vrchol – například (0, 7), (2, 5), (4, 1), (6, 15), (8, 11), (10, 3), (12, 9), (14, 13). Vrcholové pokrytí tudíž musí být alespoň velikosti osm, čehož se dá docílit třeba vybráním všech vrcholů se sudým číslem[13].

Graf 5 není souvislý. Rozdělíme si jej tedy na komponenty souvislosti a vyřešíme každou komponentu zvlášť.

Grafy 6 a 7 jsou opět stromy, jen mnohem větší. Graf 8 má tu speciální vlastnost, že každá hrana vede z jednoho z vrcholů 0,1,2,3,4,5,6, což nám okamžitě dává pokrytí velikosti 7, o kterém je taky možné dokázat, že je optimální.

Graf 9 byl generován náhodně a nalezení jeho minimálního pokrytí vyžadovalo hlavně výpočetní výkon. Vyzkoušet všech $2^{20} = 1048576$ možných podmnožin vrcholů a najít nejmenší, která je pokrytím, je sice neefektivní, ale jednoduchý a na dnešních strojích pořád použitelný algoritmus. Všimněte si, že formát popisu grafu znamená, že můžeme testovat, zda každý řádek (kromě prvního) obsahuje některý z námi vybraných vrcholů.

Způsob, kterým byl generován graf 10, byl poněkud složitější, ale není náhodný. Pokud se vám z něj podaří něco vykoukat, tak vám za to dám ještě nějaké body.

Matej; `lieskovsky.matej+pokryti@gmail.com`
e-mailová konference: `pokryti@mam.mff.cuni.cz`

[13] Tento postup je inspirován Königovou větou, jejíž užitečnost je zmíněna na již odkazované stránce anglické Wikipedie `https://en.wikipedia.org/wiki/Vertex_cover`

Články

Přeplněná tramvaj – návrh teoretického modelu (13 b)

Mgr.MM Tomáš Flídr

Abstrakt

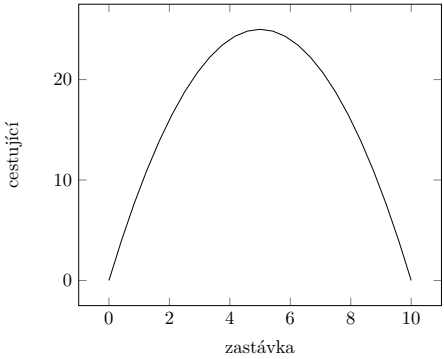
V tomto článku se teoreticky zabývám odhadem vývoje počtu cestujících na linkách bez konkrétního směru a v dlouhodobém průměru dní i časů. Používám několik metod, které mají za cíl postupně zpřesnit výsledný odhad.

Úvod

Bohužel jsem neměl dostatek experimentálních dat, takže jsem situaci popisoval čistě teoreticky. Pro začátek uvažujme pouze dlouhodobý průměr, tedy neorientovanou trasu bez konkrétního času (cestující se zpravidla vrací i zpět). V tom případě můžeme problém převést na „Čemu je úměrný počet lidí jedoucích mezi místy/městy A a B?“ (Resp. n_1 a n_2.)

Rovnoměrné rozložení

V nejtriviálnějším případě mezi každou dvojicí stanic jede stejný počet lidí. Po vydělení konstantou tohoto počtu zjistíme, že na k-té zastávce z n při číslování od 1 vystoupí $k-1$ lidí (předchozí zastávky) a nastoupí $n-k$ (následující zastávky) lidí, počet lidí se tedy zvýšil o $n-k-k+1 = n-2k+1$ (resp. sníží, je-li toto číslo záporné). Z toho vyplývá, že po l-té zastávce je uvnitř $\sum_{k=1}^{l} n-2k+1 = l\cdot(n+1) - 2\sum_{k=1}^{l} k = nl + l - 2\cdot\frac{l(l+1)}{2} = nl + l - l^2 - l = nl - l^2$ osob. Vývoj počtu cestujících pro 10 zastávek zachycuje obrázek 17.



Obrázek 17: Vývoj počtu cestujících mezi stanicemi při rovnoměrném rozložení

Přerov	Kroměříž	Brno
43 791	29 002	379 527

Tabulka 7

Jedná se o kvadratickou funkci, tedy parabolu, takže nejvyšší počet cestujících je uprostřed – ve vrcholu paraboly. Všimněme si, že na začátku i na konci je tramvaj podle očekávání prázdná. Bohužel zastávky běžně nejsou rovnocenné, takže musíme zavést nějaký jiný model.

Součet obyvatel

Pokud zanedbáme ostatní spoje, popřípadě řekneme, že jsou rozložené úměrně počtu obyvatel, můžeme říct, že počet pasažérů mezi dvěma městy záleží na velikosti těchto měst, resp. na počtu lidí, kteří z nich mohou cestovat – součtu počtů obyvatel daného města (pro n-té město o_n). Pokud mezi dvěma městy cestuje $o_n + o_k$ lidí, můžeme říci, že $o_n + o_k$ nastoupí na n-té zastávce a vystoupí na k-té. Na n-té zastávce z l tedy nastoupí $\sum_{k=n+1}^{l}(o_n + o_k)$ lidí a vystoupí jich $\sum_{k=1}^{n-1}(o_n + o_k)$. Celkový počet se zvýší (resp. sníží při záporném čísle) o

