Sem vložte zadání Vaší práce.

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra počítačových systémů

Bakalářská práce

Konverze grafů v dokumentech Microsoft Excel do jazyka LaTeX

Milan Klouček

Vedoucí práce: Ing. Jiří Kašpar

13. února 2015

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 13. února 2015 .....

České vysoké učení technické v Praze
Fakulta informačních technologií
© 2015 Milan Klouček. Všechna práva vyhrazena.
Tato práce vznikla jako školní dílo na Českém vysokém učení technickém v Praze, Fakultě informačních technologií. Práce je chráněna právními předpisy a mezinárodními úmluvami o právu autorském a právech souvisejících s právem autorským. K jejímu užití, s výjimkou bezúplatných zákonných licencí, je nezbytný souhlas autora.

Odkaz na tuto práci

Klouček, Milan. *Konverze grafů v dokumentech Microsoft Excel do jazyka LaTeX*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2015.

Abstrakt

Tato práce implementuje převodník grafů z dokumentu Microsoft Office Excel do formátu LaTeX. Analyzuje použité datové formáty a navrhuje principy převodu grafů mezi nimi a ověřuje věrnost tohoto převodu. Navržené principy konverze dokumentu implementuje v jazyce Visual Basic for Application.

Klíčová slova Microsoft Office Excel, LaTeX, graf, převodník, konverze grafu

Abstract

This work implements converter from Microsoft Office Excel charts into the LaTeX. It analyses used data patterns, suggests principles of conversion between them and verifies reliability of the conversion. Suggested principles of conversion are then being implemented in Visual Basic for Application.

Keywords Microsoft Office Excel, LaTeX, chart, converter, chart conversion

Obsah

Seznam obrázků

Úvod

Popis problémové domény

Microsoft Office Excel

Microsoft Office Excel (dále jen MS Excel) patří mezi skupinu programů označovaných jako tabulkové procesory. MS Excel vyvíjený firmou Microsoft získal mezi tabulkovými procesory dominantní postavení a stal se již firemním standardem. Většinou je dodáván jako součást kancelářského balíku Microsoft Office pro operační systémy Windows a Mac OS X. Tabulkový procesor je program, který zpracovává informace uložené v tabulce - matici s řádky a sloupci. V jednotlivých buňkách jsou uložená data či vzorce, které s těmito daty pracují a zobrazují vypočtené údaje.

Důležitou funkcí tabulkových procesorů je schopnost zobrazit tabulková data v podobě grafu. Graf nám často pomáhá lépe pochopit data, protože umožňuje jednoduše zobrazit změny v závislosti na čase či jiném parametru a vztahy různých veličin, které je jinak obtížné ze samotných dat rozpoznat.

TEX

TEX je program pro počítačovou sazbu textu. Z důvodu vyšší náročnosti na schopnosti uživatele je používaný zejména v akademických kruzích. Pro tvorbu stránky používá TEX sadu povelů, které se zapisují přímo do zdrojového textu, na základě těchto povelů je možné definovat komplexnější příkazy označované jako makra. Samotný TEX není uživatelsky příliš přívětivý a je programátorsky náročný, a tak pro něj vznikla řada formátů – balíků maker. Jedním z těchto balíků maker je i LATEX, který můžeme také dále balíčky rozšiřovat o nové funkce, např. tabulky či tvorbu grafů. Formátem LATEX a zejména rozšiřujícími balíčky pro tvorbu grafů se budeme v této práci zabývat.

Konverze

Protože různé grafy jsou nedílnou součástí akademických a jiných prací určených k tisku, vzešel požadavek na převod grafů vytvořených v MS Excel do formátu LaTeXu, přesněji do maker některého z rozšiřujících balíčků pro práci s grafy. Konvertorem pro převod grafů se budeme v této práci zabývat.

Stanovení cílů práce

Hlavním cílem práce je převodník, který bude umět převádět grafy z MS Excel do LaTeXu. K tomu jme stanovili následující úkoly:

Přehled grafů v MS Excel Cílem tohoto úkolu je získat přehled o grafech v programu MS Excel, popsat prvky grafu a jednotlivé typy grafů, se kterými se můžeme v MS Excel setkat a budeme je chtít převést.

Analýza možností grafů v systému LaTeX V předchozím úkolu jsme poznali druhy grafů v MS Excel, v tomto úkolu je potřeba najít možnosti jejich převodu do systému LaTeX. Pokud nebude možné převést všechny druhy grafů, je potřeba zajistit převod alespoň nejpoužívanějších typů grafů, poté u těchto grafů prozkoumat způsob jejich zápisu v systému LaTeX.

Analýza objektového modelu MS Excel U uvedených grafů, které budeme převádět, je třeba analyzovat jejich objektovou reprezentaci v MS Excel a popsat objekty, které budou důležité pro konverzi.

Návrh konverze Pokud známe z předchozích úkolů objektovou reprezentaci grafu v MS Excel a tvorbu grafů pomocí maker v systému LaTeX, můžeme navrhnout konverzi grafů z MS Excel do systému LaTeX. Nejdůležitějším úkolem při převodu uvedených typů grafů je zachovat smysl převáděného grafu, tedy aby stejné informace, které nám o datech poskytuje originální graf, poskytoval i převáděný graf. K tomu bychom chtěli zachovat i formátování vzhledu grafu. Cílem konverze je :

- převést velikost grafu; graf musí být rozměrově stejný,
- převést nadpis grafu a další textové prvky a popisky (pokud nějaké jsou),
- převést a zachovat rozmístění značek na osách a zobrazení mřížky,
- převést legendu, pokud je v grafu obsažena,
- u trojrozměrných grafů převést orientaci pohledu na graf,
- umožnit převod uživatelem nastaveného základního formátování grafu (barva, šířka a typ čar a podobně),

- u textových prvků převést základní formát textu, velikost, styl a barvu.

Návrh aplikace Pokud známe cíle konverze, můžeme navrhnout vhodnou aplikaci. Ta by měla splňovat následující požadavky:

- umožnit jednoduše převést graf dle požadavků na konverzi,
- poskytnout intuitivní a uživatelsky příjemné rozhraní,
- umožnit vlastní nastavení konverze,
- uložit nastavení konverze, aby uživatel nemusel nastavení pokaždé opakovat,
- poskytnout případnou pomoc.

Testování Výslednou aplikaci–převodník je potřeba nakonec náležitě otestovat na různých typech grafů.

Existující řešení

Než přejdeme k práci na převodníku, je důležité nejdříve zjistit, zda již neexistují nějaká řešení stejného problému. V případě převodu grafů se nám nepodařilo najít žádný existují převodník, který by převáděl graf do maker LaTeXu.

Běžným řešením tohoto problému je převod grafu do nějakého formátu obrázku a jeho vložení do zdrojového souboru. Toto řešení má výhodu v tom, že vzhled grafu je zcela totožný jako v programu MS Excel, nevýhoda je, že uživatel nemůže měnit graf přímo v LaTeXu. Pokud chce provést jakoukoli změnu, musí vygenerovat z MS Excel znovu obrázek grafu.

Kapitola 1

Analýza

1.1 Grafy v prostředí MS Excel

V této kapitole se seznámíme s grafem v postředí MS Excel, s jeho hlavními prvky a dále s typy grafů, se kterými se můžeme v MS Excel setkat.

V grafu zobrazujeme data, která máme uspořádána v tabulce na listu. Data jsou uspořádána do sloupců nebo řádků. Data v jednom sloupci či v jednom řádku, která jsou vykreslena v grafu, označujeme jako datovou řadu. Hodnoty datových řad, které jsou ve stejném řádku nebo sloupci, označujeme jako kategorie.

Graf v prostředí MS Excel lze vytvořit jednoduchým způsobem, označíme data v tabulce a v menu vybereme tlačítko vložit graf. Aplikace MS Excel obvykle sama určí nejlepší způsob vykreslení dat, který ale můžeme později ručně změnit. Některé typy grafů, například výsečové nebo burzovní, však vyžadují speciální uspořádání dat.

Při tvorbě grafu si musíme uvědomit, zda chceme vynést v grafu hodnoty umístěné ve sloupcích nebo řádcích tabulky. Sady těchto hodnot (v grafu sloupce stejné barvy) nazýváme datové řady. Údaje na vodorovné (nehodnotové) ose poté nazvýváme kategorie. Na obrázku **??** vidíme stejný graf jako na obrázku **??**, ale jako datové řady jsou použity řádky místo sloupců.

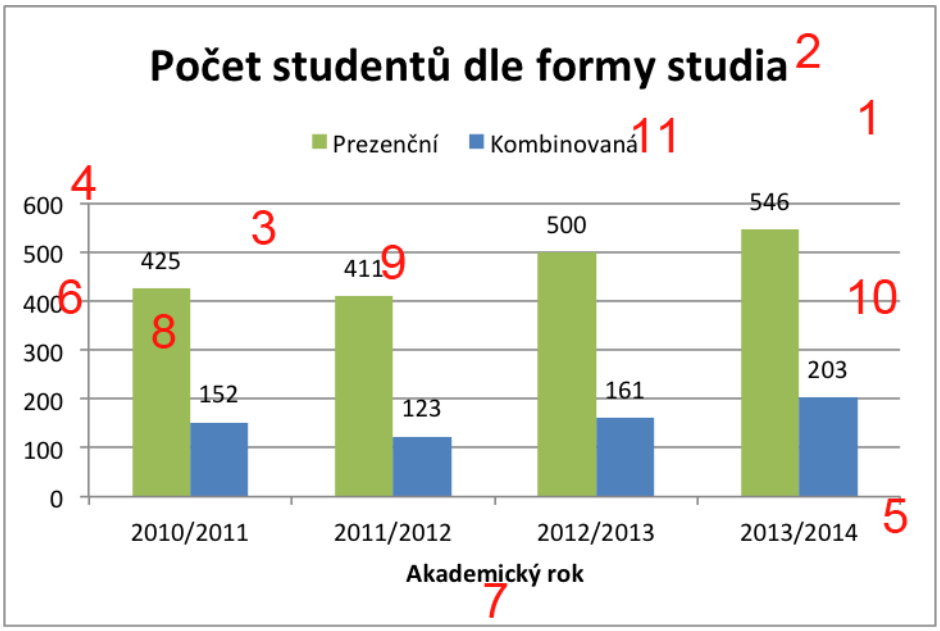
1.1.1 Prvky grafu

Graf má mnoho prvků, některé se zobrazí ve výchozím nastavení, jiné je možné přidat dle potřeby. Prvky grafu lze poté upravovat, měnit jejich vzhled, velikost, umístění v grafu. Prvky lze také odebrat.

Oblast grafu Oblast grafu ohraničuje prostor, kde se nalézají všechny prvky grafu. Můžeme u ní nastavit tažením ohraničení velikost, které se pak ostatní prvky přizpůsobují. Prvky uvnitř oblasti grafu můžeme přemisťovat. Dále je

možné nastavit formát výplně a ohraničení, jako je například průhlednost, tloušťka čáry apod.

Obrázek 1.1: Sloupcový graf

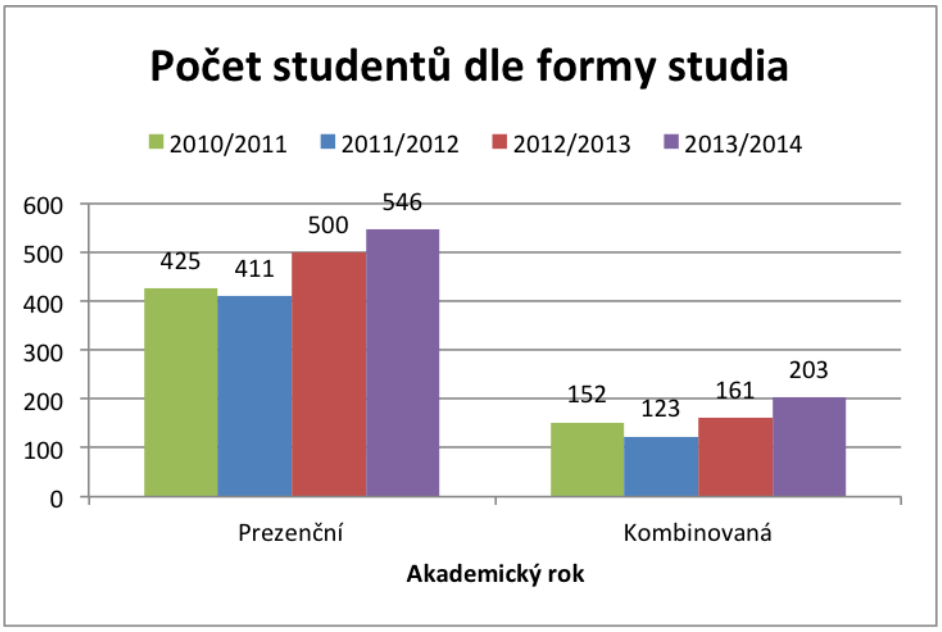


1. Oblast grafu
2. Název grafu
3. Oblast grafu mezi osami
4. Svislá osa – obsahuje hodnoty
5. Vodorovná osa – obsahuje kategorie
6. Značky
7. Název osy
8. Datové body v datové řadě
9. Popisky dat
10. Mřížka grafu
11. Legenda grafu

Název grafu a názvy os Jedná se o textové pole obsahující název grafu nebo osy, text můžeme formátovat podobně jako v textovém editoru. Při vkládání grafu lze vybrat, zda název grafu bude zobrazen nad zobrazovanou oblastí grafu a nebo ji bude překrývat, tažením můžeme pak název grafu umístit libovolně v oblasti grafu.

Oblast grafu mezi osami Je oblast ohraničená osami a zahrnující všechny datové řady. Podobně jako u celé oblasti grafu můžeme nastavit její formát, velikost této oblasti se mění automaticky podle velikosti celé oblasti grafu. V případě trojrozměrného grafu nenastavujeme formát oblasti, ale formát stěn a podstavy.

Obrázek 1.2: Sloupcový graf



Osy Na ose hodnot jsou vyneseny hodnoty grafu. Na hodnotové ose nastavujeme rozmístění značek s jejich popisky, pozici značek a také minimální a maximální hodnotu. Kategorie jsou zobrazeny na ose kategorií, nastavujeme také jejich rozmístění a pozici. V případě trojrozměrného grafu, jako je povrchový graf, má graf navíc třetí hloubkovou osu, na té jsou pak vyneseny datové řady. Osy lze formátovat v grafu jako čáry, můžeme nastavit barvu, tloušťku čáry atd.

Značky Rozdělují osu a popisky označují v grafu jednotlivé hodnoty, kategorie či datové řady. Značky se dělí na hlavní a vedlejší, hlavní značky mají

popisky hodnot. U značek můžeme nastavit jejich pozici na ose či je lze z osy odstranit.

Mřížka Mřížka grafu slouží k usnadnění čtení dat grafu, zejména na hodnotové ose. Mřížka se dělí na hlavní a vedlejší, podle toho, ze kterých značek vychází. Formátujeme podobně jako jiné prvky ve tvaru čáry v grafu.

Legenda Legenda v grafu slouží pro popis datových řad, můžeme změnit její pozici či formátovat vzhled podobně jako u názvu grafu, jednotlivé položky v legendě lze formátovat zvlášť.

Datové body Datové body představují jednotlivé hodnoty v datové řadě. Můžeme u nich uvést hodnotu pomocí popisků dat.

1.1.2 Přehled grafů v MS Excel

Mezi nejpoužívanější typy grafů, které nás budou zajímat, patří graf sloupcový a pruhový, spojnicový, bodový, plošný, povrchový a výsečový.

Sloupcový graf Sloupcový graf se nejčastěji používá pro vzájemné porovnání několika položek v závislosti na časovém období nebo podle kategorie. Skládá se z vertikálních sloupců, jejichž délka odpovídá velikosti porovnávané veličiny.

Sloupce formátujeme stejně jako jiné grafické objekty, lze nastavit formát výplně sloupce a ohraničení sloupce. Můžeme formátovat celou jednu datovou řadu nebo i každý sloupec–datový bod zvlášť. V MS Excel se nenastavuje explicitně šířka sloupce, tu MS Excel nastavuje automaticky dle velikosti grafu. Lze ale nastavit vzdálenost jednotlivých skupin kategorií a také překrývání sloupců v rámci jedné kategorie.

Podtypy sloupcového grafu:

- Skupinový sloupcový graf: tento typ grafu porovnává hodnoty pro různé kategorie, přičemž vedle sebe staví do jedné kategorie datové řady.
- Skládaný sloupcový graf: tento typ grafu porovnává vztah jednotlivých položek k celku porovnáváním příspěvku každé hodnoty k celkovému součtu pro různé kategorie.
- 100% skládaný sloupcový graf: tento typ porovnává procentuální podíl každé hodnoty k celkovému součtu pro různé kategorie.
- Prostorový sloupcový graf: tento typ se používá, když chceme porovnat data napříč kategoriemi i datovými řadami.

Pruhový graf Jedná se o jinou variantou sloupcového grafu. Datové body jsou zobrazeny vertikálně místo horizontálně. Pruhový graf má i stejné podtypy jako sloupcový graf, kromě prostorového.

Spojnicový graf Spojnicový graf umožňuje zobrazení souvislých dat v čase a jejich srovnání a je proto ideální pro zobrazení trendů v datech ve stejných intervalech. Spojnicové grafy je možné zobrazit se značkami, které zvýrazní jednotlivé bodové hodnoty v grafu.

U spojnicového grafu můžeme formátovat spojnici jako čáru a značky jako jiné grafické tvary, u značek lze formátovat kromě výplně i jejich ohraničení. Dále lze nastavit velikost a typ značky.

Podtypy spojnicového grafu:

- Spojnicový graf: tento typ zobrazuje trendy za určitou dobu nebo pro různé kategorie.
- Skládaný spojnicový graf: tento typ grafu zobrazuje trend příspěvku každé hodnoty za určitou dobu nebo pro různé kategorie.
- 100% skládaný spojnicový graf: tento typ je vhodný pro porovnání trendů příspěvků jednotlivých hodnot vzhledem k času nebo kategoriím v procentech.
- Prostorový spojnicový graf: tento typ zobrazuje data jako pásku s prostorovým efektem.

Bodový graf Bodový graf zobrazuje jednotlivé hodnoty jako body, zkombinované z hodnot na ose X a Y. Bodové grafy se používají typicky pro zobrazení vědeckých, statistických a technických dat. Bodový graf je podobný spojnicovému grafu, ale na rozdíl od něj zobrazuje hodnoty i na vodorovné ose a nelze proto používat kategorie; všechny hodnoty musí být číselné. Spojnice a body formátujeme stejně jako u spojnicového grafu.

Podtypy bodového grafu:

- Bodový graf: tento graf zobrazuje datové body bez propojovacích čar.
- Bodový s vyhlazenými spojnicemi: tento typ grafu zobrazí plynulou křivku spojující datové body. Spojnice může být zobrazena se značkami nebo bez značek.
- Bodový s rovnými spojnicemi: tento typ propojí jednotlivé datové body rovnými čárami. Opět mohou být zobrazeny se značkami nebo bez značek.

Plošný graf Plošný graf je podobný spojnicovému grafu bez značek. Používají se ke zvýraznění změny v průběhu času a upoutání pozornosti na celkovou hodnotu v trendu. Všechny podtypy plošného grafu je možné zobrazit s prostorovým efektem. Jednotlivé plochy lze formátovat stejně jako sloupce v sloupcovém grafu, nastavovat formát výplně a i ohraničení plochy.

Podtypy plošného grafu:

- Plošný graf: tento typ zobrazuje trendy za určitou dobu nebo pro různé kategorie.
- Skládaný plošný graf: tento typ grafu zobrazuje trend příspěvku každé hodnoty za určitou dobu nebo pro různé kategorie.
- 100% skládaný spojnicový graf: tento typ je vhodný pro porovnání trendů příspěvků jednotlivých hodnot vzhledem k času nebo kategoriím v procentech.
- Prostorový plošný graf: tento typ zobrazuje data jako pásku s prostorovým efektem.

Povrchový graf Povrchový graf je vhodný pro nalezení optimální kombinace dvou množin dat. Barvy podobně jako v topografické mapě označují oblasti se stejným rozsahem hodnot.