$$\sum_{k=n+1}^{l}(o_n + o_k) - \sum_{k=1}^{n-1}(o_n + o_k) =$$

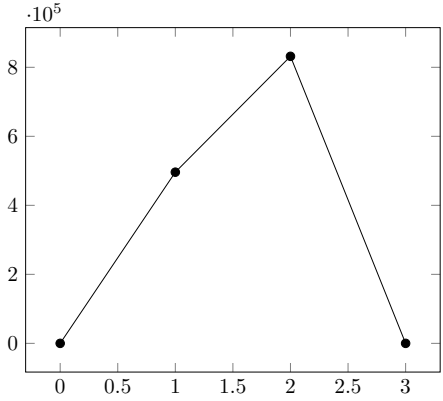
$$= o_n(l - n - n + 1) + \sum_{k=n+1}^{l} o_k - \sum_{k=1}^{n-1} o_k =$$

$$= o_n(l - 2n + 1) + \sum_{k=1}^{l} o_k - o_n - 2\sum_{k=1}^{n} o_k =$$

$$= o_n(l - 2n) + \sum_{k=1}^{l} o_k - 2\sum_{k=1}^{n-1} o_k.$$

Přitom $\sum_{k=1}^{l} o_k$ je konstantní, protože se jedná o součet obyvatel všech měst na trase, můžeme jej tedy nahradit zápisem O. Dostáváme tedy rekurzivní funkci, kde se počet cestujících po n-té zastávce změní o $o_n(l - 2n) + O - 2\sum_{k=1}^{n-1} o_k$, přičemž poslední sumace je zřejmě počet obyvatel měst před n-tým. Vzhledem k tomu, že používáme konstantní velikosti měst, bude jednodušší, než vystihnout tuto rekurzivní funkci předpisem, ukázat příklad pomocí počítačově dopočítaných dat. Pro tento příklad jsem vybral trasu Přerov – Kroměříž – Brno. Vstupní data jsou zapsána v Tabulce 7 (údaje o počtech obyvatel jsou z Wikipedie).

Strojovým zpracováním těchto dat s využitím rekurzivní funkce se startem v bodě 0 dostaneme počty 496091 a 831807, jak je graficky znázorněno v obrázku 18. Před první a po poslední zastávce nám model samozřejmě předpovídá 0 cestujících.



Obrázek 18: Počty cestujících mezi zastávkami Přerov, Kroměříž a Brno – počítáno podle součtu obyvatel

Vidíme, že v tomto případě nebyl předchozí způsob tak daleko od tohoto, ale určité rozdíly tady jsou.

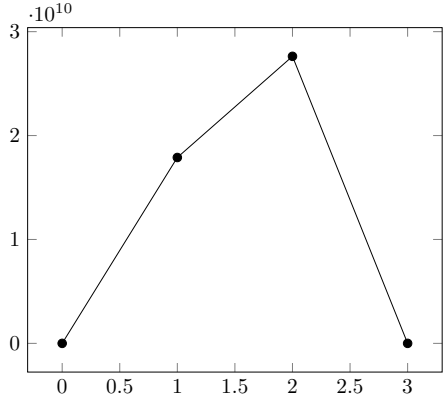
Důležitost

Dalším způsobem, jak odhadnout rozložení, je využití důležitosti města – čím důležitější město, tím více lidí do něj bude cestovat. Pro účely tohoto článku zjednodušme důležitost na veličinu přímo úměrnou počtu obyvatel – většinou se někam jezdí za někým. Z toho vyplývá, že pokud si počty obyvatel dvou měst označíme o_n, resp. o_k, bude zde cestovat $o_n \cdot o_k$ lidí z města n a obdobně $o_k \cdot o_n$ lidí z města k, celkem tedy $2o_no_k$. Pořád roznásobujeme konstantou, takže pro účely tvaru křivky můžeme počítat s tím, že mezi těmito městy bude cestovat o_no_k pasažérů.

V n-tém městě z l tedy nastoupí $\sum_{k=n+1}^{l} o_ko_n$ lidí a vystoupí $\sum_{k=1}^{n-1} o_no_k$, celkem se počet pasažérů změní o $\sum_{k=n+1}^{l} o_ko_n - \sum_{k=1}^{n-1} o_no_k = o_n(\sum_{k=n+1}^{l} o_k) - \sum_{k=1}^{n-1} o_k = o_n(O - o_n - 2\sum_{k=1}^{n-1} o_k)$. Pokud takto získanou rekurzivní funkci použijeme na výše zmíněná data, dostaneme čísla obrovských řádů, jak je vidět z obrázkuu 19. Kromě nulových krajních bodů jsou to postupně čísla 17889017619 a 27625453051.

Můžeme si všimnout, že u těchto dat se grafy téměř neliší, hlavně z toho důvodu, že města jsou řádově podobně velká. Větší rozdíl je vidět na datech z trati Kroměříž – Kojetín. Jak je vidět z Tabulky 8, jedná se o vlak mezi dvěma malými městy, z nichž jedno je znatelně větší, který projíždí přes dvě vesnice. Křivky zachycující počet cestujících mezi stanicemi na této trati počítané podle počtu obyvatel a podle důležitosti můžete vidět v obrázcích 20 a 21.

Jelikož se jedná o spoj, se kterým mám určité zkušenosti, můžu potvrdit, že graf na obrázku 21 je podstatně přesnější. V Postoupkách a Bezměrově téměř



Obrázek 19: Počty cestujících mezi zastávkami Přerov, Kroměříž a Brno – počítáno podle důležitosti

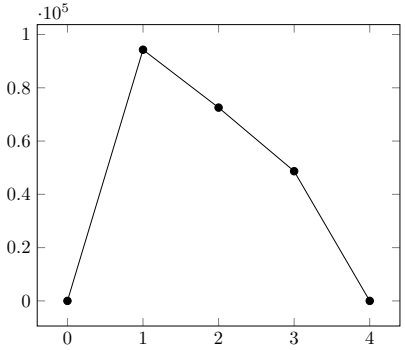
Kroměříž	Postoupky	Bezměrov	Kojetín
29002	558	520	6200

Tabulka 8

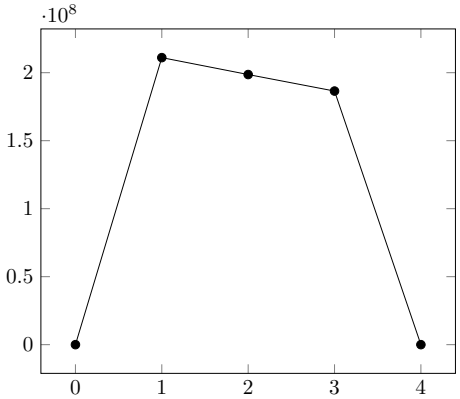
nikdo nevystupuje a nenastupuje – co by tam taky kdo dělal.

Závěr

Pravděpodobně nejpřesnější metodou je počítání s důležitostí daného města pomocí rekurzivní funkce, pro nedostatek experimentálních dat bylo tuto domněnku ale nemožné spolehlivě potvrdit. Důležitost města navíc samozřejmě ovlivňují i další faktory a úměra nemusí být nutně přímá – uvažoval jsem i o exponenciální podobě, kterou vidíme například u Prahy – v důsledku pragocentrismu je více než dvakrát důležitější než Brno. Dále třeba vzdálenost – častěji se jezdí kratší úseky než delší. Samostatnou kapitolu potom tvoří orientované trasy a konkrétní časy, můžeme tedy vidět, že faktorů rychle přibývá.



Obrázek 20: Počty cestujících na lince z Kroměříže do Kojetína – počítáno podle součtu obyvatel



Obrázek 21: Počty cestujících na lince z Kroměříže do Kojetína – počítáno podle důležitosti

Lichtenbergovy figury (14 b)

Mgr.MM Martin Zimen

Šíření blesku na obloze nebo výbojů Teslova transformátoru je nádherné. Elektrický výboj, který proniká dielektrikem, má své charakteristické větvící se struktury. Těmto skoro fraktálovitým strukturám se říká *Lichtenbergovy figury* (dále jen LF). Kromě toho, že jsou LF nádherné, má také výzkum těchto struktur praktický význam. Umožňují nám znázornit a lépe pochopit šíření elektrických výbojů. Díky tomu jsme schopni se před nimi lépe chránit. V mém příspěvku jsem se pokusil o replikaci LF ve dřevě v domácích podmínkách. Chtěl bych prozkoumal, které faktory ovlivňují vznik LF, a jak je nejlépe nastavit, aby došlo k jejich tvorbě.

Cíle mého bádání

Vytvořit LF ve dřevě je teoreticky poměrně jednoduché – naneseme na dřevo nějaký roztok soli a připojíme zdroj vysokého napětí. Z přiložených elektrod začnou vyrůstat LF. Prakticky je to složitější, protože faktorů, které ovlivňují vznik LF je hodně:

- napětí,
- vzdálenost elektrod,
- velikost plochy s naneseným roztokem,
- doba působení el. proudu,
- složení roztoku,
- koncentrace roztoku,
- množství naneseného roztoku,
- doba vsakování roztoku,
- druh dřeva,
- struktura dřeva.

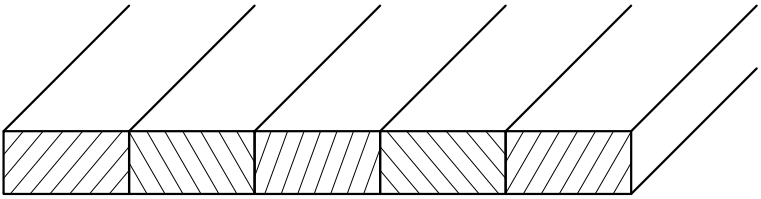
V tomto článku nejsem schopný prozkoumat vliv všech faktorů. Proto vyberu pouze některé a zbylé nebudu měnit. Pro všechny experimenty tedy budu používat stejné napětí a druh dřeva. Pokud by někdo chtěl na můj příspěvek navázat, bylo by zajímavé zkusit vytvářet LF s variabilním zdrojem napětí a různými druhy dřeva. V tomto příspěvku bych rád prozkoumal vliv ostatních faktorů.