Povrchový graf je typ prostorového grafu, můžeme u něj tedy různě nastavovat prostorový efekt jako je rotace, naklonění nebo perspektivu.

Podtypy povrchového grafu:

- Prostorový povrchový graf: tento typ grafu zobrazuje trendy hodnot ve dvou rozměrech pomocí souvislé křivky.
- Obrysový: jedná se o povrchový graf zobrazený shora.

Výsečový graf Výsečový graf slouží pro zobrazení procentuálního podílu částí vzhledem k celku, pomocí kruhu rozděleného na několik dílů. Poměr jednotlivých dílů odpovídá poměru znázorňovaného množství. Výsečové grafy mají pouze jednu datovou řadu.

Můžeme formátovat jednotlivé výseče. Dále lze nastavit velikost rozložení u rozloženého výsečového grafu či pootočení výsečí.

Podtypy výsečového grafu:

- Výsečový graf: zobrazuje příspěvek každé hodnoty k celku, je k dispozici také s prostorovým efektem.

- Rozložený výsečový: zvýrazňuje jednotlivé hodnoty pomocí rozložení na jednotlivé výseče.

- Výsečový s dílčí výsečí a výsečový s dílčími pruhy: tento typ výsečového grafu usnadňuje rozlišení malých výsečí tak, že extrahuje nižší hodnoty do vedlejšího výsečového grafu nebo skládaného pruhového grafu.

Další typy grafů jsou méně časté, patří mezi ně válcový, kuželový a jehlanový graf, prstencový graf, bublinový, paprskový a burzovní.

Válcový, kuželový a jehlanový graf Jedná se pouze o jiné zobrazení sloupcového nebo pruhového grafu s prostorovým efektem.

Prstencový graf Prstencový graf slouží podobně jako výsečový pro zobrazení procentuálního podílu částí vzhledem k celku, ale může obsahovat více než jednu datovou řadu. Prstencové grafy jsou hůře čitelné.

Bublinový graf Bublinový graf je podobný bodovému grafu, ale přidává navíc další datový sloupec pro hodnoty, které jsou vyjádřené velikostí bubliny. Tento graf je také možné zobrazit s prostorovým efektem.

Paprskový graf Paprskový graf porovnává úhrnné hodnoty několika datových řad.

Burzovní graf Tento typ grafu se používá pro znázornění ceny na burze. Jeho vytváření vyžaduje přesné uspořádání dat v tabulce.

1.2 Grafy v prostředí LaTeX

V předchozí kapitole jsme se seznámili s typy grafů, které je možné vytvořit v prostředí MS Excel. Tyto grafy bychom chtěli umět zobrazit co nejvěrněji i v prostředi LaTeX. Systém LaTeX sám o sobě neumožňuje vykreslování grafů ze zadaných dat a musíme proto použít nějaký rozšiřující baliček pro práci s grafikou a grafy. Je pravděpodobné, že nepůjde převést všechny typy grafů, ale chtěli bychom umět převádět alespoň nejpoužívanější grafy a jejich vlastnosti.

Jak již bylo zmíněno v úvodní kapitole, systém LaTeX používá pro formátování stránek a textu povely, které se zapisují přímo do zdrojového souboru. Rozšiřující balíčky se používají stejným způsobem, Přidávají do systému nové prostředí. Prostředí je oblast zdrojového textu odděleného příkazy `\begin{název prostředí}` a `\end{název prostředí}`. V tomto prostředí pak mohou platit speciální příkazy, syntaxe a nastavení, které nemusí platit vně a naopak. Některá prostředí lze do sebe vnořovat, ale nemohou se křížit.

Pokud budeme hledat na internetu prostředí schopné vykreslit grafy v LaTeXu, většina webových stránek nebo uživatelů internetových fór nás odkáže na balíček **PgfPlots** [**?**]. Proto se seznámíme s tímto balíčkem a jeho možnostmi.

Balíček PgfPlots je založen na rozšíření pro práci s grafikou PGF/TikZ. Účelem balíčku je poskytnout prostředí, ve kterém bude možné jednoduše vykreslovat grafy systému TeXnebo LaTeX. PgfPlots poskytuje nástroje pro generování grafů přímo ze zdrojového souboru. Grafy lze generovat jako funkci nebo pomocí zadaných datových bodů buď v externím datovém souboru nebo přímo ze zdrojového souboru LaTeXu.

Je možné vytvářet dvourozměrné i trojrozměrné grafy, skládané grafy, přidávat osy, popisky, mřížku či legendu. Pokud srovnáme možnosti PgfPlots a grafy MS Excel, můžeme pomocí balíčku PgfPlots vytvořit tyto typy grafů:

- sloupcový a pruhový graf
- bodový graf
- spojnicový graf
- plošný graf
- povrchový graf

Jedná se tedy o většinu nejpoužívanějších grafů, kromě grafu výsečového. PgfPlots výsečový graf nemá. Výsečové grafy nejsou příliš přehledné například při porovnávání více položek či pokud jsou malé rozdíly ve velikosit jejich hodnot a doporučuje se jejich nahrazení například sloupcovými grafy. Přesto je tento typ grafu oblíbený a my bychom ho chtěli umět převádět i do LaTeXu.

Pokusíme se tedy najít alternativní řešení, balíčků umožňujících vykreslit výsečový graf není mnoho na výběr, přímo pro výsečové grafy se nabízí **Pgf-Pie**[**?**]. Tento balíček je založen opět na PGF/TiKZ a umožňuje základní vykreslení základního výsečového grafu, nedisponuje ale možnostmi PgfPlots.

1.3 Tvorba grafů pomocí balíčku PgfPlots a Pgf-Pie

V této kapitole si předvedeme, jak se tvoří grafy v LaTeXu pomocí balíčků PgfPlots a Pgf-Pie. Všechny grafy kromě výsečového budeme vytvářet v PgfPlots, a proto se seznámíme nejprve se základními příkazy a nastaveními tohoto prostředí a poté si předvedeme tvorbu těchto grafů. Nakonec si ukážeme tvorbu výsečového grafu pomocí Pgf-Pie.

1.3.1 Základní příkazy PgfPlots

Deklarace balíčku PgfPlots Tak jak je v LaTeXu obvyklé, nejdříve musíme deklarovat použití balíčku PgfPlots. Mezi ostatní deklarace přidáme řádek: `\usepackage{pgfplots}`.

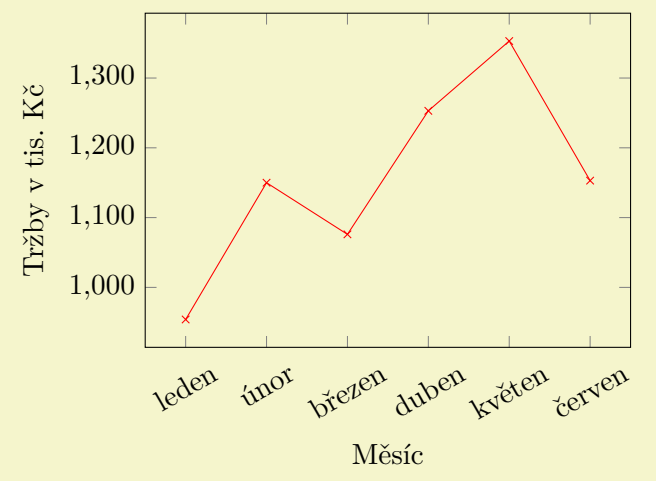
Grafické prostředí Protože PgfPlots je postaven na rozšíření PGF/TikZ, jsou grafy vykreslovány jako obrázek balíčku TikZ, prostředí tohoto obrázku je určeno příkazy `\begin{tikzpicture}` a `\end{tikzpicture}`. V tomto prostředí budou umístěny všechny příkazy grafu.

Oblast grafu Oblast grafu je ohraničena vlastním prostředím pomocí příkazů `\begin{axis}` a `\end{axis}`. Za příkazem `\begin{axis}` obvykle následují hranaté závorky s parametry grafu, např. definice legendy, popisky os a mnoho dalších.

Vykreslení grafu Příkazem `\addplot` vykreslíme graf, přesněji jednu datovou řadu. Za příkazem následují volitelné parametry uzavřené do hranatých závorek. Po parametrech může být uvedeno klíčové slovo, které specifikuje způsob předání dat do příkazu. Následují složené závorky, kde jsou umístěna data, odkaz na datový soubor nebo funkce a nakonec středník. Příkaz addplot lze do prostředí axis zadávat opakovaně pro vykreslení více datových řad do jednoho grafu.

Toto jsou základní příkazy, na obrázku **??** je ukázka vykreslení jednoduchého grafu a ukázka kódu.

Obrázek 1.3: Příklad jednoduchého grafu v PgfPlots



```
\begin{tikzpicture}
\begin{axis}[
    width=8cm, height=6cm,
    xlabel=Měsíc,
    ylabel=Tržby v tis. Kč,
    xlabel near ticks, ylabel near ticks,
    xtick=data,
    yticklabels={1000, 1100, 1200, 1300},
    xticklabels={leden, únor, březen, duben, květen, červen},
    xticklabel style={rotate=30},]
\addplot[only marks,color=red,mark=x] coordinates {
    (1,954) (2,1150) (3,1076) (4,1253) (5,1353) (6,1153)};
\end{axis}
\end{tikzpicture}
```

1.3.2 Definice vstupních dat

Grafy v MS Excel jsou vykresleny na základě dat v tabulce. PgfPlots má několik možností, jak data předat. Data se předávají příkazu `\addplot` a pro jejich specifikaci můžeme využít několika klíčových slov. Předávat lze tabulky dat, soubor či matematické funkce. Pro naši práci budou nejdůležitější data v podobě tabulky.

V předchozím příkladu bylo použito příkazu `\addplot` s klíčovým slovem `coordinates`. Toto klíčové slovo bere jako parametry souřadnice jednotlivých hodnot v závorkách oddělených mezerou. Je vhodné například pro jednoduchý bodový či spojnicový graf, ale i sloupcový.

Nejpoužívanější klíčové slovo pro definici zdroje dat je `table`, jak název napovídá, toto slovo bude přijímat jako parametry tabulku. Tabulku je možné zadat přímo do zdrojového textu, a nebo mít uloženou v externím souboru. V tabulce jsou v defaultním nastavení sloupce odděleny mezerou. Tabulka může mít více sloupců, pak je vhodné mít označené jejich názvy a lze tak specifikovat z kterých dat se má graf vykreslit. Pokud sloupce sami nespecifikujeme, jsou použity první dva. Použití příkazu `\addplot table` s přímo uvedenou tabulkou dat ukazuje následující příklad.

```
\begin{tikzpicture}
\begin{axis}[
    axis background/.style={fill=shadecolor},
    width=8cm, height=6cm,
    xlabel=Den,
    ylabel=Vlhkost,
    grid=both,
    symbolic x coords={pondělí, úterý, středa, čtvrtek, pátek},
    grid style={color={rgb:red,255;green,0;blue,0}}]
\addplot table [x=den, y=vlhkost] {
    den      teplota     vlhkost
    1        23          55
    2        25          57
```

```
    3        21          50
    4        23          49
    5        26          60
};
\end{axis}
\end{tikzpicture}
```

V defaultním nastavení je potřeba mít i data, která budeme používat jako kategorii, v číselném formátu. Často jsou ale kategorie pojmenovány slovně. V tomto případě použijeme nastavení parametru `symbolic x coords`, tedy na příkladu dnů v týdnu bychom uvedli:
`symbolic x coords={pondělí, úterý, středa, čtvrtek, pátek}`.

Oddělovačem řádku je nový řádek, oddělovačem sloupců alespoň jedna mezera. Pokud by data obsahovala mezeru, uzavřeme je do složených závorek. Oddělovače lze změnit pomocí parametrů `row sep` a `column sep` příkazu `\addplot`.

Pokud bychom chtěli vykreslit do grafu více datových řad, museli bychom pro každý příkaz `addplot` znova vypsat tabulku dat. To by bylo velmi nepraktické a tak PgfPlots nabízí možnost definovat data odděleně a používat je tak opakovaně. Použijeme příkazu `\pgfplotstableread`, ve kterém definujeme tabulku dat, a za ním uvedeme referenční název např. `\mojetabulka`. Referenční název pak použijeme jako odkaz na tabulku takto: `\addplot table \mojetabulka;`

Použití externí definice tabulky je uvedeno na příkladu a vykreslený graf je na obrázku **??**.

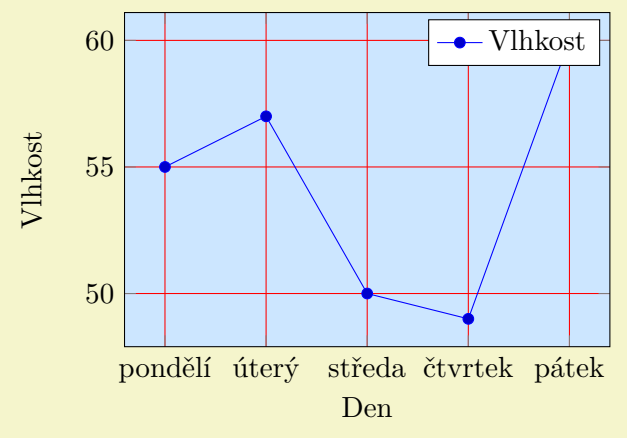
```
\pgfplotstableread{
den    teplota   vlhkost
pondělí       23          55
úterý         25          57
středa        21          50
čtvrtek       23          49
pátek         26          60
}\mojetabulka
```

Pro velká množství dat je již nutné mít tabulku uloženou v externím souboru. Definici extrerního souboru zapisujeme do složených závorek i s cestou, cestu můžeme vynechat pokud je soubor umístěn ve stejné složce jako zdrojový soubor LaTeXu. Použití datového souboru pak vypadá následovně:
`\addplot table {název souboru};`

1.3.3 Nastavení prvků grafu

Nastavení většiny prvků grafu, jako jsou název grafu, názvy os, legenda apod., se zapisuje do hranatých závorek za příkazem `\begin{axis}`, nastavení jednotlivých prvků jsou oddělena mezi sebou čárkou. Po klíčovém slově názvu

Obrázek 1.4: Použití tabulky dat – výsledný graf



parametru následuje rovnítko s hodnotou parametru. Některé parametry jsou bez hodnoty.

Parametry mají často možnost dalšího nastavení prostřednictvím stylů, s jejichž pomocí můžeme upravovat barvu, polohu, zarovnání a mnoho dalšího. Pro většinu prvků postačí defaultní nastavení, které podává dobré výsledky při zobrazení grafu a není nutné ho příliš měnit.

Seznámíme se nejprve s hlavními parametry, které nás při převodu budou zajímat nejvíce, a na konci kapitoly i s některými možnostmi základního nastavení stylů.

Oblast grafu Oblast grafu je v PgfPlots ohraničena grafickým prostředím tikzpicture. Pokud chceme nastavit pozadí, musíme deklarovat použití knihovny pozadí: `\usetikzlibrary{backgrounds}`. Potom můžeme v hranatých závorkách specifikovat pozadí, nejprve zobrazení pozadí jako obdélníku pomocí `show background rectangle` a poté vlastní styl pozadí, například barvu výplně `background rectangle/.style={fill=red}`.

Podobně lze nastavit v prostředí axis zobrazovanou část grafu, tedy část ohraničenou osami: `axis background/.style`.

V tomto prostředí se nastavuje i velikost oblasti grafu parametry `width` a `height`. Těmito parametry se nastavuje velikost zobrazované části grafu dohromady s popisky os a značek. PgfPlots graf této velikosti přizpůsobuje, neupravuje ovšem velikost písma ani legendy, proto celá oblast grafu (prostředí tikzpicture) může být ve skutečnosti větší. Tuto celkovou velikost nelze pevně nastavit. PgfPlots nabízí parametr `scale only axis`, pokud ho do parametrů zapíšeme, bude se nastavená velikost aplikovat na zobrazovanou část grafu – oblast pouze mezi osami bez značek a popisků.

Název grafu Pro pojmenování grafu slouží parametr `title`. Pokud budeme chtít nastavit umístění nebo barvu názvu, použijeme nastavení stylu pomocí zápisu `title style={}`.

Názvy os Názvy os zadáme pomocí parametry `xlabel`, `ylabel` a pro prostorový graf `zlabel`. Může se stát, že názvy os se budou překrývat s popiskami značek os, v tom případě můžeme použít parametru `xlabel near ticks`. Nastavení formátu názvu os provedeme pomocí zápisu `xlabel style={}`.

Značky os a jejich popisky Značky os jsou označovány jako xtick, ytick, ztick. Pokud nepoužijeme žádné parametry, PgfPlots sám určí nejlepší rozmístění a popisky značek. Značky rozděleny na major ticks a minor ticks, major ticks jsou hlavní značky, minor ticks jsou vedlejší značky, které leží mezi hlavními značkami, vedlejší značky nemají popisky. V případě, že chceme definovat hlavní nebo vedlejší značky na konkrétní ose, zapisujeme jako major ytick nebo minor ytick.

Nastavením parametru `ytick=data` vynutíme vytvoření značek dle příslušných hodnot dat v tabulce. Značky lze vytvořit také pro libovolné hodnoty, které sami určíme, např.: `major ytick={1000, 1100, 1200}`. Opakování značek v určité vzdálenosti můžeme zajistit pomocí zápisu `major ytick={0,50, ...,300}`. Hlavní značka se tak bude opakovat každých 50 jednotek od 0 do 300.

Na osách budeme ještě chtít nastavit minimum a maximum, to provedeme pomocí parametrů `xmin` a `xmax`.

Pro přímo definované popisky značek existuje parametr `xticklabels`, kde jsou vyjmenované popisky značek, používá se v kombinaci s parametrem `ytick`, počet popisků by měl odpovídat počtu značek. Pokud ale chceme některé značky bez popisků, necháme v závorkách místo popisku samotnou čárku.

Mřížka grafu Mřížka grafu se označuje jako grid a vytváří se podle značek os. Pokud chceme zobrazit mřížku hlavních značek, použijeme parametr `grid=major`. Hodnota parametru `both` pak zobrazí mřížku i pro vedlejší značky. Pro zobrazení mřížky pouze pro vybrané osy jsou k dispozici parametry `xmajorgrids`, `ymajorgrids` atd.

Legenda Legendu lze umístit do grafu několika způsoby. Za každým vykreslením datové řady příkazem `\addplot` použijeme příkaz `\addlegendentry {Řada 1}`, nebo můžeme příkazem `\legend{Řada1, Řada 2}` vložit legendu pro všechny datové řady najednou.

U legendy je důležitá její poloha. V defaultním nastavení je umístěna v rámečku v pravém rohu uvnitř zobrazované části grafu. Pro jednoduché nastavení pozice legendy lze využít parametru `legend pos` a jeho hodnot, které

nám umožní umístit legendu do některého ze 4 rohů a to vně nebo uvnitř zobrazované části grafu, např.: `legend pos=outer north east`. Pro specifičtější umístění legendy je již nutno použít stylů.

U legendy můžeme mimo použití stylů nastavit ještě v kolika sloupcích se má zobrazit, tj. pokud bychom chtěli legendu o třech prvcích zobrazit v jednom řádku, nastavíme parametr `legend columns=3`.

1.3.4 Písmo

V prostředí LaTeXse nastavuje velikost a typ fontu na začátku zdrojového souboru (pokud nechceme používat výchozí nastavení) a LaTeX
potom sám přizpůsobí všechny velikosti názvů kapitol a podkapitol. PgfPlots toto nastavení přebírá a používá pro popisky prvků v grafu stejný font a velikost.

Změnu velikosti fontů v PgfPlots můžeme provést několika způsoby. Například pro změnu všech popisků prvků grafu na velikost small v parametrech prostředí tikzpicture uvedeme `font=\small`. Velikost lze nastavit také jako parametr prostředí axis, pak se bude vztahovat na všechny popisky grafu, kromě nadpisu. Pokud bychom chtěli změnit velikost jen určitého prvku, deklarujeme příkaz velikosti fontu přímo před hodnotou parametru, např. pro změnu velikosti nadpisu takto `title=\Large Název grafu`. Pro další změny lze využít opět stylů.