Materiály

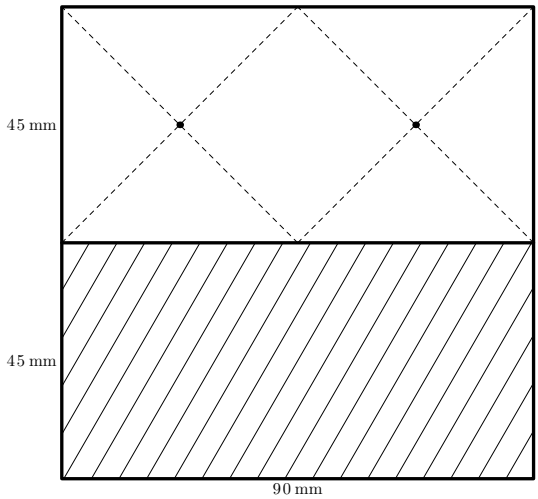
Hned první věc, kterou potřebujeme, je nějaký zdroj vysokého napětí. Zde jsem dlouho neváhal a vybral MOT (Microwave Oven Transformer, transformátor z mikrovlnky). MOT je poměrně dostupný zdroj, který svůj účel splní dokonale. Také se může hodit na další experimenty s vysokým napětím. Jak už název napovídá, takový transformátor nejlépe seženeme z nějaké staré mikrovlnky. Zapojení je velmi jednoduché – primární cívku připojíme na 230 V a ze sekundární cívky lze odebírat zhruba 2 000 V. Sekundární cívku tedy můžeme rovnou připojit do dřeva. Zde je však třeba nejvyšší opatrnosti – 2 000 V je nebezpečně vysoké napětí. Proto je potřeba se důkladně odizolovat. Jedna varianta je vše zapojit předem a zapnout MOT z bezpečné vzdálenosti a s obvodem nemanipulovat. Druhá varianta je vyrobit si násadu, která elektrody odizoluje a umožní manipulaci. Tato varianta se ukázala jako mnohem praktičtější.

Další věc, kterou potřebujeme, je dřevo. Rozhodl jsem se pro smrkovou spárovku. Hlavní motivací byla cena a dostupnost. Ukázalo se však, že spárovka byla skvělá volba. Jelikož je spárovka složená z několika dílů dřeva, mám možnost vyzkoušet mnoho různých vzorů na povrchu dřeva (Obrázek 22).

Dále je třeba nějaký roztok, který budu na dřevo nanášet. Rozhodl jsem se porovnat dva roztoky – roztok NaCl a roztok $NaHCO_3$. Oba dva roztoky jsem připravil rozpuštěním 12 g chemikálie ve 184 ml vody ($\omega = 0{,}061$).



Obrázek 22: Spárovka



Obrázek 23: Šablona

Jako poslední jsem připravil nanášecí fólii – chtěl jsem mít možnost nanášet stejnou vrstvu roztoku na stejnou plochu. Proto jsem na průhlednou fólii nakreslil dva obdélníky (Obrázek 23). Jeden jsem vystřihl (šrafovaný) a druhý sloužil k přesnému umístění elektrod. Šablonu jsem používal následovně: nejdříve jsem obtáhl obdélník na dřevo tužkou, uprostřed jsem udělal šídlem díry (podle druhého obdélníku) a poté přetřel fólii houbičkou s roztokem. Snažil jsem se tím docílit rovnoměrného nánosu roztoku na stejnou plochu.

První série měření

Jako první jsem se rozhodl testovat vliv koncentrace roztoků. Proto jsem vytvořil řadu tří roztoků $NaHCO_3$ naředěním původního roztoku s hmotnostními zlomky $\omega_A = 0{,}061$, $\omega_B = 0{,}031$ a $\omega_C = 0{,}16$. V první sérii jsem se také rozhodl testovat čistou vodu a NaCl. Nanášel jsem přes fólii houbičkou. Nanesl jsem tři vrstvy. Druhou a třetí vždy minutu po předešlé. Pět minut po nanesení první vrstvy jsem zapojil elektrody na vyznačená místa. Proud jsem nechal téct tak dlouho, dokud docházelo k viditelným změnám. Každé měření jsem provedl třikrát.

Zapisoval jsem si průběh každého měření. Pozoroval jsem, k jak silnému docházelo výboji v místě dotyku elektrod a jestli docházelo k tvorbě LF. Důležité je rozlišovat mezi LF a spálenými místy vlivem výboje (V). Výboj vytvoří nahnědlý kruh o poloměru zhruba půl centimetru (viz Obrázek 24). LF vytváří zřetelné černé čáry (viz Obrázek 25). Výsledky viz Tabulka 9.

Obrázek 24: Výboj

Obrázek 25: H_2O 2. měření

Je také třeba zmínit, že jsem přiložil elektrody k suchému, ničím nepotřenému dřevu a také ke dřevu, které bylo natřené roztokem,který již stihl úplně vyschnout. Ani v jednom případě nedošlo k tvorbě LF.

Druhá série měření

Jelikož byly LF z první série neuspokojivé, rozhodl jsem se změnit postup. Namísto přesného obdélníku a přesného umístění elektrod jsem se rozhodl sbírat více dat v rámci jednoho měření. Natíranou plochu jsem zvětšil na 11 cm × 6 cm. Roztok se ale většinou rozpil na větší plochu, proto je třeba brát tyto hodnoty orientačně. V rámci jednoho měření jsem také neumisťoval elektrody na stabilní místo, ale rozhodl jsem se s umístěním elektrod hýbat. V rámci druhé série měření jsem se soustředil hlavně na způsob nánosu a dobu schnutí. Také jsem chtěl udělat více měření roztoku NaCl.