1.3.5 Barva

Barvu lze specifikovat mnoha způsoby, nejjednodušší je prostý zápis některé základní barvy, například red, blue, black. Často používaný formát barvy je RGB, kdy se uvádí velikost jednotlivých složek barvy, přímo můžeme definovat RGB barvu takto `{rgb,255:red,200;green,0;blue,100}`.

Pokud víme, že nějakou specifickou barvu budeme používat v dokumentu opakovaně, můžeme ji definovat vlastním názvem v definicích dokumentu a pak ji používat pomocí jejího názvu. Barvu definujeme zápisem`\definecolor {jmenobarvy}{RGB}{255,127,0}`. Jsou možné i jiné formáty barvy, například cmyk, HTML a další.

1.3.6 Styly

Styly u jednotlivých prvků použijeme, pokud chceme podrobněji upravovat vzhled grafu. Pomocí stylů lze nastavovat prvkům polohu, barvu, ohraničení a mnoho dalšího. Většinu parametrů, kterými nastavujeme prvky grafu, je možné použít s nastavením stylů použitím slova style za název parametru a nastavení stylu poté zapisujeme do složených závorek oddělená čárkou, např. pro nastavení stylu názvu grafu: `title style={}`.

Styl výplně Pokud má nějaký prvek grafu výplň, můžeme nastavit její barvu zápisem `fill`. Za rovnítkem použijeme některý ze způsobů zápisu barvy. U výplně lze nastavit její průhlednost pomocí zápisu `fill opacity`, hodnota průhlednosti je od 0 do 1, přičemž hodnota 0 znamená 100% průhlednost. Pokud prvek výplň nemá, nebo je 100% průhledný, můžeme použít zápisu `fill=none`.

Styl čáry Podobně jako výplň lze nastavit styl čáry, pouze použijeme klíčové slovo `draw`. U čáry lze navíc nastavit její typ a šířku. Pro nastavení typu čáry stačí zapsat do stylu některou z definic typu čáry, například `dotted` pro tečkovanou čáru. Šířku čáry v bodech nastavíme zápisem `line width`.

Styl písma Nastavení stylu písma použijeme u textových prvků tam, kde chceme použít jiný styl písma než je definován pro celý dokument. Do nastavení stylu prvku zapíšeme `font={}`. ve složených závorkách pak můžeme nastavit:

- font pomocí `\fontfamily{}`,
- velikost písma pomocí `\fontsize{}{}`, kde v první závorce je velikost, v druhé velikost mezery řádků,
- pomocí `\fontshape{}` lze nastavit například písmo italic (it),
- pomocí `\fontseries{}` tak lze nastavit například tučné písmo (b)

Barva písma se nastavuje u textových prvků zápisem `text`.

1.3.7 Spojnicový a bodový graf

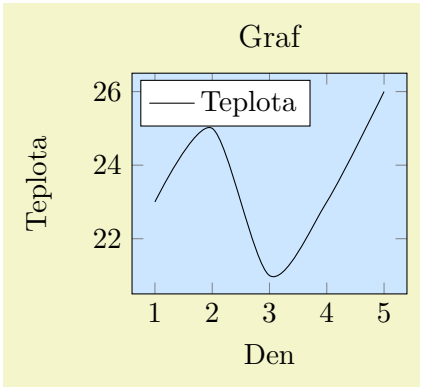
Spojnicový graf je v PgfPlots defaultním grafem, značky jsou spojeny čárou. Spojnice lze ve spojnicovém grafu vyhladit pomocí parametru `smooth`. Pokud bychom chtěli odstranit z grafu značky, použijeme parametr `no markers` příkazu `\addplot`. Naopak pokud chceme vytvořit bodový graf, použijeme parametr `only marks`, který zobrazí v grafu pouze značky bez spojovací čáry. Tvar značek si můžeme vybrat z celé řady možností nastavením parametru `mark`.

Spojnicové a i další typy grafů můžeme skládat na sebe, použitím parametru `stack plots=y`, kde hodnota parametru (nejčastěji y) určuje, na jaké ose se mají jednotlivé datové řady skládat. Pokud parametr použijeme jako parametr prostředí axis, složí se v grafu všechny datové řady. Parametr můžeme ale použít i pro každý příkaz addplot, pak se složí jen řady s tímto parametrem.

U spojnicového grafu můžeme nastavit formát a typ značek. Pro typ značky zapíšeme parametr `mark` a definici značky, například `*` pro kruh. Velikost značky v bodech průměru se nastavuje zápisem `mark size`. Styl značky poté pomocí `mark options`.

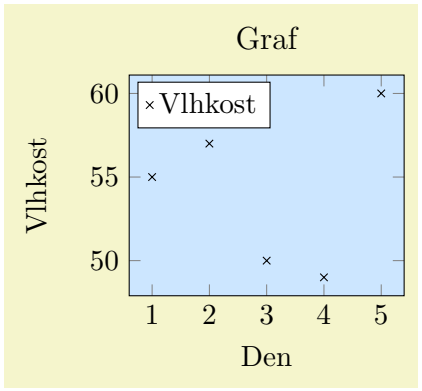
Obrázek 1.5: Spojnicový graf s vyhlazenou spojnicí

```
\begin{tikzpicture} [
    show background rectangle,
    background rectangle/.style={fill=zluta1},
]
\begin{axis}[
    axis background/.style={fill=shadecolor},
    height=4.5cm,
    xlabel=Den,
    ylabel=Teplota,
    title=\large Graf,
    xtick=data,
    legend pos= north west,
    stack plots=y,
]
\addplot[no markers, smooth]
    table[x=den,y=teplota] \weathertable;
\addlegendentry{Teplota}
\end{axis}
\end{tikzpicture}
```



Obrázek 1.6: Bodový graf

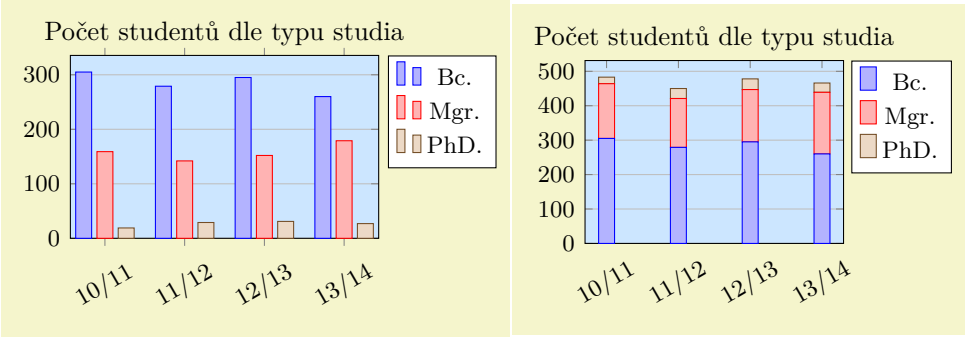
```
\begin{tikzpicture} [
    show background rectangle,
    background rectangle/.style={fill=zluta1}
]
\begin{axis}[
    axis background/.style={fill=shadecolor},
    height=4.5cm,
    xlabel=Den,
    ylabel=Vlhkost,
    title=\large Graf,
    xtick=data,
    legend pos= north west,
]
\addplot[only marks, mark=x]
    table[x=den,y=vlhkost] \weathertable;
\addlegendentry{Vlhkost}
\end{axis}
\end{tikzpicture}
```



1.3.8 Sloupcový a pruhový graf

Pro to, aby příkaz `\addplot` vykreslil datové řady jako sloupcový graf, je v PgfPlots parametr `xbar` prostředí axis, pro vykreslení pruhového grafu pak parametr `ybar`. Níže je uveden příklad sloupcového grafu a skládaného sloupcového grafu.

Obrázek 1.7: Sloupcový a skládaný sloupcový graf




```
\begin{tikzpicture} [
    show background rectangle,
    background rectangle/.style={fill=zluta1}
]
\begin{axis}[
    footnotesize,
    height=4cm,
    ybar=2pt, %pro skládaný graf nahradíme parametrem: ybar stacked
    bar width=6pt,
    x={3*6+3*2+10},
    enlarge x limits={abs=12pt},
    ymin=0,
    symbolic x coords={10/11, 11/12, 12/13, 13/14},
    axis background/.style={fill=shadecolor},
    legend pos=outer north east,
    xtick=data,
    xticklabel style={rotate=30},
    ylabel near ticks,
    ymajorgrids,
    title=Počet studentů dle typu studia,
]
\addplot table[x=Rok, y=Bc.] \studenttab;
\addplot table[x=Rok, y=Mgr.] \studenttab;
\addplot table[x=Rok, y=PhD.] \studenttab;
\legend{Bc., Mgr., PhD.}
\end{axis}
\end{tikzpicture}
```

Parametr `bar width` nastavuje šířku sloupce, parametr `ybar` určí vzdálenost mezi sloupci v jedné kategorii, `x` je vzdálenost mezi jednotlivými kategoriemi (od středu každé kategorie).

Při vykreslování sloupcového grafu se může stát, že sloupce grafu se dostanou mimo zobrazovanou oblast či se budou překrývat, proto je potřeba vhodně parametry nastavit. V příkladu je pro ilustraci uvedeno nastavení vzdálenosti mezi kategoriemi takto `x={3*6+3*2+10}` – graf zobrazuje tři datové řady, tedy 3x šířka sloupce + 3x vzdálenost mezi sloupci + mezera navíc. Parametr

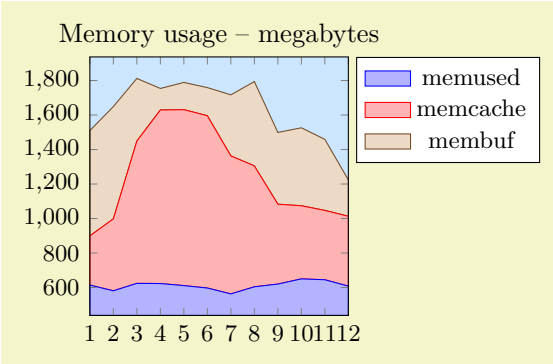
`enlarge x limits` nastavuje vzdálenost první a poslední značky od kraje grafu. V příkladu byl použit navíc parametr `ymin`, který nastavil počáteční hodnotu na ose y na 0 – tak aby graf ležel přímo na ose x.

Skládaný sloupcový graf lze vytvořit upravením parametru `ybar` přidáním klíčového slova na `ybar stacked` a podobně pruhový graf, nebo můžeme stejně jako např. u spojnicového grafu použít parametr `stack plots`. Sloupcový ani pruhový graf nelze zobrazit pomocí PgfPlots s 3D efektem a tedy ani ve válcové či kuželové podobě.

1.3.9 Plošný graf

Plošný graf se zobrazí z dat pomocí parametru prostředí axis `area style`. Na konci každého příkazu `\addplot` je potřeba ještě uvést příkaz `\closecycle`, aby vyplněná plocha grafu byla uzavřená. Pro složení grafů slouží opět parametr `stack plots`.

Obrázek 1.8: Plošný graf




```
\begin{tikzpicture} [
    show background rectangle,
    background rectangle/.style={fill=zluta1}
]
\begin{axis}[
    footnotesize,
    area style,
    stack plots=y,
    height=5cm,
    xmin=1,
    xmax=12,
    axis background/.style={fill=shadecolor},
    legend pos=outer north east,
    xtick=data,
    title=Memory usage -- megabytes,
]
\addplot table[x=hour, y=memused]       {memusage.txt} \closedcycle;
\addplot table[x=hour, y=memcache]      {memusage.txt} \closedcycle;
```

```
\addplot table[x=hour, y=membuf]        {memusage.txt} \closedcycle;
\legend{memused, memcache, membuf, freemem}
\end{axis}
\end{tikzpicture}
```

V tomto příkladu byly použity navíc parametry `xmin` a `xmax`, které určují minimum a maximum na ose x, protože v defaultním nastavení je počáteční hodnota 0.

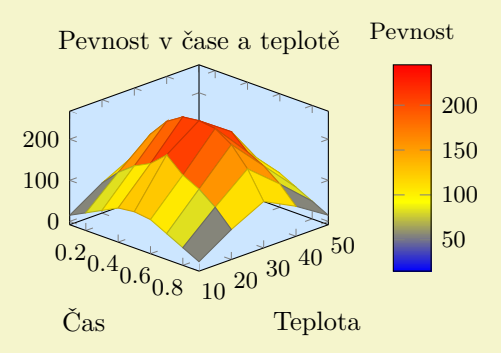
1.3.10 Povrchový graf

Povrchový graf patří mezi trojrozměrné grafy. V PgfPlots se trojrozměrný graf vykresluje pomocí příkazu `\addplot3`. Data se opět předávají ve formě tabulky, která se ale liší od obvyklého uspořádání tabulky pro povrchový graf v MS Excel. Tabulka obsahuje na řádku hodnoty pro všechny tři osy, data pro jednu hodnotu osy y se oddělují prázdným řádkem.

Povrchový graf můžeme zobrazit jako plochu parametrem příkazu addplot3 `surf` nebo jako síť `mesh`. také je zde možnost zobrazit pouze jednotlivé body pomocí parametru `scatter`.

U povrchového grafu obvykle zobrazujeme pouze jednu datovou řadu – jeden povrch, proto není nutné zobrazovat legendu. Pro lepší orientaci umožňuje PgfPlots zobrazit vedle grafu barevný sloupec s rozsahy hodnot osy z. Barevný sloupec se zobrazí použitím parametru `colorbar`. Barevný sloupec je sám grafem – prostředím axis a můžeme pro něj použít stejná nastavení prostřednicvím stylu.

Obrázek 1.9: Povrchový graf

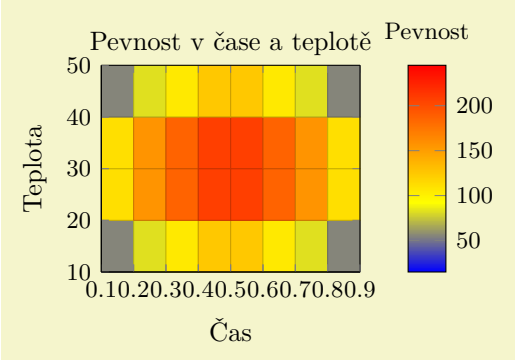



```
\begin{tikzpicture} [
    show background rectangle,
    background rectangle/.style={fill=zluta1}
]
\begin{axis}[
    footnotesize,
```

```
    view={45}{30},
    axis background/.style={fill=shadecolor},
    xlabel=Čas,
    ylabel=Teplota,
    ytick=data,
    title=Pevnost v čase a teplotě,
    colorbar,
    colorbar style={
        title=\footnotesize Pevnost,
        ytick={50,100,150,200},
    }
]
\addplot3[surf] table[y index=0, x index=1, z index=2] {tensiletable.txt};
\end{axis}
\end{tikzpicture}
```

V povrchovém grafu je důležitým parametrem parametr `view`, který určuje, pod jakým úhlem budeme na trojrozměrný graf nahlížet. První hodnota udává natočení podle vertikální osy, druhá hodnota pak dle horizontální osy. Pokud bychom tedy nastavili hodnotu parametru například na `{0}{90}`, dívali bychom se na graf shora.

Obrázek 1.10: Povrchový graf při pohledu shora



1.3.11 Výsečový graf

Pokud budeme chtít vykreslit výsečový graf, budeme používat balíček Pgf-Pie. nejpre je ho potřeba deklarovat pomocí zápisu `\usepackage{pgf-pie}`. Má jediný příkaz `\pie`, který se zapisuje do prostředí tikzpicture. Hodnoty jsou zadávány do složených závorek s názvem datové řady odděleným lomítkem. Nastavení grafu se provádí pomocí následujících parametrů:

text Tento parametr určuje popis jednotlivých výsečí. Může nabývat těchto hodnot: legend pro zobrazení ve formě legendy, label pro popisky u výsečí, pin

pro popis na stopce a inside pro popisky uvnitř grafu. Defaultní hodnota je nastavena na label.

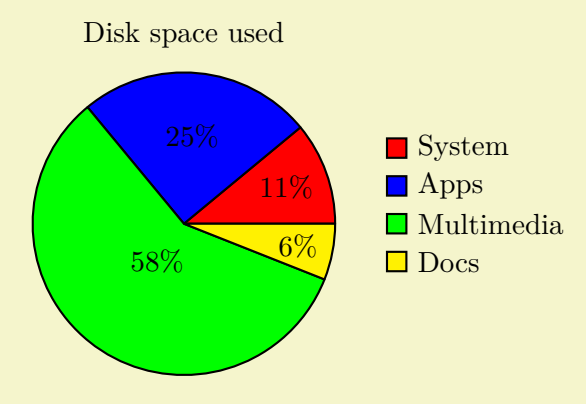
before number a after number Tyto parametry nám umožnují před a za číselnou hodnotu vložit znak, nejčastěji například % .

radius Určuje poloměr grafu v centimetrech. Defaultní hodnota je 3.

color Nastavuje barvu výsečí grafu. Pokud zadáme barvy oddělené čárkou v určitém pořadí, nastaví barvy výsečí tak jak jsou zadány jejich hodnoty za sebou.

sum Parametr sum určuje celkovu sumu hodnot, kterou graf představuje. Nastavení na auto sečte všechny hodnoty datové řady. Defaultní hodnota je nastavena na 100, v tomto případě jsou zobrazeny hodnoty výsečí v podobě procent.

Obrázek 1.11: Výsečový graf




```
\begin{tikzpicture} [
    show background rectangle,
    background rectangle/.style={fill=zluta1}
]
\pie[
        text=legend,
        radius=2,
        after number=\%,
        sum=auto,
        color={red, blue, green, yellow},
]
{11/System, 25/Apps, 58/Multimedia, 6/Docs}
\node (title) at (0, 2.5) {Disk space used};
\end{tikzpicture}
```

Balíček Pgf-Pie nemá způsob, jak nastavit název grafu, nicméně můžeme využít příkazu `\node` balíčku PGF/TiKZ. Použití je uvedeno v příkladu na předposledním řádku: `\node (title) at (0, 2.5) {Disk space used};`. Tento příkaz vytvoří "bod" pojmenovaný title s textem Disk space used na souřadnicích 0 a 2.5. Souřadnice jsou od středu obrázku, v našem případě tedy od středu výsečového grafu. Pokud má graf poloměr 2cm, nastavíme souřadnici y o něco větší, například tedy 2.5cm.

1.4 Objektový model MS Excel

Abychom mohli převést grafy z MS Excel do LaTeXu, je nutné nejprve znát objektový model MS Excel. V programu MS Excel jsou data z načteného zdrojového souboru převedena na vnitřní reprezentaci zprostředkovanou objektovým modelem. Objekty do určité míry odpovídají standardním prvkům, které jsme zvyklí používat prostřednictvím uživatelského rozhraní. MS Excel poskytuje dále celou řadu objektů, se kterými můžeme manipulovat pomocí VBA. V této práci se budeme zabývat jen objekty potřebnými pro pochopení objektového modelu a pro konverzi.

Hierarchická struktura objektového modelu

Objekty jsou uspořádány v hierarchické struktuře. Na vrcholu hierarchie stojí objekt Application, jedná se o samotnou otevřenou aplikaci programu MS Excel. Protože se jedná o kořenový objekt, je v hierarchii pouze jeden. Program MS Excel může být spuštěný vícekrát, pak má každý svou vlastní hierarchii s kořenem Application. Pod objektem Application leží objekt Workbook (sešit), který představuje v tabulkovém procesoru jeden pracovní soubor. Objekt Workbook obsahuje objekty typu Worksheet. Worksheet je již pracovní list s tabulkovým prostředím. S buňkami tabulky můžeme manipulovat pomocí objektu Range, který označuje vybraný rozsah buněk.

Navigace v hierarchické struktuře

Pro navigaci v hierarchické struktuře slouží operátor tečka (.), kterým propojujeme jednotlivé objekty. Například chceme získat na aktivním lisu hodnotu v buňce A1, což je atribut Value objektu Range, použijeme tedy zápis Application.ActiveWorkbook. ActiveSheet.Range("A1").Value. Některé objekty jsou defaultní a tak je lze vynechat, například objekt Application, ActiveWorkbook i ActiveSheet. Pokud je vynecháme, MS Excel jedná tak, jako kdyby byly uvedeny.