Roztok	1. měření	2. měření	3. měření
H_2O	slabý V bez LF —	slabý V s LF $r = 1{,}7\,\mathrm{cm}$	slabý V bez LF —
$NaHCO_3A$	střední V s LF $r = 2\,\mathrm{cm}$	střední V s LF $r = 1\,\mathrm{cm}$	střední V bez LF —
$NaHCO_3B$	střední V bez LF —	střední V s LF $r = 2\,\mathrm{cm}$	střední V s LF $r = 1\,\mathrm{cm}$
$NaHCO_3C$	střední V s LF $r = 2{,}5\,\mathrm{cm}$	střední V s LF $r = 0{,}2\,\mathrm{cm}$	střední V s LF $r = 0{,}2\,\mathrm{cm}$
NaCl	silný V bez LF —	silný V bez LF —	silný V bez LF —

Tabulka 9: Výsledky prvního měření

Rozhodl jsem se, že pro každý roztok vyzkouším 5 různých dob vsakování. Každý roztok jsem nechal vsakovat t_c, přičemž t_s od nanesení kapaliny jsem přebytečný roztok ze dřeva setřel kapesníkem. Časy shrnuje Tabulka 10.

Měření	t_c	t_s
1.	5 min	∞
2.	5 min	5 min
3.	10 min	7 min
4.	17 min	7 min
5.	22 min	7 min

Tabulka 10: Výsledky měření

Jelikož sbírám v rámci jednoho měření více dat, rozhodl jsem se snížit celkový počet měření. Každé jsem tedy dělal pouze dvakrát. V rámci každého měření jsem přikládal elektrody, než došlo k většímu výboji, a pozoroval. Tentokrát jsem nic nezapisoval, ale srovnával konečné obrazce (viz příloha na konci).

Výsledky měření

Provedl jsem celkem 35 měření a přes 100 přiložení elektrod. Rád bych vám zde prezentoval pozorování z těchto měření. Poté bych se pokusil tato pozorování nějak fyzikálně interpretoval. Rád bych zde zmínil, že jsem otevřený jiným interpretacím. Budu rád, když se čtenář pokusí vysvětlit, proč k daným jevům dochází, sám.

Pozorování. Struktura dřeva je nepředvídatelná. Někdy stačí posunout elektrodu o malý kousíček a zcela to změní tvorbu LF.

To je hlavní důvod, proč jsem změnil způsob provádění experimentů mezi sériemi.

Pozorování. K tvorbě LF může dojít i při přetření dřeva čistou vodou (viz 2. měření H_2O).

Z tohoto pozorování vidíme, že pro vznik LF není roztok zásadní. Jak uvidíme později, roztok má svůj význam, ale není esenciální. Vzhledem k tomu, že LF nevznikají na suchém dřevě, musí být voda přítomna, aby zvýšila vodivost dřeva. Používal jsem pitnou vodu, která vede elektrický proud. Použití destilované vody by podle mě bylo bezpředmětné, protože se do destilované vody rychle rozpustí látky ze dřeva, které umožní vodivost. Nicméně by to bylo zajímavé vyzkoušet.

Pozorování. Roztoky $NaHCO_3$ a NaCl způsobují výboj při přiložení elektrod (viz první série měření).

Velikost výboje je ovlivněna vodivostí materiálu. Náš obvod se skládá ze čtyř prvků zapojených sériově – zdroje, materiálu a dvou přechodů mezi materiálem a elektrodou. Se zvýšením vodivosti materiálu na něm klesne napětí, tudíž se zvýší napětí mezi materiálem a elektrodou. Takže dojde k silnějšímu výboji.

Pozorování. Pokud dochází k příliš silnému nebo příliš slabému výboji, nevytváří se LF.

To jde ruku v ruce s výboji u elektrod. Výboje u elektrod indikují velké napětí mezi elektrodami a materiálem. Takže na samotném materiálu napětí klesne. Proto se LF nevytváří.

To lze vidět nejlépe z fotek v druhé sérii měření. Pokud dochází k silnému výboji, vznikají charakteristické spáleniny od výboje (viz Obrázek 24) a LF se nevytváří. Pokud dochází ke střednímu výboji, vznikají malinkaté LF (viz měření roztoků $NaHCO_3$ v první sérii). Pokud dochází k příliš slabému výboji (měření H_2O), tak k tvorbě LF nedochází (teče příliš malý proud).

Pro tvorbu LF je tedy třeba dosáhnout určitého napětí na materiálu.

Pozorování. Množství roztoku a doba vsakování ovlivňuje vznik LF. Pokud je roztoku na dřevě hodně, dochází k větším výbojům, a tudíž se nevytváří LF nebo se tvoří velké spálené oblasti, jejichž struktura LF připomíná spíše vzdáleně. Množství roztoku ovlivňuje rychlost tvorby a velikost LF.

Zde vycházím z druhé série měření. Pokud je roztoku na dřevě hodně, dochází k silným výbojům a spálení velkých oblastí. Pokud je roztoku na dřevě málo, dochází ke vzniku menších, ale detailnějších LF. Z tohoto důvodu ve většině měření v první sérii LF nevznikly.

Pozorování. Vzdálenost elektrod nehraje velkou roli.

V druhé sérii měření jsem umisťoval elektrody do různých vzdáleností. V rámci jednoho měření nebyl rozdíl mezi umístěním elektrod. Pokud docházelo k výbojům, nehrála vzdálenost roli – LF se nevytvářely. Pokud docházelo k tvorbě LF, tvořily se, ať už byly elektrody vzdálené 4 cm nebo 12 cm. Při příliš malé vzdálenosti elektrod docházelo k obloukovému výboji mezi elektrodami vzduchem.

K tvorbě LF tedy nedocházelo.