Přehled objektů grafu

V následujícím přehledu jsou uvedeny důležité objekty a jejich atributy. Jedná se pouze o popis nejdůležitějších objektů, které budou potřebné pro převod grafu. Popis všech objektů a jejich atributů, které se vážou ke grafům v MS Excel je mimo rozsah této práce a je dostupný na webu Microsoftu[**?**].

Objekt Chart Grafy, kterými se budeme zabývat v této práci, jsou objekty typu Chart, který patří v hierarchii objektového modelu pod objekt Worksheet. Objekt Chart bude pro nás výchozím objektem, ze kterého budeme odvozovat ostatní objekty a atributy při převodu grafu.

- **ChartType** Tento atribut objektu Chart vrací typ grafu. Je možné používat název nebo číselnou hodnotu. Pro grafy, kterými se budeme zabývat v konverzi má tyto názvy a hodnoty:

typ grafu	**n**ázev	**h**odnota
spojnicový	xlLine	4
spojnicový skládaný	xlLineStacked	63
100% spojnicový skládaný	xlLineStacked100	64
spojnicivý se značkami	xlLineMarkers	65
skládaný spoj. se značkami	xlLineMarkersStacked	66
100% skládaný spoj. se zn.	xlLineMarkersStacked100	67
bodový	xlXYScatter	-4196
bodový se spojnicemi	xlXYScatterLines	74
bodový se sp. bez značek	xlXYScatterLinesNoMarkers	75
bodový s vyhlazenými spoj.	xlXYScatterSmooth	72
bodový s vyhl. spoj. bez zn.	xlXYScatterSmoothNoMarkers	73
sloupcový	xlColumnClustered	51
skládaný sloupcový	xlColumnStacked	52
100% skládaný sloupcový	xlColumnStacked100	53
pruhový	xlBarClustered	57
skládaný pruhový	xlBarStacked	58
100% skládaný pruhový	xlBarStacked100	59
plošný	xlArea	1
skládaný plošný	xlAreaStacked	76
100% skládaný plošný	xlAreaStacked100	77
výsečový graf	xlPie	5

- **HasTitle** Je True, pokud má graf zobrazovaný název, jinak je False.
- **HasLegend** Vrací True, pokud má graf legendu.

Objekt ChartTitle Představuje název grafu, budou nás zajímat tyto atributy:

- **Text** Atribut text vrací textový řetězec s názvem grafu. Pro získání názvu grafu tedy použijeme zápis Chart.ChartTitle.Text.

- **Left a Top** Tyto atributy určují přesné umístění názvu grafu, umístění je určeno jako vzdálenost v bodech zleva a shora od levého horního okraje nadřazeného objektu. V případě názvu grafu je nadřazeným objektem objekt ChartArea.

- **Format** Tento atribut vrací objekt ChartFormat, který v tomto případě představuje formát názvu grafu. Více o objektu ChartFormat v následující podkapitole.

Objekt ChartFormat Objekt ChartFormat určuje formát prvků grafu. V případě převodu grafu nás budou zajímat především objekty **LineFormat** a **FillFormat**. K těmto objektů se dostaneme přes atributy Format.Line a Format.Fill.

Objekt FillFormat specifikuje formát výplně u prvků grafu, které jsou tvořeny plochami a objekt LineFormat specifikuje formát čáry prvků grafu, které jsou tvořeny čárou nebo mají ohraničení. U obou těchto objektů nás budou zajímat především tyto atributy:

- **Visible** Atribut Visible určuje, zda je objekt viditelný, pokud ne, je zcela průhledný, hodnota atributu může být buď True nebo False.

- **Transparency** Tento atribut vrací průhlednost objektu jako hodnotu od 0 do 1. Přitom hodnota 0 je zcela neprůhledný a hodnota 1 zcela průhledný.

- **ForeColor** Tento atribut vrací objekt ColorFormat, který představuje barvu objektu, popsáno v odstaveci ColorFormat.

- **DashStyle** U objektu LineFormat nás bude ještě zajímat atribut DashStyle, který určuje typ čáry, dle hodnoty atributu jsou to tyto typy:

typ čáry	**n**ázev	**h**odnota
plná čára	msoLineSolid	1
tečkovaná s čtvercovými body	msoLineSquareDot	2
tečkovaná s kulatými body	msoLineRoundDot	3
čárkovaná	msoLineDash	4
čerchovaná	msoLineDashDot	5
dvojčerchovaná	msoLineDashDotDot	6
čárkovaná s dlouhými čarami	msoLineLongDash	7
čerchovaná s dlouhými čarami	msoLineLongDashDot	8

- **Weight** Atribut Weight určuje tloušťku čáry v bodech.

Objekt Font Objekt Font vrací některé atributy objektů, které obsahují text.

- **Size** Velikost písma v bodech.
- **FontStyle** Styl písma, může být například bold nebo italic.
- **Color** Barva písma.

Objekt ColorFormat ColorFormat reprezentuje barvu příslušného objektu. Odkaz na objekt vrací zároveň také hodnotu 0 - 16777215, která reprezentuje RGB barvu ve čtyřech bytech jako 00BBGGRR. První byte (zleva) má hodnotu nula, druhý byte BB označuje hodnotu modré barvy v rozmezí 0-255, třetí byte hodnotu zelené barvy v rozmezí 0-255 a čtvrtý byte hodnotu červené v rozmezí 0-255.

Objekt ChartArea Tento atribut vrací objekt ChartArea, který reprezentuje celou oblast grafu.

- **Height a Width** Na oblasti grafu nás bude zajímat především její velikost – atributy Height a Width, které určují výšku a šířku oblasti v bodech.
- **Format** Dále nás bude zajímat formát výplně a ohraničení, tedy atributy Format.Fill a Format.Line, viz odstavec ChartFormat.

Objekt PlotArea Objekt PlotArea reprezentuje v grafu oblast, kde je vykreslen graf.

- **Height a Width** Atributy Height a Width vracejí stejně jako u objektu ChartArea její velikost v bodech, jedná se však o velikost včetně popisků os a značek.
- **InsideHeight a InsideWidth** Tyto atributy vrací velikost oblasti grafu mezi osami.
- **Format** Formát výplně získáme z atributu Format.Fill a formát ohraničení oblasti z Format.Line, viz odstavec ChartFormat. Ohraničení se může krýt s osami, které mají v tomto případě přednost.

Objekt Legend Objekt legend představuje legendu grafu, stejně jako objekt ChartTitle, má atributy Top a Width určující jeho pozici vůči nadřazenému objektu – ChartArea a atribut Format, který určuje vzhled legendy viz odstavec ChartFormat.

Objekt ChartGroup Objekt Chart se skládá z jedné nebo více ChartGroups, jedna ChartGroup je členem kolekce ChartGroups. ChartGroup představuje skupinu stejného typu grafů v jednom grafu, uplatní se například v případě smíšeného grafu. V našem případě nás budou zajímat dva atributy objektuChartGroup a to:

- **GapWidth** Tento atribut určuje mezeru mezi jednotlivými kategoriemi, používá se pro nastavení vzdálenosti v sloupcovém a pruhovém grafu.
- **Overlap** Atribut Overlap nastavuje překrytí či naopak odstup sloupců resp. pruhů v rámci jedné kategorie.

Objekt Series Objekt Series reprezentuje v grafu jednu datovou řadu. Je členem kolekce SeriesCollection. Pokud chceme získat konkrétní datovou řadu, použijeme zápis SeriesCollection(i), kde i je index datové řady. Na atributy objektu Series budeme odkazovat vždy, když budeme chtít přistupovat k vlastnostem a nastavením datové řady.

- **Formula** Tento atribut vrací řetězec reprezentující jednu datovou řadu v grafu. Řetězec má tvar adres rozsahů buněk v pracovním sešitu oddělených čárkou, například takto: =SERIES(Sheet1!$B$1,Sheet1!$A$2:$A$5, Sheet1$B$2:$B$5,1). První adresa (A2) ukazuje na umístění názvu datové řady, druhá (A2 až A5) na názvy kategorií a třetí (B2 až B5) na samotná data.

  Tento atribut nám tedy umožní zjistit umístění zdrojových dat grafu v tabulce, protože objekt Chart nemá žádnou metodu či atribut, který by odkazoval na zdrojová data v tabulce.

- **MarkerSize** Atribut MarkerSize určuje velikost značek jedné datové řady u bodového nebo spojnicového grafu se značkami. Velikost se udává jako poloměr v bodech od 2 do 72.

- **MarkerStyle** Specifikuje typ značky, možné typy jsou tyto:

typ značky	**n**ázev	**h**odnota
automatický výběr	xlMarkerStyleAutomatic	-4105
kruh	xlMarkerStyleCircle	8
dlouhá čára	xlMarkerStyleDash	-4115
kosočtverec	xlMarkerStyleDiamond	2
krátká čára	xlMarkerStyleDot	-4118
bez značek	xlMarkerStyleNone	-4142
obrázek	xlMarkerStylePicture	-4147
znaménko plus	xlMarkerStylePlus	9
čtverec	xlMarkerStyleSquare	1
hvězda	xlMarkerStyleStar	5
trojúhelník	xlMarkerStyleTriangle	3
písmeno x	xlMarkerStyleX	-4168

- **Format** Atribut Format vrací ChartFormat datové řady tak jak je popsáno v odstavci ChartFormat, dle typu grafu určují jeho atributy Line a Fill vzhled datové řady.

  Atribut Fill vrací u datových řad FillFormat, který určuje dle typu grafu:

  - u sloupcového a pruhového grafu výplň sloupce nebo pruhu,
  - u bodového grafu a spojnicového grafu se značkami vrací 0, ačkoli v MS Excel lze nastavit formát výplně značky, toto bude patrně chyba v objektovém návrhu MS Excel,
  - u plošného grafu vrací atribut výplň plochy,
  - u výsečového grafu společný formát výplně všech výsečí.

  Atribut Line vrací u datových řad LineFormat, který určuje dle typu grafu:

  - u sloupcového a pruhového grafu ohraničení sloupce nebo pruhu,
  - u bodového grafu a spojnicového vrací atribut Line formát spojnice, avšak jeho změna pomocí jazyka VBA má i vliv na formát ohraničení značky, toto je opět chyba v MS Excel,
  - u plošného grafu ohraničení plochy,
  - u výsečového grafu společný formát ohraničení všech výsečí.

- **MarkerForeroundColor a MarkerBackgroundColor** V předchozím odstavci bylo uvedeno, že v objektovém modelu MS Excel nemáme přístup k nastavením formátu značek. Pomocí atributu MarkerForeroundColor můžeme ale zjistit barvu ohraničení značky a pomocí atributu MarkerBackgroundColor barvu výplně značky.

- **Points** Metoda Points vrací odkaz na kolekci objektů Point v datové řadě. K jednotlivým bodům kolekce bodů v datové řadě se dostaneme použitím metody Points takto: Series.Points(i), kde i je index datového bodu, datové body jsou seřazeny v grafu zleva doprava, tedy v grafu nejvíce vlevo je datový bod s indexem 1 a nejvíce vlevo datový bod s indexem Points.Count.

Objekt Point Objekt Point představuje datový bod v datové řadě, například jeden sloupec v sloupcovém grafu nebo jednu značku v bodovém grafu. Vztahují se k němu stejné atributy jako k datové řadě, které jsme uvedli odstavci Series.

V grafu má obvykle jedna datová řada stejné nastavení například formátu výplně, v případě výsečového grafu, který má jen jednu datovou řadu chceme výseče odlišit. Výseče jsou v tomto případě datové body jedné datové řady a proto provedeme nastavení na jednotlivých datových bodech.

Objekt Axis Metoda Axes objektu Chart vrací kolekci Axes, která obsahuje osy grafu – objekty Axis. K jednotlivým osám grafu pak přistpujeme z kolekce takto: k ose kategorií pomocí Axes(xlCategory), k ose hodnot pomocí Axes(xlValue) a u trojrozměrného grafu k ose datových řad pomocí Axes(xlSeries).

- **HasTitle a AxisTitle** Pokud je atribut HasTitle True má osa zobrazen název osy a lze získat pomocí atributu AxisTitle objekt název osy. Můžeme u něj používat stejné atributy jako u názvu grafu viz odstavec ChartTitle.
- **MajorTickMark a MinorTickMark** Tyto atributy určují pozici hlavních a vedlejších značek na ose:

pozice značky	**n**ázev	**h**odnota
křížem přes osu	xlTickMarkCross	4
na ose uvnitř	xlTickMarkInside	2
vně osy	xlTickMarkOutside	3
bez značek	xlTickMarkNone	-4142

- **MajorUnit a MinorUnit** Atributy MajorUnit a MinorUnit určují na ose hodnot vzdálenost hlavních a vedlejších značek.
- **MinimumScale a MaximumScale** Tyto atributy se vztahují pouze k ose hodnot, určují minimální a maximální hodnotu na ose.
- **TickMarkSpacing a TickLabelSpacing** Na ose kategorií nebo v případě trojrozměrného grafu na ose datových řad určuje počet kategorií nebo řad mezi značkami resp. mezi popisky značek, například hodnota 2 znamená, že značku (popisek) má první, třetí, pátá atd. kategorie nebo řada. Hodnoty těchto atributů mohou být od 1 do 31999.
- **HasMajorGridlines a HasMinorGridlines** Pokud má atribut hodnotu True, má graf na této ose zobrazenu hlavní resp. vedlejší mřížku. Pokud ano, vrátí nám atribut **MajorGridlines** nebo **MinorGridlines** objekt **Gridlines**. Protože je mřížka tvořena čárami, bude nás u tohoto objektu zajímat pouze jeho formát čáry, viz odstavec ChartFormat.
- **TickLabelPosition** Atribut TickLabelPosition určuje pozici popisků značek na ose, může mít tyto hodnoty:

pozice popisku značky	**n**ázev	**h**odnota
vpravo nebo nahoře	xlTickLabelPositionHigh	-4127
vlevo nebo dole	xlTickLabelPositionLow	-4134
blízko osy	xlTickLabelPositionNextToAxis	4
bez popisků	xlTickLabelPositionNone	-4142

- **Format** Protože osa je zobrazena jako čára, bude nás zajímat formát čáry osy, viz odstavec ChartFormat.

1.5 Visual Basic for Applications

Jazyk Visual Basic for Applications (dále jen VBA) [**?**], je objektově orientovaný událostmi řízený programovací jazyk odvozený od jazyka Visual Basic (VB). Jeho hlavním rozdílem oproti jazyku VB je to, že v něm nelze psát samostatně spustitelné aplikace, zato je však obsažen zdarma v každém kancelářském balíku Microsoft Office a nepotřebuje tedy vlastní vývojové prostředí, programování probíhá přímo v prostředí aplikací Microsoft Office.

Naprogramovaným úlohám v prostředí Microsoft Office říkáme makra, obvykle slouží pro automatizaci nějaké činnosti, například výpočtů nebo úpravy textu či tabulek. Pomocí VBA ale můžeme vytvořit i rozsáhlejší aplikaci, i když nepůjde spustit samostatně. Součástí Microsoft Office je i záznamník maker, kterým můžeme nahrávat naši činnost a poté se podívat na posloupnost příkazů jazyka VBA, které byly během činnosti vykonávány a zaznamenány.

V jazyce VBA lze jednoduše vytvářet grafické uživatelské rozhraní (GUI) pomocí formulářů. Do formulářů lze vkládat tlačítka, textová a zaškrtávací pole a další prvky známé z běžných aplikací. Podobný formulář bude využívat i námi vytvořený převodník.

Kapitola 2

Návrh konverze

2.1 Princip

Převodník nejprve převede data grafu z excelové tabulky do podoby, kterou vyžaduje PgfPlots (Pgf-Pie pro případ výsečového grafu) a uloží ji do externího souboru, pro případ výsečového grafu zapíše data přímo do zdrojového souboru. Poté postupně převede prvky grafu podle jejich objektové reprezentace na makra LaTeXu (rozšiřujícího balíčku) a zapíše je na výstup.

Převodník bude zpracovávat najednou vždy jen jeden vybraný graf na aktivním listu, proto budeme na aktuálně zpracovávaný graf odkazovat pomocí objektu ActiveChart.

2.2 Převod dat

Umístění dat grafu v excelové tabulce lze zjistit prostřednictvím atributu **SeriesCollection(i).Formula**, kde je i pořadí datové řady. Převodník tak prochází v cyklu jednotlivé řady a z atributu Formula extrahuje umístění názvu datové řady, rozsah, kde jsou názvy kategorií a rozsah, kde jsou hodnoty. Pro každou datovou řadu pak zapisuje do jednoho sloupce její název a hodnoty, které prochází ve vnořeném cyklu. Názvy kategorií pak stačí projít jenom jednou a zapsat jako první sloupec.

Protože PgfPlots na rozdíl od MS Excel vyžaduje, aby tabulka dat obsahovala názvy kategorií i datových řad, je potřeba v případě, že v excelové tabulce tyto názvy nejsou, je do tabulky dat doplnit. Pokud převodník zjistí, že tabulka neobsahuje názvy datových řad, nebo názvy kategorií, doplní místo nich pořadové číslo řady nebo kategorie. Kromě toho je třeba doplnit název prvního sloupce, kde jsou názvy kategorií, zde můžeme vybrat libovolný název, nesmí se ovšem shodovat s názvem žádné datové řady.

Protože PgfPlots neumí vykreslit 100% typy skládaných grafů přímo, musíme využít skládaných grafů a data upravit. Sečteme data ze všech řad v jedné

kategorii a vydělíme 100, tak získáme hodnotu jednoho procenta, touto hodnotou pak podělíme jednotlivé hodnoty v každé řadě. Hodnoty zaokrouhlíme na dvě desetiná místa, takto bude součet hodnot všech řad v jedné kategorii dávat 100 s odchylkou v řádu setin, což nebude na grafu postřehnutelné.

2.3 Barva

Objekty, u nichž lze definovat barvu, mají obvykle atribut Color či ForeColor, který vrací číselnou hodnotu ve 4 bytech jako 00BBGGRR. V PgfPlots je jeden ze způsobů definice barvy zápisem `{rgb,255:red,RR;green,GG;blue,BB}` a proto můžeme jednoduše extrahovat z číselné hodnoty atributu Color složky RR, GG a BB takto:

- červenou složku (RR) získáme výpočtem Color mod[1] 256,
- zelenou složku (GG) získáme jako (Color / 256)[2] mod 256,
- modrou složku (BB) pak získáme jako Color / 65536.

2.4 Text a Font

Pro převod textových prvků jako název grafu, legenda či popisky os je potřeba umět převádět text z MS Excel do LaTeXu. Protože MS Excel a LaTeXpracují s písmy odlišně, bylo by komplikované převádět přesně text tak jak je. Museli bychom převádět znak po znaku a u každého znaku sledovat jeho velikost, barvu, styl písma apod. Proto si určíme, které vlastnosti textu a jak budeme převádět. Převádět budeme tyto vlastnosti textu:

- velikost písma,
- barva textu,
- styl písma (tučné, kurzíva).

Tyto vlastnosti budeme převádět pro celý prvek grafu najednou, například nadpis bude tedy zobrazen celý jednou velikostí textu, jednou barvou a jedním stylem písma. Toto nastavení přispěje i k celkové přehlednosti a jednotné úpravě grafu.

PgfPlots používá jako font pro textové prvky grafu defaultní font LaTeXového dokumentu. Toto nastavení jsme se rozhodli zachovat, protože dokument a graf tak budou mít jednotný styl, který uživatel může změnit v nastavení dokumentu. Font tedy nebudeme nijak převádět.

[1] Operace zbytek po celočíselném dělení.
[2] Lomítko zde představuje operaci celočíselné dělení.

2.5 Formát čáry

Při převodu stylu prvků grafu budeme nastavovat i formát čáry a to u těchto prvků:

- ohraničení názvu grafu,
- ohraničení legendy,
- mřížka grafu,
- osy grafu,
- ohraničení oblasti grafu,
- spojnice u spojnicového nebo bodového grafu se spojnicemi.

K atributům čáry se dostaneme z objektu přes jeho formát **Format.Line**. Například pro ohraničení názvu grafu ChartTitle.Format.Line. Převádět budeme tyto vlastnosti:

Tloušťka Tloušťku v bodech získáme z atributu čáry **Weight** a zapíšeme do nastavení stylu příslušného prvku jako `line width=Weight`.

Typ Typ čáry získáme z atributu **DashedStyle** a převedeme, některé typy nemají v PgfPlots zastoupení a pak je tedy převedeme na jim nejbližší typ. Například v MS Excel může být tečkovaná čára složená z hranatých nebo kulatých bodů, do PgfPlots oba typy převedeme jako tečkovanou čáru. Do nastavení stylu prvku podle hodnoty atributu **DashedStyle** zapíšeme:

- `dotted` pro 2 a 3,
- `dashed` pro 4 a 7,
- `dashdotted` pro 5 a 8,
- `dashdotdotted` pro 6 a 9,
- pro 1 a ostatní hodnoty bychom zapisovali `solid`, ale toto je výchozí hodnota v PgfPlots a není ji potřeba zapisovat.

Barva Barvu čáry získáme z atributu **ForeColor** a zapíšeme pomocí rgb zápisu `draw={rgb,255:red,R;green,G;blue,B}`.

Průhlednost Průhlednost čáry získáme z jejího atributu **Transparency**. Při hodnotách 1 a více je průhlednost 100% a čára se vůbec nezobrazuje. V tomto případě to budeme převádět buď jako `draw=none` a nebo u prvků, kde v PgfPlots ve výchozím nastavení není čára zobrazována (např. ohraničení názvu grafu), nebudeme zapisovat nic. Pro hodnoty menší než 1 musíme nejprve převést hodnotu průhlednosti, protože v MS Excel je hodnota 100% průhlednosti 1, zatímco v PgfPlots 0, proto zapíšeme `draw opacity=1-ChartArea.Format.Line.Transparency`. Hodnotu 0, v PgfPlots odpovídá 1, nemusíme opět zapisovat, protože průhlednost 0% je defaultní pro každý zobrazovaný prvek.

2.6 Formát výplně

Pro některé prvky grafu budeme nastavovat výplň obrazce, typicky rámečku nebo pozadí. K tomu budeme potřebovat převádět formát výplně, podobně jako formát čáry u těchto prvků:

- výplň rámečku názvu grafu,
- výplň rámečku legendy,
- pozadí oblasti grafu mezi osami,
- pozadí oblasti grafu,
- výplň sloupců (pruhů) u sloupcového nebo pruhového grafu,
- výplň ploch u plošného nebo povrchového grafu.

K atributům výplně se dostaneme z objektu přes jeho formát **Format.Fill**. Například pro výplň rámečku názvu grafu ChartTitle.Format.Fill. Převádět budeme pouze barvu a průhlednost, podobně jako u převodu formátu čáry:

Barva Barvu výplně získáme z atributu **ForeColor** a zapíšeme `fill={rgb,255:red,R;green,G;blue,B}`.

Průhlednost Průhlednost výplně získáme z jejího atributu **Transparency**. Při hodnotách 1 a více je průhlednost 100% a výplň se vůbec nezobrazuje. V tomto případě to budeme převádět jako `fill=none` a nebo u prvků, kde v PgfPlots ve výchozím nastavení není výplň není zobrazována (např. výplň oblasti grafu), nebudeme zapisovat nic. U těchto prvků také nebudeme zapisovat nic, pokud barva výplně bude v MS Excel mít hodnotu none což zjistíme tak, že atribut **Visible** má hodnotu 0.

Pro hodnoty průhlednosti menší než 1 musíme nejprve převést hodnotu průhlednosti, protože v MS Excel je hodnota 100% průhlednosti 1, zatímco v PgfPlots 0, proto zapíšeme `fill opacity=1-Transparency`. Hodnotu 0,

v PgfPlots odpovídá 1, nemusíme opět zapisovat, protože průhlednost 0% je defaultní pro každý zobrazovaný prvek.

2.7 Oblast grafu

Při konverzi oblasti grafu nás bude zajímat její velikost, ohraničení a výplň. Protože velikost oblasti grafu je odvozena od velikosti zobrazované části grafu a nenastavuje se přímo, budeme se jí věnovat v kapitole Velikost grafu.

Nastavení ohraničení a výplně oblasti grafu se zapisuje do hranatých závorek za příkaz `\begin{tikzpicture}`. Vždy zapíšeme příkaz `show background rectangle` pro zobrazení oblasti grafu a na další řádek potom `background rectangle/.style={}`, kde do závorek uvedeme vlastní nastavení ohraničení a výplně oblasti grafu.

Ohraničení oblasti grafu Nastavení ohraničení zjistíme z formátu objektu **ChartArea.Format.Line**. Při převodu budeme postupovat podle kapitoly Formát čáry. Pokud zjistíme, že oblast nemá ohraničení, nebudeme zapisovat nic.

Výplň oblasti grafu Nastavení výplně oblasti grafu je podobné jako nastavení ohraničení. K atributům formátu výplně se dostaneme pomocí **ChartArea.Format.Fill** a převedeme tak jak je popsáno v kapitole Formát výplně. V defaultním nastavení nemá oblast žádnou výplň a nemusíme tak zapisovat nic.

2.8 Velikost grafu

Při převodu velikosti grafu se musíme rozhodnout, která velikost je pro zobrazení grafu nejdůležitější a jaké jsou možnosti jejího převedení. Zda celá oblast grafu, která obsahuje všechny prvky grafu, zobrazovaná oblast grafu, která obsahuje oblast ohraničenou osami spolu s prvky os nebo pouze oblast mezi osami. Protože velikost oblasti grafu nelze v PgfPlots nastavit a zobrazovaná oblast grafu se přizpůsobuje v MS Excel i PgfPlots prvkům os grafu a může docházet k určitým rozdílům, rozhodli jsme se pro převedení přesné velikosti oblasti mezi osami.

Velikost části grafu mezi osami získáme z **PlotArea.InsideWidth** a **PlotArea.InsideHeight**. Za příkazem `\begin{axis}` mezi hranaté závorky zapíšeme pro nastavení šířky `width=PlotArea.InsideWidth` a výšky `height=PlotArea.InsideHeight`. Pro to, aby se velikost aplikovala pouze na oblast mezi osami, je nutno zapsat řádek `scale only axis`.

2.9 Oblast grafu mezi osami

V předchozí kapitole, jsme převedli velikost grafu dle oblasti grafu mezi osami, dále nám zbývá u oblasti grafu mezi osami převést její výplň a ohraničení. Nastavení se zapisuje do vlastností prostředí axis.

Výplň oblasti grafu mezi osami Nastavení výplně získáme z **PlotArea.Format.Fill**. na rozdíl od výplně celé oblasti grafu je v defaultním nastavení barva výplně bílá, zatímco v PgfPlots výplň není (je průhledná a zobrazuje tak barvu výplně oblasti grafu). Proto budeme barvu výplně grafu mezi osami vždy nastavovat. Pokud žádná výplň není – atribut **Visible** má hodnotu 0, zapíšeme `fill=none`. V ostatních případech zjistíme barvu a převedeme dle kapitoly Formát výplně.

Ohraničení oblasti grafu mezi osami V PgfPlots nastavení ohraničení oblasti mezi osami není k dispozici, místo toho se nastavují samotné osy. V defaultním nastavení jsou zobrazeny všechny osy a vytváří tak zároveň ohraničení oblasti grafu mezi osami, naproti tomu v MS Excel jsou zobrazeny pouze osa hodnot a kategorií, ohraničení se v defaulním nastavení nezobrazuje. Převod ohraničení můžeme tedy provést takto:

- pokud ohraničení není v MS Excel nastaveno, zobrazíme pouze osu hodnot vlevo a osu kategorií dole zápisem `axis y line=left` a `axis x line=down`,
- pokud je ohraničení nastaveno, nebudeme zapisovat nic a budou tedy zobrazeny navíc i osy vpravo a vlevo, formát ohraničení bude nastaven dle formátu os.

2.10 Název grafu a popisky os

Převod grafu a převod popisu os bude probíhat stejným způsobem, jedná se v obou případech o text v rámečku, proto se mu budeme věnovat v jedné kapitole. Zda má graf název, zjistíme z atributu grafu **HasTitle**, pokud je True, můžeme přejít k nastavení názvu grafu. Samotný text názvu získáme z atributu **ChartTitle.Text** a zapíšeme mezi hranaté závorky, kde se provádí nastavení prostředí axis jako `title=ChartTitle.Text`.

K názvům os budeme přistupovat přes atribut **Text** objekt; jednotlivých os, tj. **Axes(xlValue)** pro osu hodnot a **Axes(xlCategory)** pro osu, kde jsou vyneseny kategorie.Nejprve zjistíme, zda má osa zobrazen popisek – atribut **HasTitle** má hodnotu True. Pokud ano, zapíšeme pro osu hodnot řádek `ylabel=Axes(xlValue).AxisTitle.Text` a pro osu kategorií `xlabel=Axes(xlCategory).AxisTitle.Text`. V případě pruhového grafu pouze prohodíme ylabel a xlabel.

Ostatní vlastnosti názvu grafu a názvu os se nastavují pomocí stylu. Pro nastavení stylu nadpisu pak zapíšeme příkaz `title style={}` a do závorek zapíšeme oddělené čárkou nastavení, které specifikují jednotlivé vlastnosti nadpisu. U názvu os zapíšeme stejná nastavení do `ylabel style={}` a `xlabel style={}`.

Velikost písma Velikost písma v bodech získáme z atributu **Font.Size** a zapíšeme jako `font={\fontsize{Font.Size pt}{1em}\selectfont}`.

Tučné písmo a kurzíva Informaci o stylu písma získáme z **Font.FontStyle**. Pokud je jeho hodnota Bold, zapíšeme `\fontseries{b}` před příkaz `\select font` v nastavení písma. Pokud je jeho hodnota Italic, zapíšeme na stejné místo `\fontshape{it}`. Pro hodnotu Bold Italic zapíšeme oba příkazy vedle sebe.

Barva písma Barvu písma získáme jako RGB hodnotu z atributu **Font. Color**. Takto získanou hodnotu je potřeba převést na složky R, G a B (viz kapitola Barva). Potom zapíšeme nastavení barvy písma jako `color={rgb,255: red,R;green,G;blue,B}`.

Ohraničení rámečku Rámeček, ve kterém je umístěn název grafu či popisky os může být ohraničen, nastavení zjistíme z **Format.Line** a převedeme dle kapitoly Formát čáry. V defaultním nastavení se rámeček u těchto prvků nezobrazuje a nemusíme zapisovat nic.

Výplň rámečku Z **Format.Fill** získáme nastavení výplně a převedeme dle kapitoly Formát výplně. V defaultním nastavení se výplň rámečku těchto prvků nezobrazuje a nemusíme tedy převádět.

Pozice názvu V PgfPlots se pozice prvků grafu uvádí vůči bodu, kde se kříží osy grafu (levý spodní roh oblasti mezi osami) zatímco v MS Excel vůči levému hornímu rohu nadřazeného objektu, což je v případě názvu grafu objekt ChartArea. Navíc pozice se v PgfPlots uvádí v procentech výšky či šířky (oblasti mezi osami) a v MS Excel v bodech. Umístění tedy spočítáme takto:

- zleva jako **ChartTitle.Left** - **PlotArea.InsideLeft**. Pro získání pozice v procentech se výsledek vydělí hodnotou atributu **PlotArea.Inside Width**,
- a zhora jako **PlotArea.InsideTop** + **PlotArea.InsideHeight** - **Chart-Title.Top**. Pro získání pozice v procentech se výsledek vydělí hodnotou atributu **PlotArea.InsideHeight**.

Pozice popisů os nebudeme převádět, jejich pozici necháme nastavit PgfPlots automaticky.

2.11 Legenda grafu

Z atributu **HasLegend** zjistíme, zda graf legendu obsahuje, pokud má atribut hodnotu True, můžeme legendu převádět.

Nejdříve je potřeba získat názvy datových řad, které pak legenda popisuje. K tomu slouží atribut **SeriesCollection(i).Formula**, postup je stejný jako při převodu dat (odkaz) – procházíme v cyklu jednotlivé datové řady a extrahujeme umístění názvu datové řady. Názvy datových řad oddělené čárkou poté zapíšeme do závorek příkazu `\legend{}`. Příkaz poté zapíšeme do prostředí axis.

Pro nastavení stylu legendy budeme postupovat podobně jako u nastavení stylu nadpisu. Zapíšeme příkaz `legend style={}` a do závorek zapíšeme oddělené čárkou nastavení vlastností legendy. Abychom zachovali přehlednost grafu, budeme nastavovat styl legendy jako celek, tedy všechny názvy datových řád v legendě budou zobrazeny stejnou velikostí písma, barvou i stylem.

Velikost písma Velikost písma legendy v bodech získáme z atributu **Legend.Font.Size** a tuto velikost zapíšeme jako
`font={\fontsize{Legend.Font.Size pt}{1em}\selectfont}`.

Tučné písmo a kurzíva Styl písma legendy získáme z atributu **Legend.Font.FontStyle**. Pokud je jeho hodnota Bold, zapíšeme `\fontseries{b}` před příkaz `\selectfont` v nastavení písma. Pokud je jeho hodnota Italic, zapíšeme na stejné místo `\fontshape{it}`. Pro hodnotu Bold Italic zapíšeme oba příkazy vedle sebe.

Barva písma Barvu písma získáme jako RGB hodnotu z atributu **Legend.Font.Color**. Takto získanou hodnotu je potřeba převést na složky R, G a B (viz kapitola Barva). Potom zapíšeme nastavení barvy písma jako `color={rgb,255:red,R;green,G;blue,B}`.

Ohraničení rámečku legendy Rámeček, ve kterém je umístěna legenda, může být ohraničen, nastavení ohraničení zjistíme z formátu čáry **Legend.Format.Line**. Převedeme dle kapitoly Formát čáry. na rozdíl od ohraničení názvu grafu se v defaultním nastavení rámeček zobrazuje a v případě, že nemá být zobrazen, je potřeba zapsat `draw=none`.

Výplň rámečku legendy Nastavení výplně rámečku získáme z **Legend.Format.Fill** a převedeme dle kapitoly Formát výplně. V defaultním nastavení PgfPlots se ovšem výplň rámečku legendy zobrazuje v bílé barvě, zatímco v MS Excel zachovává nastavení barvy pozadí oblasti grafu - je průhledné. V tomto případě musíme zapsat `fill=none`.

Pozice legendy Pozici legendy budeme nastavovat přímo dle souřadnic, bez ohledu na to, zda v MS Excel bylo specifikováno umístění například nad grafem, vlevo nebo i nastaveno zobrazení přes graf. Pozici legendy spočítáme takto:

- zleva jako **Legend.Left** - **PlotArea.InsideLeft**. Pro získání pozice v procentech se výsledek vydělí hodnotou atributu **PlotArea.Inside Width**,
- a zhora jako **PlotArea.InsideTop** + **PlotArea.InsideHeight** - **Legend.Top**. Pro získání pozice v procentech se výsledek vydělí hodnotou atributu **PlotArea.InsideHeight**.

2.12 Osy

K ose hodnot budeme přistupovat přes objekt **Axes(xlValue)**, k ose kategorií přes objekt **Axes(xlCategory)**. U prostorového grafu je navíc osa datových řad **Axes(xlSeries)**. Všechny objekty a atributy os budou odvozeny od těchto objektů. Nastavení os budeme zapisovat do vlastností prostředí axis.

Pro nastavení prvků os je nejříve potřeba určit, která osa je osou hodnot a která osou kategorií. To zjistíme z typu grafu určeného atributem **ChartType**. Osa x je hodnotová pouze pro pruhový graf. Pokud bude hodnotovou osou osa x, nahradíme v následujících zápisech osu y osou x (písmeno y písmenem x). V případě povrchového grafu je pak hodnotovou osou osa z, osou kategorií osa x a osou datových řad osa y.

Bodový graf zobrazuje i osu kategorií jako hodnotovou, přistupuje se k ní stejně jako k ose kategorií přes objekt **Axes(xlCategory)**, ale budou pro ni platit stejná nastavení, jako kdyby byla osou hodnot x.

Zobrazení a pozice značek na ose Umístění hlavních značek na ose zjistíme z atributu **MajorTickMark**. Pokud bude hodnota tohoto atributu -4142, osa hlavní značky neobsahuje a mohli bychom zapsat příkaz `ymajorticks =False`, avšak tím by se nezobrazily ani popisky značek, proto musíme použít nastavení stylu značky, kde zapíšeme parametr draw=none takto: `major y tick style={draw=none}`. Pro ostatní hodnoty atributu zapíšeme řádek `ytick align=`

- `inside` pro MajorTickMark = 2,
- `outside` pro MajorTickMark = 3,
- `center` pro MajorTickMark = 4.

Pozici vedlejších značek nebudeme převádět, protože vedlejší značky jsou v PgfPlots vždy umístěny stejně jako hlavní. Z pozice značek na ose ale můžeme zjistit, zda osa má zobrazeny vedlejší značky. Defaultně se vedlejší značky nezobrazují a ani nemají popisky a tak v případě, že je hodnota atributu -4142 nebudeme zapisovat nic. Pokud je hodnota atributu **MajorTickMark** různá od -4142, na ose jsou zobrazeny i vedlejší značky.

To, zda se zobrazují popisky značek, zjistíme z atributu **TickLabelPosition**, pokud je hodnota tohoto atributu rovna -4142, popisky se nezobrazují a musíme zapsat příkaz `yticklabels={}` s prázdými závorkami.

Rozložení značek na ose hodnot Abychom převedli rozložení značek na ose hodnot, tj. u kterých hodnot budou zobrazeny značky, musíme nejdříve zjistit minimum a maximum na ose a zapsat. Minimální hodnotu na ose získáme z atributu **MinimumScale** a zapíšeme `xmin=MinimumScale`, maximální hodnotu z atributu **MaximumScale** a zapíšeme `xmax=MaximumScale`.

Potom můžeme přejít k rozložení značek, využijeme zápisu, kdy určíme první hodnotu, druhou a poslední. Na to budeme ještě potřebovat znát vzdálenost mezi hlavními značkami a tu získáme z atributu **MajorUnit**. Zapíšeme: `ytick={MinimumScale,MinimumScale+MajorUnit,..., MaximumScale}`.

V případě, že mají být zobrazeny i vedlejší značky, zapíšeme jejich rozložení podobně jako u hlavních značek. Vzdálenost mezi vedlejšími značkami získáme z atributu **MinorUnit** a zapíšeme `minor ytick={MinimumScale,Minimum Scale+MinorUnit,...,MaximumScale}`.

Rozložení značek na ose hodnot ve 100% skládaném grafu V případě 100% typů skládaných grafů musíme na ose hodnot provést úpravu. Protože atributy MinimumScale, MaximumScale, MinorUnit a MajorUnit jsou ve 100% grafu v rozmezí 0 až 1, vynásobíme je všechny hodnotou 100. Nakonec je třeba ještě upravit popisky značek tak, aby obsahovaly znak procenta %. Zapíšeme tedy `yticklabels={}`, do závorek pak musíme vypsat pro každou značku její popisek s procenty.

Rozložení značek na ose kategorií Protože PgfPlots umí standardně používat pouze číselné hodnoty pro souřadnice, musíme pro osu kategorií u nečíselných hodnot převést kategorie na symbolické souřadnice. Použijeme pro to zápis `symbolic x coords={}`, kde do závorek zapíšeme názvy kategorií oddělené čárkou. Názvy kategorií získáme pomocí atributu grafu **SeriesCollection(i).Formula**, kde z libovolné datové řady můžeme zjistit umístění názvů kategorií v tabulce a ty poté v cyklu načíst a zapsat.