Zavedu si měrné napětí na materiálu (U). Bude to napětí na jednotku délky. Pokud označím napětí zdroje V_z, vzdálenost elektrod d, rezistivitu materiálu ρ a odpor obvodu mimo materiál R, mohu vyjádřit měrné napětí jako

$$U(d) = \frac{\rho}{d\rho + R} V_z.$$

Jde tedy o hyperbolu posunutou doleva o R. V rozumném Δd se tedy U nezmění tolik a udrží se v intervalu vhodném pro tvorbu LF.

Je tedy třeba držet d v nějakém rozumném intervalu. Ten je ale dostatečně velký, takže pokud se pohybujeme řádově v centimetrech, nehraje vzdálenost velkou roli.

Pozorování. $NaHCO_3$ je pro tvorbu LF lepší než NaCl.

Podle tabulek je při $\omega = 0{,}061$ vodivost NaCl zhruba dvakrát větší než vodivost $NaHCO_3$. S tímto vysvětlením však nejsem zcela spokojený, protože naprosto ignoruje chemický vliv. Po vsáknutí $NaHCO_3$ totiž dřevo zežloutne, zatímco po vsáknutí NaCl dřevo barvu nezmění. Myslím, že tato změna barvy nějak souvisí s tvorbou LF. Jelikož však nejsem chemik, nechám tuto otázku otevřenou pro někoho povolanějšího.

Pozorování. LF mají tendenci kopírovat strukturu letokruhů.

Letokruhy jsou střídání se měkčích a tvrdších vrstev dřeva. Měkké dřevo snadněji prohoří. Proto se LF šíří ve směru letokruhů.

Pozorování. V první sérii měření nehrála koncentrace roztoku takový vliv na tvorbu LF. Dřevo bylo příliš mokré, aby docházelo k tvorbě LF.

Pozorování. Větší vodivost způsobuje větší rychlost tvoření LF. Pokud se LF tvoří pomaleji, jsou detailnější.

Větší vodivost způsobuje větší proud materiálem. Větší proud rychleji a více spálí okolní dřevo. Proto se LF šíří rychleji a jsou méně detailní.

Závěr

LF se mi povedlo v domácích podmínkách replikovat na dřevě pomocí MOT a roztoku $NaHCO_3$.

Provedl jsem přes 100 přiložení elektrod. Následně jsem analyzoval vzniklé obrazce a hledal zákonitosti. Ty jsem se poté pokusil nějak vysvětlit. Některé se mi však vysvětlit nepodařilo a nechávám je otevřené.

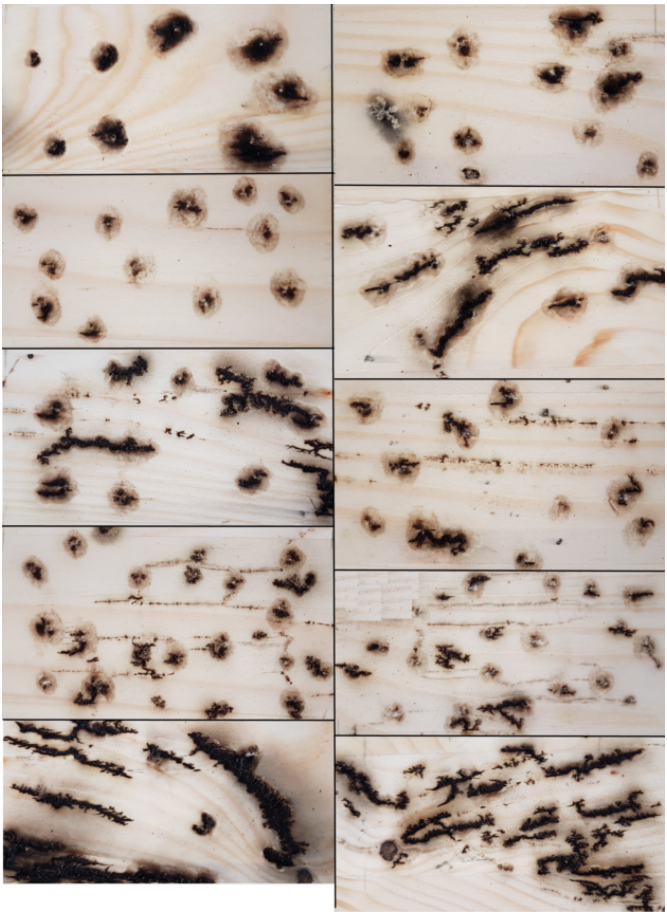
K tvorbě LF je důležité správné napětí na materiálu. Toho dosahujeme regulací vodivosti materiálu pomocí nanášení vodivého roztoku – např. vodným roztokem NaCl nebo $NaHCO_3$, který se ukázal jako lepší volba. Důležité je, aby nebyl materiál příliš mokrý – je třeba dosáhnout vhodného nasáknutí materiálu. Na vhodnou koncentraci roztoku se mi nepodařilo přijít kvůli tomu, že jsem nanášel příliš mnoho roztoku. Vzdálenost elektrod nehraje příliš velkou roli. LF mají tendenci kopírovat strukturu letokruhů dřeva.

Nezkoumal jsem vliv napětí zdroje a druhu dřeva. Tím by se dalo na moji práci navázat. Také by bylo zajímavé vyzkoušet jiné roztoky. Zajímavé by bylo prozkoumat vliv hustoty letokruhů a toho, jestli jsou elektrody umístěné podél letokruhů, nebo napříč. Vliv koncentrace roztoků by bylo potřeba změřit znovu.

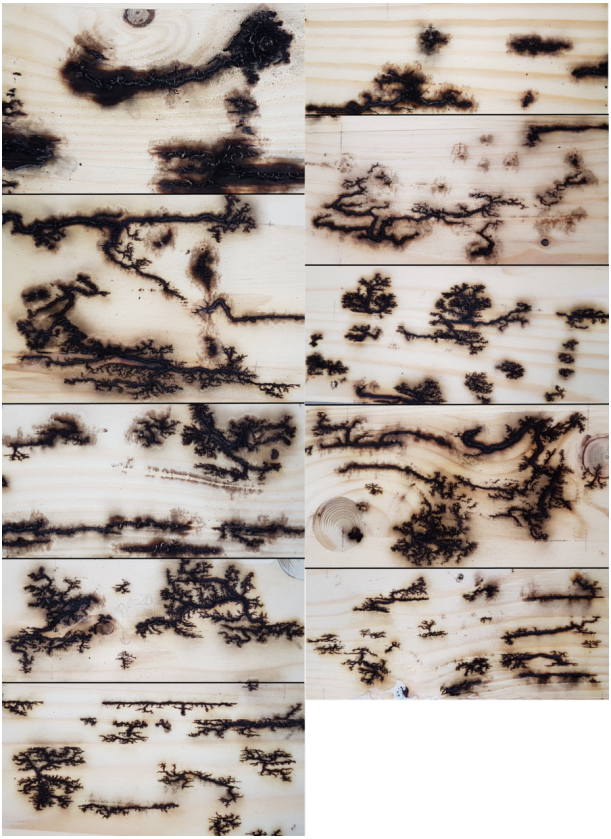
Přílohy

2. série

Zde naleznete všechna měření v druhé sérii. Každý obrázek se skládá z 10 fotek – fotek jednotlivých měření. První měření je nahoře, páté je dole. Každé měření jsem provedl dvakrát – dvě fotky vedle sebe.



Obrázek 26: NaCl



Obrázek 27: $NaHCO_3$

Výsledková listina 4. čísla

Poř.	Jméno	R.	$\sum_{-1}$	Úlohy t2	t4	t5	k	k	čl	$\sum_0$	$\sum_1$
1.	Mgr.MM T. Sourada	4	38,1	1,0						1,0	38,1
2.	Mgr.MM O. Chlubna	2	34,8							0	34,8
3.	Mgr.MM V. Materna	3	32,6	5,6						5,6	32,6
4.–5.	Mgr.MM T. Flídr	1	26,0			13,0				13,0	26,0
	Mgr.MM K. Hloušková	3	30,5			7,0				7,0	26,0
6.	Doc.MM K. Rosická	4	142,5			10,0				10,0	25,7
7.	Dr.MM M. Kalousková	3	50,3							0	24,8
8.	Mgr.MM M. Souza de Joode	2	40,3							0	20,5
9.	Bc.MM P. Kolář	3	17,0			5,0				5,0	17,0

Poř.	Jméno	R.	$\sum_{-1}$	Úlohy t2	t4	t5	k	k	čl	$\sum_0$	$\sum_1$
10.	Bc.[MM] B. Kopčák	3	17,5		1,5					1,5	14,5
11.	Mgr.[MM] M. Zimen	4	28,0						14,0	14,0	14,0
12.	Bc.[MM] J. Štěpo	Z9	13,5	4,0						4,0	13,5
13.–14.	Bc.[MM] J. Piroutek	3	13,2	5,5	1,0					6,5	13,2
	Mgr.[MM] J. Růžička	4	45,9			10,0				10,0	13,2
15.–17.	Doc.[MM] J. Havelka	3	104,4					6,0		6,0	13,0
	Bc.[MM] K. Kamenářová	4	15,7							0	13,0
	Mgr.[MM] J. Pallová	4	41,8				5,0			5,0	13,0
18.	Bc.[MM] K. Vokálová	3	12,4	5,4		7,0				12,4	12,4
19.–21.	Mgr.[MM] M. Boček	Z9	21,4							0	11,0
	Bc.[MM] A. Hollmannová	2	15,0					6,0		6,0	11,0
	Bc.[MM] M. Vícha	Z9	11,0							0	11,0
22.	Bc.[MM] E. Neumannová	2	10,1		8,0					8,0	10,1
23.–24.	Dr.[MM] K. Balej	4	89,4					6,0		6,0	9,0
	Dr.[MM] L. Kundratová	4	71,4							0	9,0
25.	F. Bujnovský	1	8,8							0	8,8
26.	M. Němec	3	8,5		8,5					8,5	8,5
27.	Mgr.[MM] O. Gonzor	2	29,9							0	8,2
28.	L. Vomelová	3	7,1		5,1					5,1	7,1
29.–30.	Bc.[MM] A. Foglarová	4	12,6							0	7,0
	Mgr.[MM] L. Kopfová	4	38,7							0	7,0
31.	Mgr.[MM] E. Vítková	3	31,3		3,0					3,0	5,8
32.	Dr.[MM] B. Hroncová	4	80,8							0	5,6
33.	V. Jůzková	2	5,5							0	5,5
34.	Dr.[MM] R. Olšák	4	53,4				5,0			5,0	5,0
35.	J. Kvapil	1	3,7							0	3,7
36.–37.	L. Kunčarová	3	3,6							0	3,6
	V. Žák	3	3,6		1,6					1,6	3,6
38.	J. Kováč	4	3,5							0	3,5
39.	Mgr.[MM] M. Holubička	3	37,8	3,0						3,0	3,0
40.	R. Zavřel	3	2,6							0	2,6
41.	N. Koscelanská	3	2,0							0	2,0
42.–43.	J. Přerovská	3	1,0							0	1,0
	M. Turinská	2	1,0							0	1,0

Sloupeček $\sum_{-1}$ je součet všech bodů získaných v našem semináři, $\sum_0$ je součet bodů v aktuální sérii a $\sum_1$ součet všech bodů v tomto ročníku. Tituly uvedené v předchozím textu slouží pouze pro účely M&M.

Výsledková listina 25. ročníku

Poř.	Jméno	R.	$\sum_{-1}$	Číslo 3	4	5	6	$\sum_{1}$
1.	Mgr.MM T. Sourada	4.	38,1	7,8	27,0	2,3	1,0	38,1
2.	Mgr.MM O. Chlubna	2.	34,8	4,8	19,5	10,5	0	34,8
3.	Mgr.MM V. Materna	3.	32,6	17,0	0	10,0	5,6	32,6
4.–5.	Mgr.MM T. Flídr	1.	26,0	13,0	0	0	13,0	26,0
	Mgr.MM K. Hloušková	3.	30,5	10,0	8,0	1,0	7,0	26,0
6.	Doc.MM K. Rosická	4.	142,5	3,0	3,0	9,7	10,0	25,7
7.	Dr.MM M. Kalousková	3.	50,3	4,8	7,5	12,5	0	24,8
8.	Mgr.MM M. Souza de Joode	2.	40,3	8,0	12,5	0	0	20,5
9.	Bc.MM P. Kolář	3.	17,0	0	0	12,0	5,0	17,0
10.	Bc.MM B. Kopčák	3.	17,5	0	0	13,0	1,5	14,5
11.	Mgr.MM M. Zimen	4.	28,0	0	0	0	14,0	14,0
12.	Bc.MM J. Štěpo	Z9.	13,5	5,0	4,5	0	4,0	13,5
13.–14.	Bc.MM J. Piroutek	3.	13,2	0	0	6,7	6,5	13,2
	Mgr.MM J. Růžička	4.	45,9	0	0	3,2	10,0	13,2
15.–17.	Doc.MM J. Havelka	3.	104,4	0	7,0	0	6,0	13,0
	Bc.MM K. Kamenářová	4.	15,7	0	8,0	5,0	0	13,0
	Mgr.MM J. Pallová	4.	41,8	8,0	0	0	5,0	13,0
18.	Bc.MM K. Vokálová	3.	12,4	0	0	0	12,4	12,4
19.–21.	Mgr.MM M. Boček	Z9.	21,4	0	11,0	0	0	11,0
	Bc.MM A. Hollmannová	2.	15,0	0	0	5,0	6,0	11,0
	Bc.MM M. Vícha	Z9.	11,0	0	11,0	0	0	11,0
22.	Bc.MM E. Neumannová	2.	10,1	0	0	2,1	8,0	10,1
23.–24.	Dr.MM K. Balej	4.	89,4	3,0	0	0	6,0	9,0
	Dr.MM L. Kundratová	4.	71,4	9,0	0	0	0	9,0
25.	F. Bujnovský	1.	8,8	8,8	0	0	0	8,8
26.	M. Němec	3.	8,5	0	0	0	8,5	8,5
27.	Mgr.MM O. Gonzor	2.	29,9	4,5	0	3,7	0	8,2
28.	L. Vomelová	3.	7,1	0	0	2,0	5,1	7,1
29.–30.	Bc.MM A. Foglarová	4.	12,6	0	7,0	0	0	7,0
	Mgr.MM L. Kopfová	4.	38,7	0	7,0	0	0	7,0
31.	Mgr.MM E. Vítková	3.	31,3	1,5	0	1,3	3,0	5,8
32.	Dr.MM B. Hroncová	4.	80,8	0	5,6	0	0	5,6
33.	V. Jůzková	2.	5,5	5,5	0	0	0	5,5
34.	Dr.MM R. Olšák	4.	53,4	0	0	0	5,0	5,0
35.	J. Kvapil	1.	3,7	3,7	0	0	0	3,7
36.–37.	L. Kunčarová	3.	3,6	3,6	0	0	0	3,6

Poř.	Jméno	R.	$\sum_{-1}$	Číslo 3	4	5	6	$\sum_{1}$
	V. Žák	3.	3,6	0	0	2,0	1,6	3,6
38.	J. Kováč	4.	3,5	3,5	0	0	0	3,5
39.	Mgr.^{MM} M. Holubička	3.	37,8	0	0	0	3,0	3,0
40.	R. Zavřel	3.	2,6	2,3	0	0,3	0	2,6
41.	N. Koscelanská	3.	2,0	0	2,0	0	0	2,0
42.–43.	J. Přerovská	3.	1,0	0	1,0	0	0	1,0
	M. Turinská	2.	1,0	0	0	1,0	0	1,0

Sloupeček $\sum_{-1}$ je součet všech bodů získaných v našem semináři, $\sum_{0}$ je součet bodů v aktuální sérii a $\sum_{1}$ součet všech bodů v tomto ročníku. Tituly uvedené v předchozím textu slouží pouze pro účely M&M.

Kontakty:

M&M, OPMK, MFF UK
Ke Karlovu 3
121 16 Praha 2

E-mail: mam@matfyz.cz
Web: mam.matfyz.cz
FB: casopis.MaM

matfyz
M&M