V MS Excel máme na rozdíl od osy hodnot atributy **TickLabelSpacing** a **TickMarkSpacing**, které určují vzdálenost popisků značek a značek samotných. Pomocí těchto atributů je možné nastavit zobrazovaní popisků značek tak, že mohou být u značek i mimo značky. V PgfPlots podobné nastavení

není, popisky mohou být jen u značek. Proto jsme se rozhodli převést toto nastavení takto:

- pokud je vzdálenost značek rovná 1, znamená to, že bude zobrazena pro každou kategorii značka a zapíšeme `xtick=data`,
- pokud je vzdálenost značek d větší než 1, zapíšeme každou tuto d-tou kategorii oddělenou čárkou do `xtick={}`,
- pokud je vzdálenost popisků p větší než 1 a zároveň p Mod d je 0, zapíšeme každou tutou p-tou kategorii oddělenou čárkou do příkazu `xticklabels={}`, v případě, že některá kategorie má značku, ale nemá popisek, zapíšeme místo této kategorie do příkazu samotnou čárku,
- pokud je hodnota atributu **TickLabelPosition** rovna -4142, nezobrazují se žádné popisky a zapíšeme `xticklabels={}`.

Vedlejší značky nebudeme pro osu kategorií převádět, protože v MS Excel se vedlejší značky na ose kategorií zobrazují vždy mezi hlavními značkami, které označují jednotlivé kategorie, v PgfPlots ale nemůžeme zobrazit značky pro neexistující hodnoty.

Formát osy Pro převod formátu osy využijeme kapitolu Formát čáry.

2.13 Mřížka grafu

Při převodu mřížky nejprve pro osu hodnot a osu kategorií zjistíme, kterou mřížku obsahuje a zapíšeme parametr prostředí axis pro její zobrazení (v defaultním nastavení se mřížka nezobrazuje a nemusíme zapisovat nic). V případě pruhového grafu je opět nutné prohodit pro hlavní mřížku ymajorgrids a ymajorgrids, pro vedlejší mřížku yminorgrids a yminorgrids. Převod bude vypadat takto:

- `ymajorgrids=true` pro Axes(xlValue).HasMajorGridlines=True,
- `yminorgrids=true` pro Axes(xlValue).HasMinorGridlines=True,
- `xmajorgrids=true` pro Axes(xlCategory).HasMajorGridlines=True,
- `xminorgrids=true` pro Axes(xlCategory).HasMinorGridlines=True.

V PgfPlots nelze nastavit vzhled mřížky zvlášť pro hodnotovou osu a zvlášť pro osu kategorií. Lze nastavit zvlášť vedlejší a hlavní mřížku pro obě osy najednou. Proto je třeba zvolit, v případě že by se v excelovém grafu lišil vzhled mřížek os, která osa pro nás bude pro vzhled určující. Protože mřížka grafu slouží pro zlepšení čitelnosti grafu a zejména vynášených hodnot, budeme

vzhled mřížky určovat dle mřížky hodnotové osy. Pro nastavení stylu hlavní mřížky zapíšeme do nastavení prostředí axis `major grid style={}` a vedlejší mřížky `minor grid style={}`. Do závorek pak zapíšeme oddělené čárkou nastavení formátu mřížky. Protože mřížka je tvořená pouze čárami, formát hlavní a vedlejší mřížky získáme z atributů **Axes(xlValue).MajorGridlines. Format.Line** a **Axes(xlValue).MinorGridlines.Format.Line** a převedeme dle kapitoly Formát čáry.

2.14 Bodový a spojnicový graf

V této kapitole budeme převádět bodový i spojnicový typ grafu, protože v PgfPlots se pro oba tyto typy existuje v PgfPlots jeden základní typ spojnicového grafu se značkami. Převod provedeme tak, že ve for cyklu projdeme všechny datové řady a pro každou zapíšeme příkaz `\addplot[] table[] {soubor};`. Pro každou datovou řadu tak budeme nastavovat zvlášť i parametry příkazů addplot a table. Všechny objekty a atributy datové řady budeme vztahovat k objektu SeriesCollection(i), kde i je index datové řady.

Typ grafu Nejprve zjistíme zda se jedná o graf se spojnicemi nebo pouze s body, to zahrnuje i případy bodového grafu se spojnicemi, případně spojnicového grafu bez spojnic. Pro bodový i spojnicový graf to zjistíme z parametru datové řady **Format.Line.Visible**. Pokud bude jeho hodnota False, tato datová řada je tvořena pouze body a do parametrů příkazu addplot zapíšeme `only marks`. V opačeném případě se může jednat o spojnicový graf se značkami nebo bez. To zda obsahuje značky zjistíme z atributu **MarkerStyle**, který určuje typ značek. Pokud bude mít hodnotu -4142, tato řada neobsahuje značky a zapíšeme parametr `no markers`.

Nastavení spojnic U spojnicového grafu, případně u bodového grafu se spojnicemi, budeme převádět nastavení spojnice. K tomu využijeme nastavení formátu čáry dle kapitoly Formát čáry. Tato nastavení se zapisují oddělená čárkou přímo do hranatých závorek jako parametr příkazu addplot.

Kromě formátu čáry můžeme u spojnice nastavit, zda mají být vyhlazené nebo rovné. To zjistíme z atributu **Smooth**. Pokud bude mít hodnotu True, zapíšeme `smooth`, v opačném případě nemusíme zapisovat nic.

Nastavení značek U bodového a spojnicového grafu budeme převádět i některá nastavení značek. Převést bychom chtěli typ značky, velikost značky a formát, podobně jako u spojnic.

Typ značky Pro nastavení typu značek budeme převádět hodnoty atributu **MarkerStyle**. Značky v MS Excel a v PgfPlots si přesně neodpovídají

a tak některé typy budeme převádět na jim nejpodobnější. Do parametrů příkazu addplot zapíšeme `mark=`:

- `square*` pro MarkerStyle = 1 (čtverec),
- `diamond*` pro MarkerStyle = 2 (kosočtverec),
- `triangle*` pro MarkerStyle = 3 (trojúhelník),
- `asterisk` pro MarkerStyle = 5 (čverec s hvězdou),
- `*` pro MarkerStyle = 8 (kruh),
- `+` pro MarkerStyle = 9 (čtverec s plus),
- `-` pro MarkerStyle = -4118 (krátká pomlčka) a -4115 (dlouhá pomlčka),
- `square` pro MarkerStyle = -4147 (obrázek),
- `x` pro MarkerStyle = -4168 (čtverec s x),
- hodnota MarkerStyle = -4142, znamená, že graf neobsahuje značky (viz odstavec TypGrafu)
- pro hodnotu MarkerStyle = -4105 nebudeme zapisovat nic, jedná se o automatické nastavení MS Excel a proto v PgfPlot ponecháme nastavení také automatické, ačkoli tak dostaneme pravděpodobné jiný typ značky než v MS Excel.

Velikost značky Velikost značky získáme z atributu **markerSize** a zapíšeme `mark size=markerSize`. Během převodu jsme ale zjistili, že velikost je v pgfplots 2x větší než v excelovém grafu, proto hodnotu markerSize navíc vydělíme 2.

Formát značky U značek, stejně jako u čáry grafu, bychom chtěli umět převést jejich formát. V kapitole analýza (odkaz) jsme ale zjistili, že v MS Excel nemáme pomocí VBA možnost, jak získat formát výplně či linky značky. Zjistit lze pouze barvu linky a výplně a proto převedeme alespoň tyto atributy, ostatní nastavení značek bude možné učinit pouze prostřednictvím zásahu uživatele.

Barvu linky a výplně získáme z atributů **MarkerForegroundColor** a **MarkerBackgroundColor**. Pokud má některý z těchto atributů hodnotu -1, znamená to, že barva je nastavována automaticky a v takovém případě ji nebudeme převádět, PgfPlots nastaví barvu také automaticky, která se ovšem bude pravděpodobně od barvy v MS Excel lišit. V opačném případě získané hodnoty převedeme dle kapitoly Barvy (odkaz). Barvu linky zapíšeme jako `draw={rgb,255:red,R;green,G;blue,B}` a barvu výplně značky jako `fill=`

`{rgb,255:red,R;green,G;blue,B}` a zapíšeme do složených závorek nastavení formátu značek `mark options={}`.

V MS Excel nemáme možnost, jak získat šířku linky značky, ale víme, že defaultní hodnota je 1. V PgfPlots je ale šířka linky značky defaultně rovna šířce spojnice u spojnicového grafu. Abychom částečně zachovali nastavení z MS Excel, zapíšeme do závorek nastavení formátu značek zápis `line width=1`, tím nastavíme napevno šířku linky značky. Změna tohoto nastavení bude možná opět pouze ručně.

2.15 Sloupcový a pruhový graf

Protože sloupcový a pruhový graf jsou dva typy stejného grafu, budeme popisovat jejich převod v této kapitole společně. Převod budeme provádět na sloupcovém grafu, pro pruhový graf potom postačí ve všech nastaveních navzájem vyměnit písmena x a y.

Aby PgfPlots vykreslil z dat sloupcový graf, musíme zapsat mezi parametry prostředí axis parametr `ybar` pro skupinový sloupcový graf nebo `ybar stacked` pro skládaný sloupcový graf. Mezi parametry prostředí axis budeme zapisovat i následující nastavení sloupcového nebo pruhového grafu.

Šířka sloupce Pokud chceme převést šířku sloupce do PgfPlots, musíme ji nejdříve vypočítat, protože v MS Excel nemáme žádný atribut, který by obsahoval tento parametr. MS Excel nastavuje šířku sloupce automaticky dle šířky grafu (oblasti grafu mezi osami), vzdálenosti mezi jednotlivými kategoriemi a vzdálenosti (nebo překrytí) mezi sloupci datových řád v jedné kategorii. Šířku grafu získáme z atributu **PlotArea.InsideWidth**, v případě pruhového grafu z **PlotArea.InsideHeight**. Vzdálenost mezi kategoriemi z atributu **ChartGroups(1).GapWidth** a vzdálenost mezi řadami v kategorii **ChartGroups(1).Overlap**.

Celou šířku grafu je možno vyjádřit jako součet šířek všech sloupců, všech mezer mezi kategoriemi a všech mezer mezi řadami. Protože mezera mezi řadami má kladnou hodnotu, když se řady překrývají, a zápornou, když jsou od sebe vzdáleny, budeme mezeru mezi řadami odečítat. Vzdálenost prvního sloupce první kategorie od začátku grafu a posledního sloupce poslední kategorie od konce grafu je vždy polovina vzdálenosti mezi kategoriemi. Šířku grafu označíme jako w a šířku sloupce b. Vzdálenost mezi kategoriemi g a vzdálenost mezi řadami o, ale až po vydělení hodnotou 100, protože oba atributy jsou v MS Excel udávány v procentech šířky sloupce. Počet kategorií označíme c a počet datových řad s. Šířku sloupce spočítáme potom takto:

1. šířka všech sloupců je b . c . s
2. šířka všech mezer mezi kategoriemi včetně vzdálenosti od začátku a konce grafu je b . g . c

3. šířka všech mezer mezi řadami je b . o . c . (s - 1)

4. šířka grafu w = b . c . s + b . g . c - b . o . c . (s - 1)

5. z toho vyjádříme šířku sloupce b = w / (b . c . (s + g - o . (s - 1)))

V případě skládaného sloupcového grafu je výpočet jednodušší, protože pro každou kategorii máme pouze jedne sloupec obsahující všechny řady. Spočítáme b = w / (c . (1 + g)).

Šířku sloupce zaokrouhlíme na celé číslo a zapíšeme jako `bar width=b pt`.

Mezera mezi sloupci v kategorii V předchozím odstavci jsme použili atribut **ChartGroups(1).Overlap** pro výpočet šířky sloupce, tuto šířku sloupce použijeme pro nastavení mezery, případně překrytí sloupců jednotlivých řad v kategorii. Vzdálenost d mezi sloupci spočítáme takto (použijeme stejné označení proměných jako v předchozím odstavci): d = o . b . (-1). Hodnotou -1 násobíme proto, že záporná hodnota atributu Overlap v MS Excel znamená mezeru a kladná překrytí, zatímco v PgfPlots naopak. Výsledek zaokrouhlíme na celé číslo a zapíšeme jako hodnotu parametru `ybar=d pt`.

Vzdálenost mezi kategoriemi a od začátku grafu V PgfPlots musíme ještě nastavit vzdálenost v mezi jednotlivými kategoriemi a vzdálenost kategorie od začátku grafu. Pro nastavení vzdálenosti v mezi kategoriemi vydělíme šířku grafu počtem kategorií a zaokrouhlíme na celé číslo: v = w / c. Potom zapíšeme jako `x=v pt`.

Nakonec spočítáme vzdálenost s kategorie od začátku grafu. Protože vzdálenost je od začátku grafu do středu kategorie, postačí nám vydělit vzdálenost mezi kategoriemi 2 a zaokrouhlit: s = v / 2. Zapíšeme řádek `enlarge x limits={abs=s pt}`.

Formát sloupců U sloupcového nebo pruhového grafu budeme nastavovat formát výplně sloupců a formát linky sloupců. Nastavení formátu získáme z atributu **SeriesCollection(i).Format.Fill** a **SeriesCollection(i).Format.Line**, kde i je index datové řady. nastavení formátu budeme zapisovat do parametrů příkazu addplot. Postup převodu je popsán v kapitolách Formát výplně a Formát čáry, u formátu linky musíme vzít v úvahu, že v defaultním nastavení se linka v MS Excel nezobrazuje, zatímco v PgfPlots ano, proto pokud není v MS Excel nastavena, musíme zapsat do parametrů `draw=none`.

2.16 Plošný graf

Při převodu plošného grafu zapíšeme do parametrů prostředí axis řádek `area style`. V případě skladáného grafu přidáme řádek `stack plots=y`, protože plošný graf má hodnoty vždy na ose y. U plošného grafu musíme ještě na

konci každého příkazu addplot za zápisem, který určuje umístění dat, zapsat příkaz `\closedcycle` a až za ním zapíšeme středník.

Nastavení formátu ploch grafu je stejné jako nastavení sloupců u sloupcového grafu v kapitole Sloupcový a pruhový graf, odstavec Formát sloupců.

2.17 Povrchový graf

Povrchový graf je typ prostorového grafu se třemi osami. Zatímco v MS Excel graf převádí tabulkové hodnoty na spojitý povrch tvořený trojúhelníky, který je poté obarven dle rozpětí jednotlivých hodnot podobně jako vrstevnicová mapa, tak v PgfPlots jsou jednotlivé kordináty propojeny pomocí obdélníkových ploch, které mohou být obarveny každá zvlášť dle hodnot nebo společně přechodem. Proto se bude při převodu z MS Excel výsledný graf v PgfPlots více lišit, než při převodu dvourozměrných grafů. Jiné jsou i možnosti nastavení vzhledu.

Pro vykreslení prostorového grafu zapíšeme příkaz `\addplot3[] table[] {soubor}`.

Podtypy povrchového grafu Základní povrchový graf zobrazíme pomocí zápisu parametru `surf` do hranatých závorek příkazu addplot3. Povrchový drátěný graf pak pomocí parametru `mesh`. Oba tyto podtypy povrchového grafu můžeme zobrazit jako obrysové grafy. PgfPlots nemá na tyto podtypy speciální parametr, ale můžeme využít toho, že obrysový graf je pohledem na povrchový graf shora, pohled shora nastavíme pomocí zápisu `view{0}{90}` do parametrů prostředí axis.

Převod dat Povrchový graf vyžaduje na rozdíl od dvourozměrných typů grafů jinak uspořádanou tabulku dat, tak jak je popsáno v kapitole Analýza – Povrchový graf. Pro převod dat můžeme využít již vytvořenou tabulku dat dle kapitoly Převod dat. Tuto tabulku budem procházet ve dvou vnořených cyklech, v prvním procházíme datové řady (sloupce) a v druhém kategorie (řádky). Pro každou hodnotu zapíšeme jeden řádek, ve kterém budou odděleny mezerou název kategorie, název datové řady a hodnota. Poté, co zapíšeme hodnoty a kategorie celé jedné datové řady tj. projdeme jeden vnořený cyklus, zapíšeme prázdný řádek.

Do parametrů příkazu table zapíšeme umístění dat v datovém souboru pro jednotlivé osy, využijeme pro to zápis dle indexu sloupce. Protože jsme do prvního sloupce zapsali název datové řady, do druhého název kategorie a do třetího vlastní hodnoty, zapíšeme `table[y index=0, x index=1, z index=2]`.

Nastavení os Při nastavení os musíme vzít v úvahu, že osou hodnot je nyní osa z a osa y je hloubková osa, na kterou se vynáší řady. K hloubkové ose budeme přistupovat pomocí objektu **Axes(xlSeries)** a provedeme stejné

nastavení jako pro osu x (osu kategorií) dle kapitoly Osy. Pro osu z provedeme stejná nastavení jako pro osu y (osu hodnot) tak, že ve všech nastaveních vyměníme písmeno y za písmeno z. Nastavení osy x zůstává stejné.

Nastavení pohledu Při zobrazovaná prostorového povrchového nebo drátěného grafu budeme převádět úhel, pod kterým graf pozorujeme – elevaci a jeho natočení – rotaci. Rotaci podle osy z ve stupních získáme z atributu grafu **Rotation** a odklon od vodorovné osy z atributu **Elevation**. Toto nastavení zapíšeme do prostředí axis jako jeho parametr zápisem `view{Rotation}{Elevation}`.

Další nastavení se v MS Excel a PgfPlots liší. V MS Excel je možnost nastavení perspektivy, která v PgfPlots (zatím) chybí. Dále se v MS Excel zobrazovaná velikost grafu při různé rotaci a elevaci mění, přizpůsobuje se velikosti celé oblasti grafu, ačkoli atributy grafu **PlotArea.InsideWidth** a **PlotArea.InsideHeight** zůstavájí stejné a nemáme tak možnost získat skutečnou aktuální velikost oblasti grafu mezi osami a nemůžeme ji tedy přesně do PgfPlots převést. Budeme tedy převádět pouze rotaci a elevaci a graf bude mít šířku a výšku dle atributů z MS Excel.

Legenda grafu Povrchový graf v MS Excel nemá legendu zobrazující datové řady, ale legendu, která přiřazuje jednotlivým hodnotám barvy, v PgfPlots legenda u povrchového grafu není, ale máme možnost zobrazit tzv. colorbar, kde je spektrum s hodnotami. Pokud bude mít graf v MS Excel zobrazenou legendu – atribut **HasLegend** bude True, nebudeme zapisovat příkaz `legend{}` pro zobrazení legendy datových řad, ale můžeme místo toho v PgfPlots zobrazit colorbar pomocí zápisu `colorbar` do parametrů prostředí axis.

Formát oblasti grafu Ve dvourozměrných grafech nastavujeme formát výplně oblasti grafu mezi osami, v trojrozměrném grafu je tato oblast tvořena stěnami a podstavou, proto místo objektu PlotArea budeme převádět objekty **BackWall** a **Floor**. V PgfPlots ale nemůžeme nastavit zvlášť formát bočních stěn a podstavy. Zvolíme tedy pouze jeden z těchto objektů, například formát bočních stěn **BackWall.Format.Fill** a dále převedeme dle kapitoly Oblast grafu mezi osami.

V případě obrysového a obrysového drátěného grafu nebudeme formát oblasti grafu nastavovat, protože při pohledu shora zabírá graf celou podstavu, zatímco stěny nejsou vidět.

2.18 Výsečový graf

Při převodu výsečového grafu budeme používat balíček Pgf-Pie, který nabízí základní zobrazení výsečového grafu. Do prostředí tikzpicture zapíšeme pří-

kaz `\pie [] {}`, kde do hranatých závorek uvedeme parametry grafu a do složených data.

Nastavení dat na rozdíl od předchozích grafů budeme data pro výsečový graf zadávat přímo do zdrojového souboru a nikoli do externího datového souboru. V cyklu procházíme jedinou datovou řadu výsečového grafu a do složených závorek zapisujeme oddělené čárkou pro každou kategorii hodnotu, lomítko a název kategorie. Aby se data zobrazila v podobě kruhu s hodnotami, musíme ještě zapsat do parametrů příkazu pie `sum=auto`.

V případě, že bychom chtěli místo hodnot zapsat procentuální vyjádření podílu hodnot na celku, musíme hodnoty převést na procenta. Nejprve si spočítáme koeficient, který představuje jedno procento, to je suma všech hodnot vydělená stem. Tímto koeficientem pak podělíme jednotlivé hodnoty, které musíme nakonec zaokrouhlit na celé číslo, protože Pgf-Pie nedovede zobrazit v podobě procent jiné než celočíselné hodnoty.

Legenda a popisky dat Legenda ve výsečovém grafu zobrazuje kategorie. V Pgf-Pie mohou být názvy kategorií zobrazeny buď jako popisky přímo v grafu, nebo ve formě legendy, nemůžeme tedy zobrazit obojí najednou, tak jako je to možné v MS Excel. Proto pokud má být zobrazena legenda a tedy hodnota atributu **HasLegend** je True, zobrazíme názvy kategorií ve formě legendy. Zapíšeme do parametrů příkazu pie `text=legend`.

Pokud graf nemá legendu, mohou být kategorie v Pgf-Pie zobrazeny ještě jako popisky dat přímo na grafu, to zjistíme z atributu **SeriesCollection(1). DataLabels.ShowCategoryName** pokud bude True zobrazíme názvy kategorií na grafu. Ve výsečovém grafu zjistíme umístění popisků dat z atributu **SeriesCollection(1).DataLabels.Position** a převedeme takto:

- pozici 4, 3 a -4108 na `text=inside`,
- ostatní umístění na `text=outside`.

Velikost výsečového grafu Protože výsečový graf zabírá celý prostor oblasti grafu mezi osami, můžeme zjistit jeho průměr v bodech z atributu **PlotArea.InsideWidth**. V PgfPie se ale nastavuje velikost grafu jako poloměr v centimetrech, musíme tedy body převést na centimetry. V MS Excel představuje 72 bodů 1 palec, průměr v palcích je PlotArea.InsideWidth / 72 a poloměr PlotArea.InsideWidth / 144. Jeden palec má 2,54 cm, poloměr v centimetrech zapíšeme tedy takto: `radius=(PlotArea.InsideWidth / 144) * 2,54`.

Formát grafu V Pgf-Pie můžeme nastavit pouze barvu výsečí, jiné nastavení formátu není k dispozici. Barvu jednotlivých výsečí získáme z atributu

SeriesCollection(1).Points(i).Format.Fill.ForeColor, kde i je index výseče dle pořadí kategorie, kterou výseč představuje. Barvy výsečí převedeme dle kapitoly Barvy a zapíšeme oddělené čárkou do parametru `color={}`.

Kapitola 3

Návrh aplikace

V této kapitole se budeme věnovat návrhu aplikace. Jak bylo popsáno v úvodu v kapitole stanovení cílů práce, jsou na aplikaci kladeny tyto požadavky:

- umožnit jednoduše převést graf dle požadavků na konverzi,
- poskytnout intuitivní a uživatelský příjemné rozhraní,
- umožnit vlastní nastavení konverze,
- uložit nastavení konverze, aby uživatel nemusel nastavení pokaždé opakovat,
- poskytnout případnou pomoc.

3.1 Diagram případů užití

Nejprve je třeba popsat funkcionalitu systému, to co má systém umět. K tomu slouží diagram případů užití (Use case diagram), který popisuje chování systému tak, jak ho vidí uživatel. Diagram případů užití se skládá z aktérů, případů užití a vztahů mezi nimi.

Aktér reprezentuje objekt mimo systém, který se sytémem interaguje. V diagramu případů užití naší aplikace bude pouze jeden aktér a to uživatel aplikace, pro kterého je aplikace navrhována.

Případy užití specifikují část funkcionality systému – použití aplikace aktérem. Následují scénáře jednotlivých případů užití, jednotlivé scénáře budou obsahovat:

- vstupní podmínky, které musí být splněny, aby mohl být případ užití započat,
- tok událostí,

- následné podmínky – podmínky, které jsou po úspěšném ukončení případu pravdivé,
- alternativní tok – alternativy k hlavnímu toku,
- chybový tok – situace, které jsou považovány za chybu.

Případ užití - spuštění aplikace

Vstupní podmínky: Uživatel chce začít pracovat s aplikací.

Tok událostí:

1. Uživatel spustí v pracovním sešitu aplikaci jako makro MS Excel.
2. Otevře se formulář s uživatelským rozhraním aplikace.
3. Aplikace načte konfigurační soubor pro nastavení konverze.

Následné podmínky Je zobrazeno uživatelské rozhraní aplikace s načtenou konfigurací, v rozbalovacím okně sešitů je zobrazen aktivní sešit.

Chybový tok: Pokud nebylo možno načíst konfigurační soubor, systém ohlásí chybu a pro nastavení konverze bude použito defaultního nastavení.

Případ užití - vybrat graf

Vstupní podmínky: Uživatel má spuštěnu aplikaci; protože pracovní sešit může mít více listů, chce si uživatel vybrat i list a graf na něm umístěný.

Tok událostí:

1. Uživatel vybere list.
2. Uživatel vybere graf na vybraném listu.
3. Aplikace zobrazí typ vybraného grafu a informaci, zda se jedná o podporovaný typ.

Alternativní tok:

- K bodu 1: uživatel nebude měnit zobrazený aktivní list, pokračuje přímo bodem 2 hlavního toku.
- K bodu 2: pokud list neobsahuje žádný graf, aplikace toto oznámí uživateli a uživatel může opakovat bod 1.

Následné podmínky: Uživatel může vybraný graf konvertovat.

Případ užití - změnit cílový adresář

Vstupní podmínky: Uživatel chce změnit cílový adresář, kam budou ukládány datové soubory grafu.

Tok událostí:

1. Uživatel stiskne tlačítko pro výběr cílového adresáře.
2. Pomocí dialogového okna operačního systému vybere uživatel cílový adresář.
3. Uživatel stiskne v dialogovém okně Uložit.
4. Aplikace uloží změnu adresáře do konfiguračního souboru.

Alternativní tok

- K bodu 3: uživatel stiskne v dialogovém okně Storno. Cesta k cílovému adresáři zůstává beze změn.

Následné podmínky V textovém okně je zobrazen nový cílový adresář, změna je uložena v konfiguračním souboru.

Chybový tok: Pokud nebylo možno změnu uložit, aplikace oznámí chybu.

Případ užití - spustit konverzi

Vstupní podmínky: Uživatel vybral graf k převodu – případ užití Vybrat graf a chce provést jeho konverzi.

Tok událostí:

1. Uživatel stiskne tlačítko pro provedení konverze .
2. Aplikace provede převod grafu.
3. Aplikace zobrazí v textovém okně makra systému LaTeX.
4. Aplikace oznámí ukončení konverze.

Následné podmínky Uživatel si může z textového okna zkopírovat makra do svého zdrojového souboru.

Chybový tok: Pokud nebylo možno graf převést, aplikace oznámí chybu a její důvod.

Případ užití - změnit nastavení konverze

Vstupní podmínky: Uživatel chce změnit nastavení konverze.

Tok událostí:

1. Uživatel stiskne tlačítko pro nastavení konverze.
2. Aplikace otevře okno s nastavením konverze.
3. Uživatel nastaví konverzi pomocí prvků formuláře, jako jsou zaškrtávací pole, textová pole apod.
4. Uživatel stiskne tlačítko pro uložení nastavení.
5. Aplikace uloží změnu nastavení do konfiguračního souboru.

Alternativní tok

- K bodu 4: uživatel stiskne tlačítko pro zrušení provedených změn, nastavení zůstane beze změn.

Následné podmínky Konverze může proběhnout dle nové konfigurace. V konfiguračním souboru je uloženo zvolené nastavení.

Chybový tok: Pokud nebylo možno nastavení uložit do konfiguračního souboru, aplikace oznámí chybu.

Případ užití - zobrazit nápovědu

Vstupní podmínky: Uživatel si chce prohlédnout nápovědu k práci s aplikací.

Tok událostí:

1. Uživatel stiskne tlačítko pro zobrazení nápovědy.
2. Aplikace otevře okno s nápovědou.
3. Uživatel uzavře nápovědu stisknutím tlačítka.

Následné podmínky Nápověda je zobrazena jako vyskakovací okno, pro další práci s aplikací je nutno okno uzavřít.

3.2 Návrh uživatelského rozhraní

Aplikace a její uživatelské rozhraní (GUI) bude realizováno formulářem, na který umisťujeme ovládací prvky známé z operačního systému či jiných aplikací, jako jsou tlačítka, textová pole, rozbalovací pole atd. Potřebné prvky vybereme tak, aby co nejlépe vyhovovaly popsaným případům užití systému:

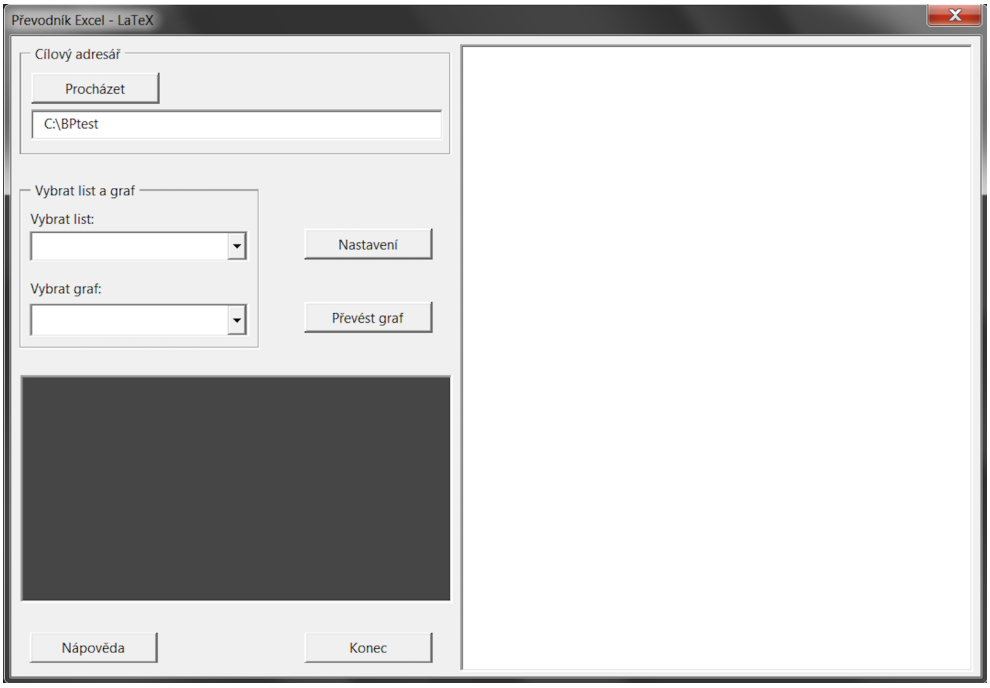
- pro výběr listu a grafu použijeme rozbalovací seznamy,
- pro výběr cílového adresáře, kam budou umístěny datové soubory, použijeme tlačítko Procházet,
- umístění cílového adresáře bude zobrazeno pomocí textového pole, kam může uživatel zadat cestu i přímo,
- pro provedení konverze bude na formuláři tlačítko Převést graf,
- výsledek konverze bude zobrazen v textovém poli, odkud si ho uživatel bude moci zkopírovat,
- nápověda se zobrazí po stisknutí tlačítka Napověda, uzavře se tlačítkem OK.
- formulář nastavení se otevře po stisknutí tlačítka Nastavení,
- na formuláři nastavení budou volby, zda chceme nastavení použít nebo ne, umožněny pomocí zaškrtávacích polí, nastavení, kde je třeba vyplnit hodnotu, budou textová pole,
- na formuláři nastavení bude tlačítko OK pro potvrzení a uložení nastavení a vrácení na původní formulář, tlačítko Storno pak bude pro návrat beze změn
- hlášení o průběhu konverze se budou zobrazovat v textovém poli
- aplikace se ukončí tlačítkem Konec.

Otestování použitelnosti

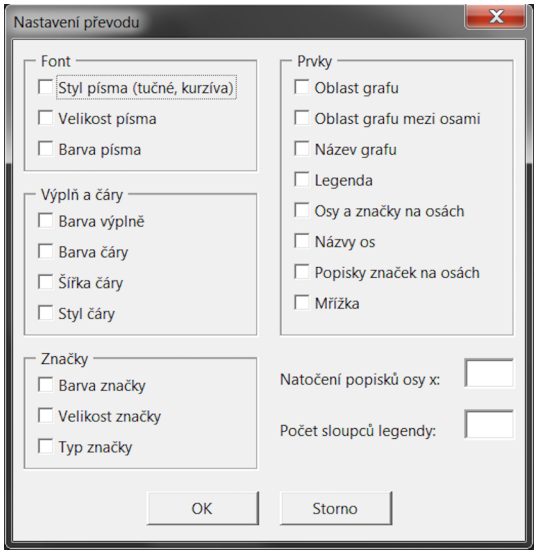
Návrh GUI můžeme otestovat pomocí Nielsenovy heuristické analýzy[**?**], která se skládá z deseti základních pravidel, jež by měly být splněny, aby byla aplikace použitelná.

Viditelnost stavu systému Uživatel musí vždy vědět, v jakém stavu se aplikace nachází a co provádí. K tomuto v aplikaci převodníku slouží textové pole, kde jsou zobrazovány informace o stavu.

Obrázek 3.1: Návrh GUI aplikace



Obrázek 3.2: Nastavení aplikace



Shoda mezi systémem a reálným světem Práce se systémem musí odpovídat reálnému světu. Toto pravidlo se vztahuje především ke vzhledu aplikace, například že tlačítko vypadá jako tlačítko apod. Toto je v naší aplikaci

splněno.

Uživatelská kontrola a svoboda Systém musí poskytovat uživateli možnost vrátit se zpět či přerušit akci. V případu výběru cílového adresáře to řeší dialogové okno operačního systému, v případě použití formuláře nastavení se uživatel může vrátit zpět beze změn použitím tlačítka Zrušit.

Konzistence a standardizace Systém by měl být konzistentní vzhledově i co se týče ovládání, měl by se držet obecných zvyklostí, používat standardní ovládací prvky. V aplikaci je to zajištěno použitým prostředím VBA, které využívá standardní prvky operačního systému a používáním obvyklých popisků tlačítek jako Procházet, OK, Storno apod.

Prevence chyb Systém by měl předcházet chybovým stavům, V aplikaci je to zajištěno upozorněním v těchto případech:

- uživatel nevybral graf a stiskl tlačítko Převést graf,
- uživatel nechal prázdné textové pole s cílový adresářem,
- v nastavení úhlu natočení popisků osy uživatel zadal jinou hodnotu než číslo mezi -90 a 90,
- v nastavení počtu sloupců legendy zadal uživatel jiné číslo než 1 až počet datových řad

Rozpoznání místo vzpomínání Uživatel by měl být během používání aplikace co nejméně zatížen, potřebné akce a informace by měly být v systému snadno dosažitelné, nepotřebné naopak zůstat skryté. V aplikaci je toto pravidlo zajištěno tím, že nastavení převodu se zobrazí pouze po stisknutí tlačítka Nastavení, pokud uživatel nechce nic nastavovat, zůstává skryto.

Flexibilní a efektivní použití Systém musí být efektivní a jednoduchý pro použití, zároveň musí poskytovat dostatečný počet voleb pro pokročilé uživatele. Aplikace je navržena jako velmi jednoduchá a umožňuje rychlé převedení grafu, pokud chce, může uživatel využít nabídky Nastavení.

Estetický a minimalistický design Systém by měl zobrazovat co nejméně informací a voleb, tak aby práce v něm byla přehledná a rychlá. Aplikace je navržena jako minimální a graficky jednoduchá.

Pomoc uživateli poznat, pochopit a vzpamatovat se z chyb Systém by měl uživateli poskytnout smysluplné chybové zprávy a možnost chyby opravit, zároveň by měl uživatele informovat o možném řešení problému. Toto je v aplikaci zajištěno chybovými zprávami i s návrhem řešení chyby.

Nápověda a návod Systém musí poskytovat nápovědu tam, kde ji uživatel očekává a kdy je nejvíce potřeba. Aplikace obsahuje nápovědu pro práci s aplikací po stisknutí tlačítka Nápověda a také kontextovou nápovědu po najetí myší na příslušné ovládací prvky.

Kapitola 4

Testování

Funkci převodníku jsme otestovali na několika příkladech grafů. U každého testu jsou dva obrázky, první je vygenerovaný obrázek grafu z MS Excel, druhý je pak převedený graf v LaTeXu. Nejedná se o vložený obrázek, tento graf je generován přímo ze zdrojového souboru této práce, kam byla vložena makra vytvořená převodníkem a z převodníkem vytvořených datových souborů.

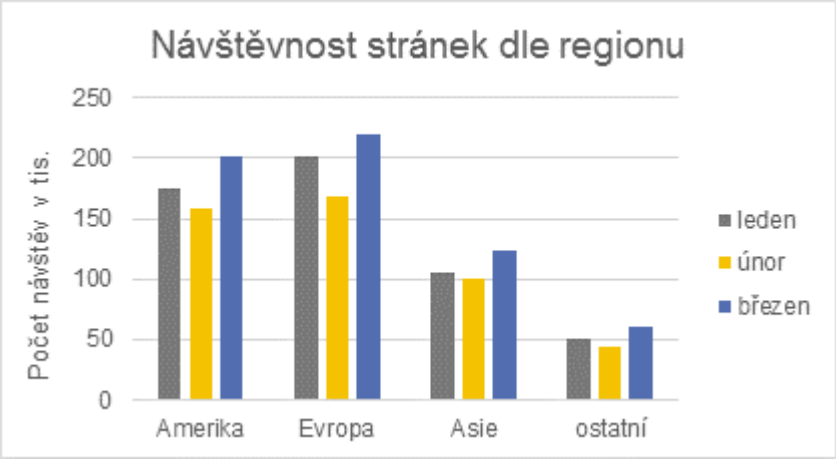
Protože příslušná makra jsou poměrně dlouhá, nebudeme je do práce kopírovat, zájemce si je může prohlédnout ve zdrojovém souboru této práce umístěném na přiloženém CD.

Testování probíhalo na verzi systému MS Office Excel 2013, v jiných verzích může převodník podávat mírně odlišné výsledky. Každý test je na zvláštní stránce, aby čtenář mohl porovnat originální a převedený graf.

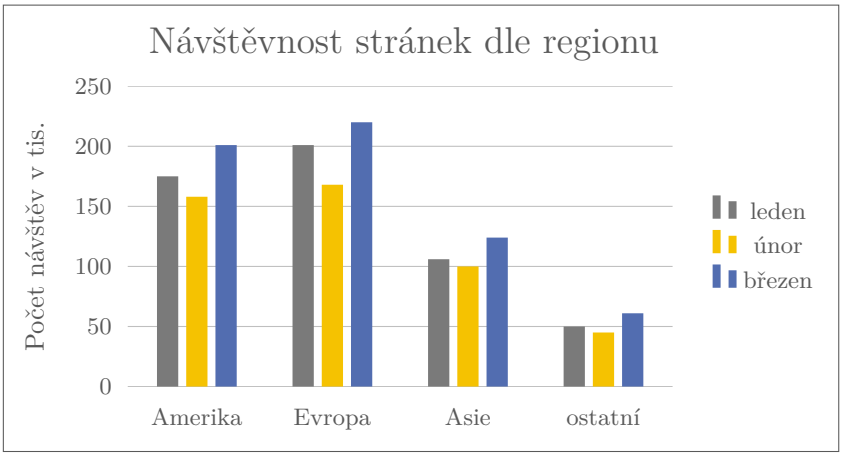
4.1 Testování převodu sloupcového skupinového grafu

V tomto testu převádíme sloupcový skupinový graf. V tomto grafu jsme pozměnili překrytí řad tak, aby v jedné kategorii byla mezi sloupci mezera. Ostatní nastavení grafu byla ponechána beze změny.

Obrázek 4.1: Sloupcový graf v MS Excel



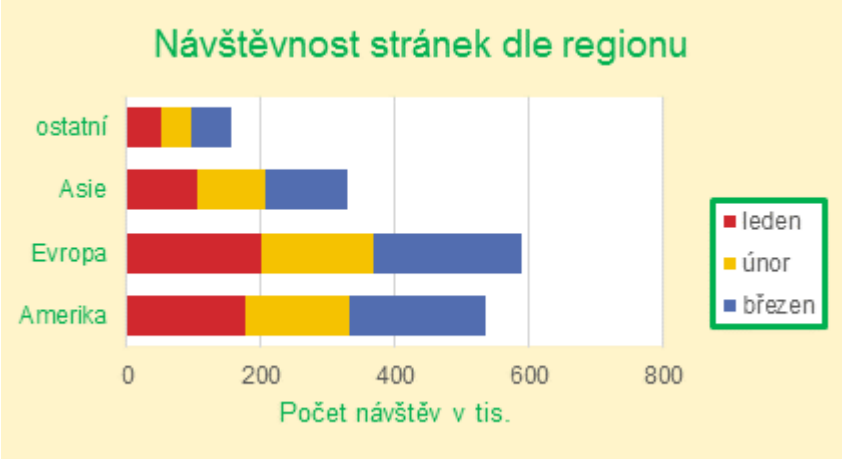
Obrázek 4.2: Sloupcový graf převedený do LaTeXu



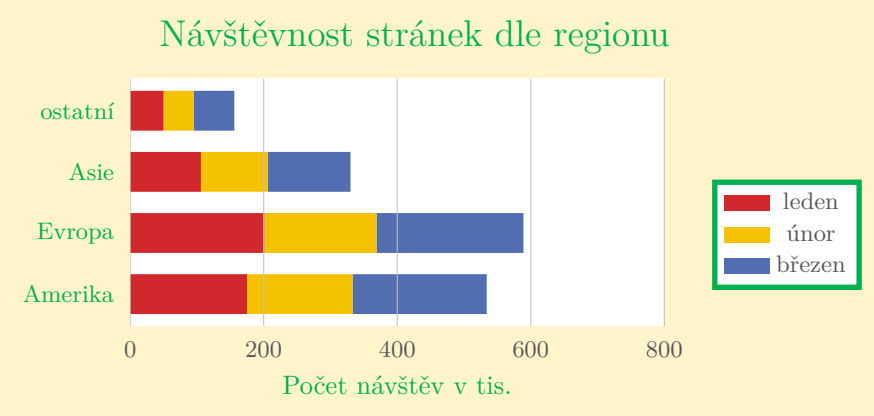
4.2 Testování převodu pruhového skládaného grafu

V tomto testu jsme otestovali kromě převodu skládaného grafu také změnu barvy názvu grafu, popisky na ose kategorií, výplně oblasti grafu a ohraničení legendy.

Obrázek 4.3: Pruhový skládaný graf v MS Excel



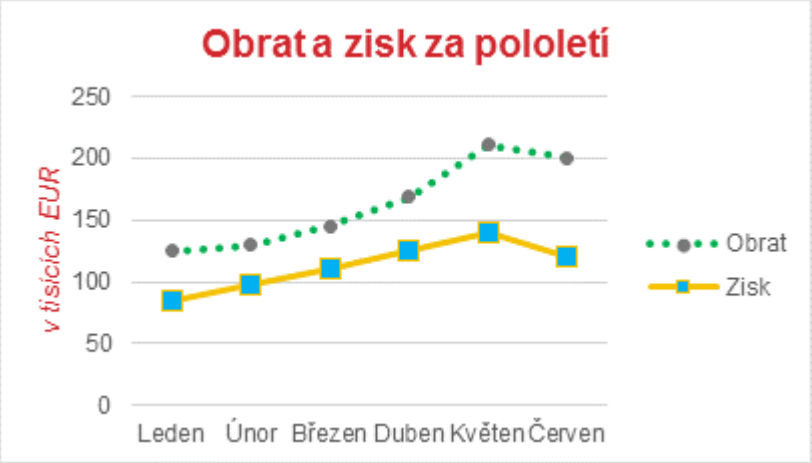
Obrázek 4.4: Pruhový skládaný graf převedený do LaTeXu



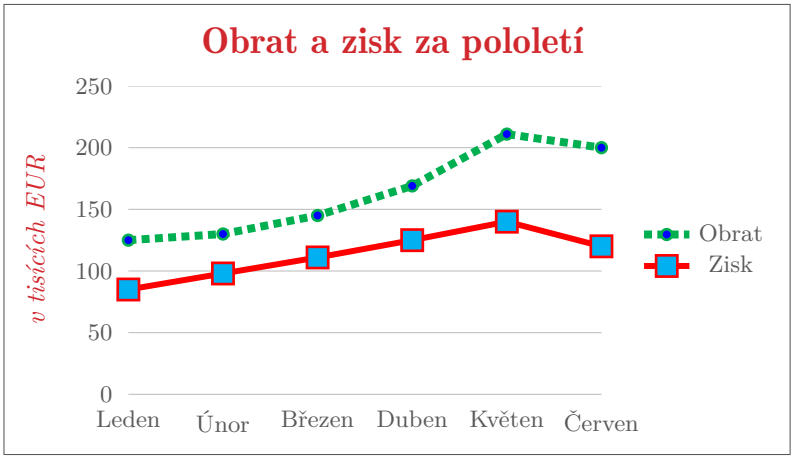
4.3 Testování převodu spojnicového grafu

V tomto testu převádíme spojnicový graf se značkami. V datové řadě Obrat byla nastavena čára na tečkovanou a zelenou a značky byly nechány nastavené automaticky, v datové řadě Zisk tomu bylo naopak, čára byla ponechána nastavena automaticky a značky byly nastaveny na modrou barvu výplně.

Obrázek 4.5: Spojnicový graf se značkami v MS Excel



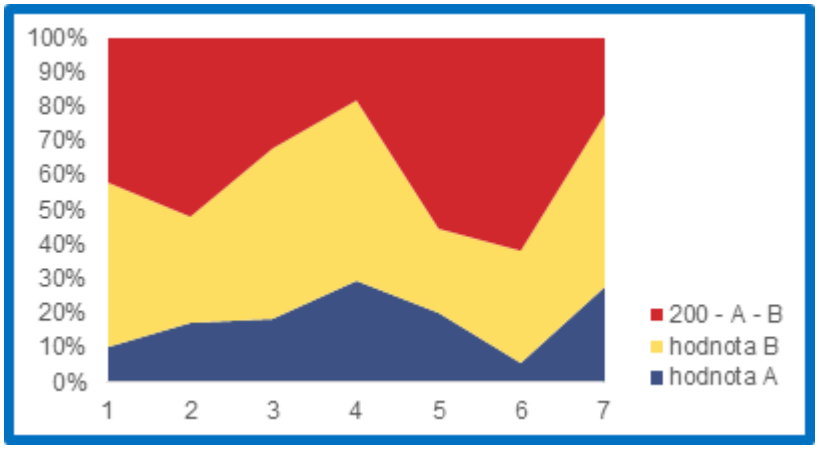
Obrázek 4.6: Spojnicový graf se značkami převedený do LaTeXu



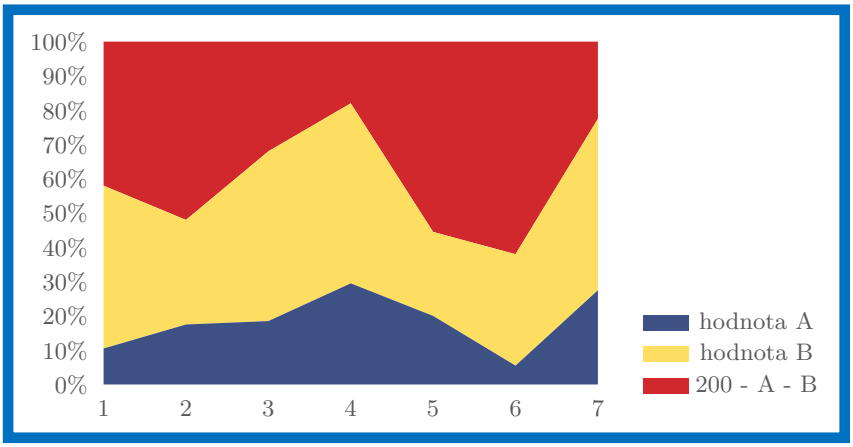
4.4 Testování převodu plošného grafu

V tomto testu převádíme plošný 100% skládaný graf.

Obrázek 4.7: Plošný 100% skládaný v MS Excel



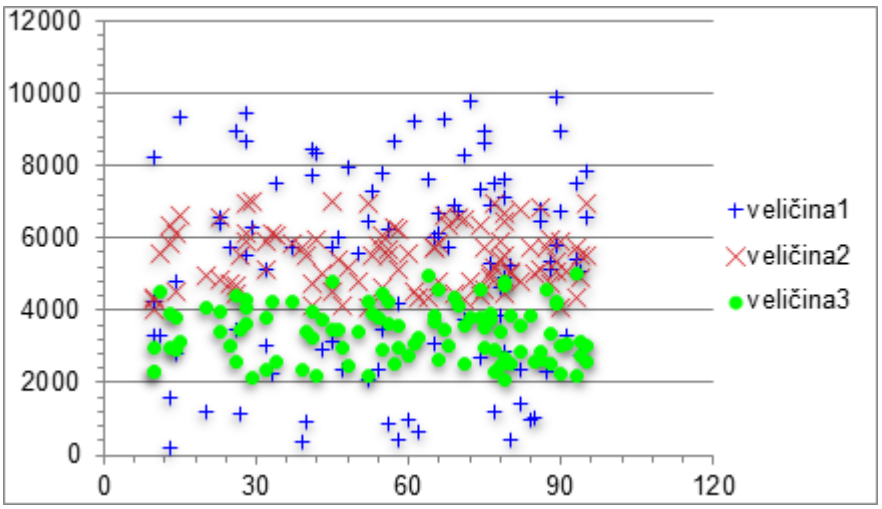
Obrázek 4.8: Plošný 100% skládaný graf převedený do LaTeXu



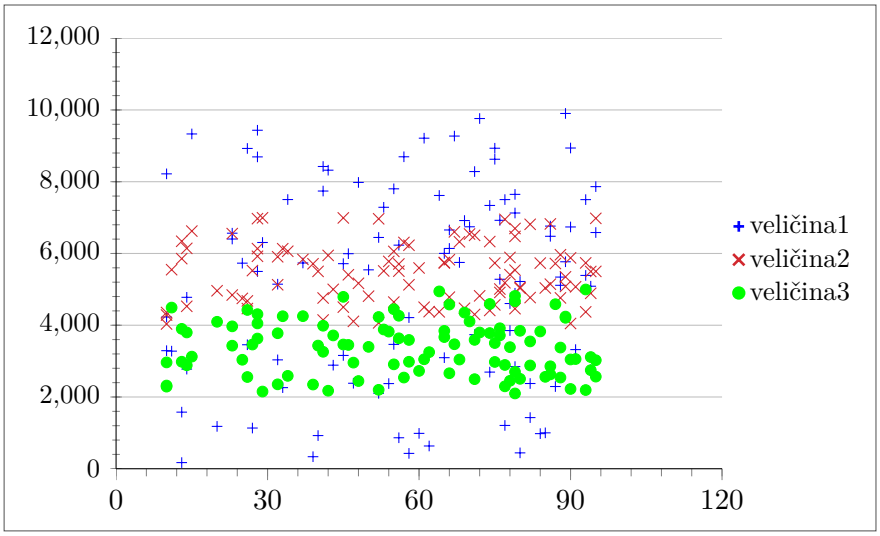
4.5 Testování převodu bodového grafu

Zde testujeme bodový graf. Pro test byly vygenerovány náhodné datové body.

Obrázek 4.9: Bodový graf v MS Excel



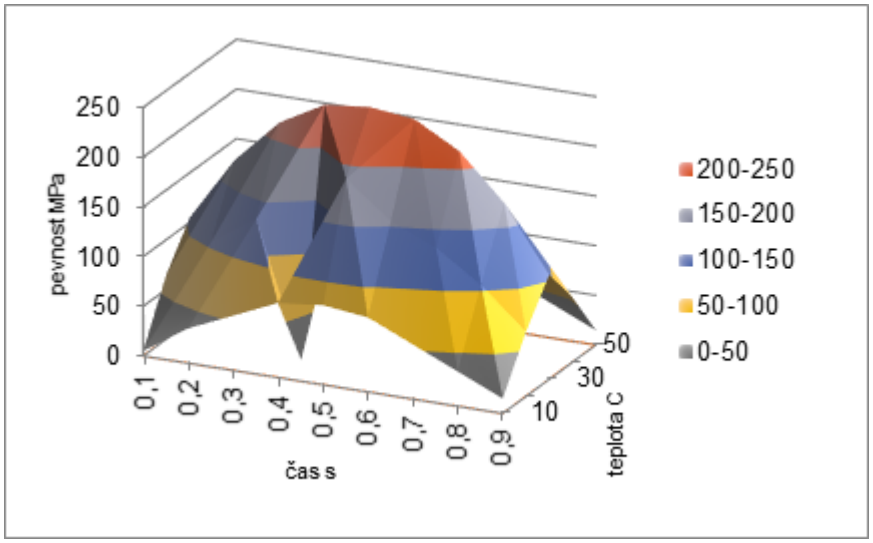
Obrázek 4.10: Bodový graf převedený do LaTeXu



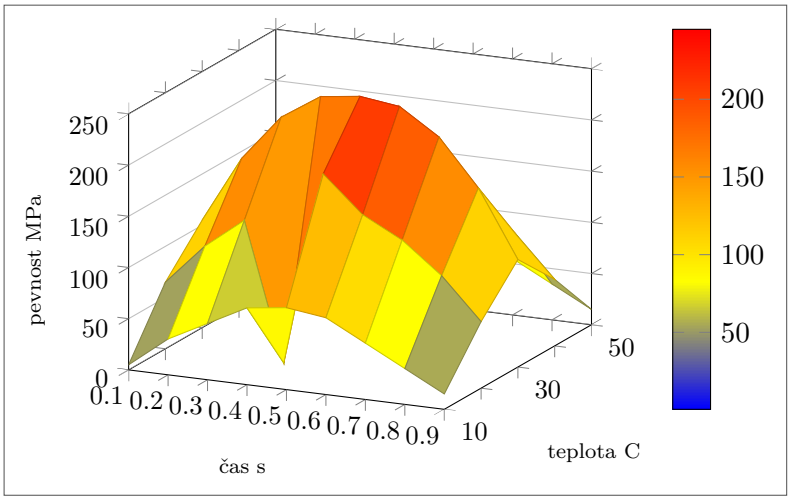
4.6 Testování převodu povrchového grafu

V tomto testu je převáděn povrchový graf, zde jsou patrné největší rozdíly při převodu – jiném pojetí povrchového grafu v MS Excel a PgfPlots.

Obrázek 4.11: Povrchový graf v MS Excel



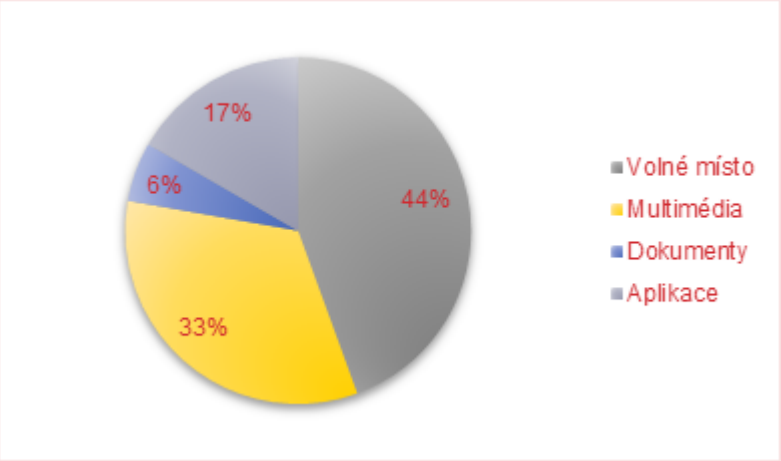
Obrázek 4.12: Povrchový drátěný graf převedený do LaTeXu



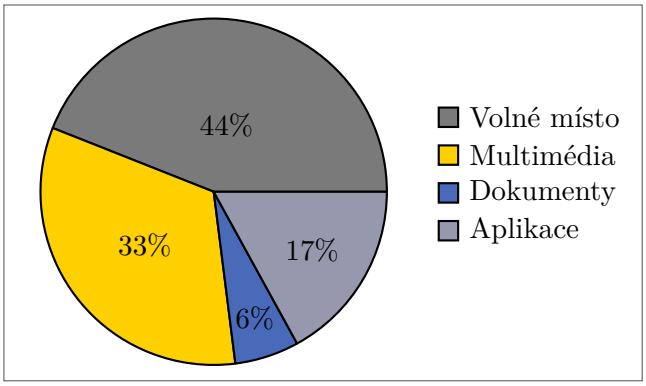
4.7 Testování převodu výsečového grafu

V tomto testu je převáděn výsečový graf. Jsou patrná omezení balíčku Pgf-Pie.

Obrázek 4.13: Výsečový graf v MS Excel



Obrázek 4.14: Výsečový graf převedený do LaTeXu



Závěr

V této práci jsme se nejprve seznámili s grafem v prostředí MS Excel, jeho prvky a typy. Poté jme analyzovali možnosti zobrazení grafů v prostředí LaTeX. Zjistili jsme, že pro zobrazení grafů v prostředí LaTeXje třeba rozšiřujících balíčků, avšak nejsou k dispozici balíčky pro všechny typy grafů, které nabízí MS Excel. S použitím balíčků PgfPlots a PgfPie jsme ale byli schopni zobrazit tyto nejpoužívanější typy grafů v MS Excel, které jsme se také rozhodli převést:

- sloupcový a pruhový graf
- bodový graf
- spojnicový graf
- plošný graf
- povrchový graf
- výsečový graf

Protože grafy v MS Excel a jejich prvky mají mnoho možností nastavení a ne všechna je možné převést, museli jsme se rozhodnout, která nastavení budeme převádět a která ne. Rozhodli jsme se převést zejména ta, která jsou důležitá z hlediska podání informace grafem o zobrazovaných datech, ale i ta, která umožní uživateli si graf do určité míry přizpůsobit.

Pro vybrané typy grafů a jejich prvky a nastavení jsme pak provedli analýzu jejich objektové reprezentace v MS Excel tak, abychom je mohli převést do maker rozšiřujících balíčků.

Poté jsme provedli návrh konverze. Převodník převádí výše uvedené typy grafů včetně jejich skládaných a 100% skládaných variant. Převodník má samozřejmě své limity podle možností zobrazení grafů v LaTeXu.

Nejčastějším problémem, který vychází z objektového modelu MS Excel je, že nemáme přístup k nastavení, kde je ponechána automatická volba, například automatické nastavení barev některých prvků, automatický typ značky u

bodového grafu apod. Tento problém jsme vyřešili tím, že u prvků, kde je zvoleno automatické nastavení v MS Excel, necháme automaticky nastavit tyto prvky také v LaTeXu.

Nakonec jsme provedli návrh aplikace a její otestování. Převodník převádí uvedené typy grafů a plní tak úkol jednoduché konverze grafů z MS Excel do systému LaTeX. Pokud bychom chtěli převádět i další typy grafů, bylo by nutné navrhnout vlastní rozšiřující balíčky pro tvorbu těchto grafů, to by bylo ale již nad rámec této práce. Více práce by bylo nutné věnovat také formátování vzhledu grafu, pokud bychom chtěli, aby převedené grafy byly co nejvěrnější. Takový převod není ale vždy žádoucí, grafy by se měly svým formátem přizpůsobit spíše vzhledu LaTeXového dokumentu.

Příloha **A**

Seznam použitých zkratek

GUI Graphical user interface

VB Visual Basic

VBA Visual Basic for Application

MS Excel Microsoft Office Excel

Příloha **B**

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `excel` ..... adresář s převodníkem
  - `converter.xlsm` ..... sešit s makry převodníku
  - `config.txt` ..... konfigurační soubor převodníku
- `thesis` ..... zdrojová forma práce ve formátu LaTeX
  - `img` ..... obrázky použité v práci
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF