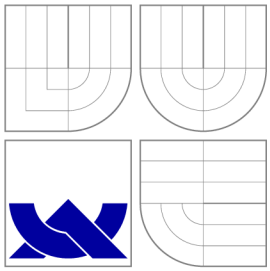
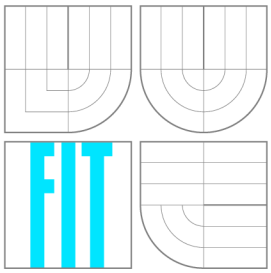


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

AUTOMATICKÉ VYČÍSLOVÁNÍ CHEMICKÝCH ROVNIC Z FOTOGRAFIE

AUTOMATIC QUANTIFICATION OF CHEMICAL EQUATIONS FROM PHOTOS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — MARTIN KRAINA
AUTHOR

VEDOUCÍ PRÁCE — Ing. VÍTĚZSLAV BERAN, Ph.D.
SUPERVISOR

BRNO 2016

Abstrakt

Cílem této práce je návrh a implementace aplikace pro mobilní platformu Android, která uživateli poskytuje nástroj automatizovaného vyčíslování chemických rovnic. Systém umožňuje pořídit snímek rovnice pomocí fotoaparátu mobilního telefonu a vybrat umístění rovnice ve fotografii. Poté již sám provede kroky potřebné k rozpoznání textu z obrazu a vyčíslení získané rovnice a následně zobrazí uživateli vyčíslenou formu zadané chemické rovnice. V rámci řešení bylo navrženo grafické uživatelské rozhraní splňující požadavky uživatelů na funkcionalitu, přehlednost a jednoduchou ovladatelnost aplikace.

Abstract

The target of this thesis is to design and implement Android application, which provides automated chemical equation balancing tool. System enables taking a photo of reaction using the smartphone camera and choosing reaction position in the picture. Then the system accomplish operations necessary to optical character recognition and chemical equation balancing. Result of these operations is presented to the user. Within the implementation, graphical user interface meeting users requirements on functionality and easy controllability was designed.

Klíčová slova

Android, Chemické rovnice, OCR, Uživatelské rozhraní, Tesseract

Keywords

Android, Chemical equations, OCR, User interface, Tesseract

Citace

KRAINA, Martin. *Automatické vyčíslování chemických rovnic z fotografie*. Brno, 2016. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Beran Vítězslav.

Automatické vyčíslování chemických rovnic z fotografie

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Vítězslava Berana, Ph.D. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Martin Kraina
17. května 2016

Poděkování

Tímto bych chtěl poděkovat vedoucímu práce, panu Ing. Vítězslavu Beranovi, Ph.D., za cenné rady, vstřícný přístup a vedení této práce.

Obsah

Kapitola 1

Úvod

Tato práce se si klade za cíl poskytnout jednoduchý nástroj pro automatizované vyčíslování chemických rovnic. Chemické rovnice symbolicky znázorňují chemické reakce. Udávají informace o reaktantech (látky do reakce vstupující), produktech (výsledné látky reakce) a poměru v jakém spolu látky reagují. Tento poměr se ovšem dá vyčíst pouze z řádně vyčíslené rovnice. Vyčíslení rovnic může zabrat spoustu času a tudíž je výhodné mít nástroj pro zpracování tohoto problému.

Nástrojů na vyčíslování rovnic je k dispozici několik, většinou se jedná o počítačové programy, nebo online řešení. Věřím, že v dnešní době, kdy má téměř každý k dispozici chytrý telefon, bude daleko vhodnější a pro potenciální uživatele přínosnější, budou-li mít možnost řešit tuto problematiku pomocí aplikace pro chytrý telefon, která ovšem bude schopna fungovat nezávisle na internetovém připojení, jelikož i dnes se stále můžeme dostat do situace, kdy jednoduše nebude internet k dispozici.

Chemické rovnice se v již existujících aplikacích, řešících obdobnou problematiku většinou zadávají pomocí klávesnici, což není především v případě delších rovnic příliš vhodné a tento proces může být v některých případech dokonce časově náročnější, než samotné manuální vyčíslení. Z tohoto důvodu je vhodné umožnit uživateli vyfocení zadané rovnice a nechat získání rovnice k vyčíslení na samotné aplikaci.

Na základě dostupných statistik věnujících se tržnímu podílu jednotlivých operačních systémů použitých v chytrých mobilních telefonech lze dojít k závěru, že pokud chceme oslovit co největší počet uživatelů, je vhodné zacílit implementaci aplikace na operační systém Android. Podle statistik z roku 2014 totiž byl počet prodaných telefonů či s tímto operačním systémem 5x vyšší než počet konkurenčních zařízení se systémem iOS[6]. Aplikace nejprve umožní uživateli vybrat formu zadání chemické rovnice určené k vyčíslení a to buď pomocí klávesnice, nebo zadáním fotografie, ze které aplikace rovnici vyčte. Tuto fotografii aplikace umožní vyfotit či vybrat již existující snímek z galerie. Následně provede vyčíslení rovnice a zobrazí uživateli výsledek spolu s přehledem vypočítaných molárních hmotností jednotlivých látek, které se v reakci vyskytují.

Teorie potřebná k rozboru problematiky, návrhu a implementaci aplikace je vypsána a detailněji vysvětlena v kapitole 2. V rámci této kapitoly je tedy popsána platforma Android, zpracovávání obrazu a rozpoznávání textu v něm. Jelikož je primárně nutné samotnou chemickou reakci dokázat vyčíslit, je v této kapitole věnovaný prostor také metodice vyčíslování chemických rovnic. Vzhledem k existenci nástrojů zabývající se tématem této práce, jsou některé tyto již existující nástroje taktéž představeny a rozebrány.

Pokud podrobíme zadání práce důkladnému rozboru, zjistíme, že je velice důležité na-

vrhnout vhodnou funkcionalitu a grafické uživatelské prostředí aplikace, aby bylo vyhověno všem požadavkům. Také je vhodné celý systém rozdělit do jednotlivých modulů. Tato problematika je tedy zpracována v kapitole 3, Systém vyčíslující chemické rovnice.

Je-li úkol rozebrán a žádaný systém kvalitně navržen, je již možné implementovat veškeré moduly, vytvořit grafické uživatelské prostředí a vše spojit ve výslednou mobilní aplikaci. Podrobnému vysvětlení kritických částí implementace se věnuje kapitola 4, Mobilní aplikace ChemEval. V této kapitole je také popsáno testování implementovaného systému a jeho vyhodnocení.

Kapitola 2

Teorie

Proces vyčíslování chemické rovnice získané pořízením fotografie pomocí mobilního zařízení sestává ze zpracování obrazu a rozpoznání textu. Získaný text je poté potřeba vyhodnotit a vyčíslit pomocí metod k tomu používaných. Použité technologie jsou v této kapitole zdokumentovány. Také je popsán vývoj pro platformu Android a existující aplikace řešící problematiku vyčíslování chemických rovnic.

2.1 Chemické rovnice

Primární funkcí této práce je vyčíslení chemické rovnice. Co je ale chemická rovnice,k čemu je potřebné její vyčíslení a jak se provádí?

Chemická rovnice je grafických vyjádřením zákona o zachování hmotnosti a zákona o zachování náboje, při iontovém zápisu rovnice. Chemická rovnice tak vyjadřuje, že hmota, potažmo atomy, je nezničitelná. Stejný počet atomů, co do reakce vstoupí, musí z ní i vystoupit[5]. Je tvořena vždy levou a pravou stranou oddělenými šipkou. Na straně levé jsou reaktanty, tedy látky do reakce vstupující, na straně pravé pak látky vzniklé reakcí - produkty[8].

2.1.1 Důvody vyčíslování chemických rovnic

Vyčíslování chemických reakcí se provádí za účelem zjištění poměru látkových množství reagujících látek. Jejich pomocí jsme schopni následně zjistit například kolik potřebujeme látky jedné, aby vznikla požadovaná hmotnost látky druhé, nebo naopak kolik látky nám vznikne při reakci daného množství látky. Například pro chemickou rovnici $xA + yB \rightarrow zC$ vyjádříme poměry látkových množství takto: $x : y : z = n(A) : n(B) : n(C)$. $n(A) = \frac{m(A)}{x \cdot M(A)}$, kde *m(A)* je hmotnost látky A, *M(A)* je molární hmotnost téže látky a x je počet molů látky.

2.1.2 Způsoby vyčíslování chemických rovnic

Způsobů vyčíslování chemických rovnic je několik a liší se jak vhodností použití u daného typu rovnic, tak svou složitostí a časovou náročností.

Metoda pokus-omyl za metodu v pravém slova smyslu označit nelze, ale je poměrně rychlá u triviálních rovnic, u rovnic složitějších značně nespolehlivá

Řešení úvahou má svá pravidla, může být velmi rychlá i u složitějších rovnic, předpokládá solidní matematický aparát a zkušenosti

Metoda rovnosti vyměňovaných elektronů ve většině případů spolehlivé a dostatečně rychlé řešení, vyžaduje schopnost určení oxidačních čísel

Řešení pomocí soustavy lineárních rovnicje metoda nejvíce vhodná k zautomatizování, byla proto vybrána k implementaci a níže si tuto metodu přiblížíme.

Pro chemickou rovnici $KNO_3 + C \rightarrow K_2CO_3 + CO + N_2$ získáme matici na základě počtu jednotlivých prvků ve sloučeninách reakce.

$$\left(\begin{array}{c|ccccc} & KNO_3 & C & K_2CO_3 & CO & N_2 \\ K & 1 & 0 & -2 & 0 & 0 \\ N & 1 & 0 & 0 & 0 & -2 \\ O & 3 & 0 & -3 & -1 & 0 \\ C & 0 & 1 & -1 & -1 & 0 \end{array}\right)$$

Získanou matici dále musíme upravit do takzvaného schodovitého tvaru, kde jsou pod diagonálou samé nuly. To se provádí postupnou záměnou řádků matice a vzájemným odčítáním mezi řádky.

$$\left(\begin{array}{c|ccccc} & KNO_3 & C & K_2CO_3 & CO & N_2 \\ O & 3 & 0 & -3 & -1 & 0 \\ N & 0 & 0 & 1 & \frac{1}{3} & -2 \\ K & 0 & 0 & -1 & \frac{1}{3} & 0 \\ C & 0 & 1 & -1 & -1 & 0 \end{array}\right)$$

$$\left(\begin{array}{c|ccccc} & KNO_3 & C & K_2CO_3 & CO & N_2 \\ O & 3 & 0 & -3 & -1 & 0 \\ C & 0 & 1 & -1 & -1 & 0 \\ K & 0 & 0 & -1 & \frac{1}{3} & 0 \\ N & 0 & 0 & 1 & \frac{1}{3} & -2 \end{array}\right)$$

$$\left(\begin{array}{c|ccccc} & KNO_3 & C & K_2CO_3 & CO & N_2 \\ O & 3 & 0 & -3 & -1 & 0 \\ C & 0 & 1 & -1 & -1 & 0 \\ K & 0 & 0 & -1 & \frac{1}{3} & 0 \\ N & 0 & 0 & 0 & \frac{2}{3} & -2 \end{array}\right)$$

Nyní již můžeme sestavit soustavu rovnic

$$\begin{aligned} x_4 &= 3x_5 \\ x_4 &= 3x_3 \\ x_2 - x_3 &= x_4 \\ 3x_1 - 3x_3 &= x_4 \end{aligned}$$

Tuto soustavu rovnic vyřešíme: $x_4 = 3x_5, x_3 = x_5, x_2 = 4x_5, x_1 = 2x_5$, jak jde vidět, x_5 bude nabývat nejmenší hodnotu, proto $x_5 = 1$ a získáme $x_4 = 3, x_3 = 1, x_2 = 4, x_1 = 2$. Nyní již jen přiřadíme vypočítané hodnoty jednotlivým látkám a rovnice je vyčíslena.

$$2KNO_3 + 4C \rightarrow K_2CO_3 + 3CO + N_2$$

2.1.3 Výpočet hmotností

Výše uvedeným způsobem se tedy dobereme k vyčíslené rovnici. Pro některé úlohy, jako například již zmíněný výpočet hmotností, je ovšem zapotřebí zjistit také molární hmotnosti látek. Tato molární hmotnost látky je fyzikální veličina udávající váhu jednoho molu látky. Právě koeficienty získané vyčíslením rovnice značí počty molů látek reakce. Molární hmotnost atomu je rovna relativní atomové hmotnosti prvku a lze ji vyčíst z periodické tabulky prvků(např. M(H)=1.008). Molární hmotnost prvku je v rámci sloučeniny násobena počtem tohoto prvku a sčítána s molární hmotností dalších prvků obsažených ve sloučenině.

$M_{KNO_3} = M(K) + M(N) + M(O_3) = 39,098 + 14.006 + 3 \cdot 16 = 101.104$

$M_{K_2CO_3} = M(K) + M(N) + M(O_3) = 2 \cdot 39,098 + 12.009 + 3 \cdot 16 = 138.205$

Získali-li jsme již jednotlivé molární hmotnosti, pomocí jednoduché trojčlenky již můžeme získat potřebné hmotnosti. Například z výše vyčíslené rovnice chceme zjistit, kolik gramů dusičnanu draselného (KNO_3) potřebujeme k vytvoření 10 gramů uhličitanu draselného (K_2CO_3).

$$\begin{array}{ccccccccc} 2KNO_3 & + & 4C & \rightarrow & K_2CO_3 & + & 3CO & + & N_2 \\ x & & \ldots\ldots\ldots & & 10g & & & & \\ 2 \cdot 101.104 & & \ldots\ldots\ldots & & 1 \cdot 138.205 & & & & \end{array}$$

Získáváme tedy rovnici

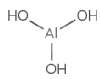
$$x = \frac{2 \cdot 101.104 \cdot 10}{1 \cdot 138.205}$$

$$x = 14.63g$$

K vytvoření 10 g uhličitanu draselného je tedy zapotřebí 14.63 g dusičnanu draselného.

2.1.4 Speciální způsoby zápisu reagujících látek

V reakcích se kromě jednoduchých sloučenin mohou objevit i látky s mírně složitějším zápisem, jako je například hydroxid hlinitý, $Al(OH)_3$. Tento zápis značí, že skupina uvedena v závorce je v látce zastoupena právě třikrát, jak lze vidět na obrázku níže. Celkem je tedy v této látce 3 atomy vodíku, 3 atomy kyslíku a 1 atom hliníku. Obdobně se tyto počty vyjadřují například ve vzorci hexachloridoplatičitanu amonného, $(NH4)_2[PtCl_6]$, kde je skupina NH_4 zastoupena dvakrát a skupina $PtCl_6$ jednou, na hranatou závorku tedy nemusíme brát zřetel a výsledné počty jsou 6 atomů chlóru, jeden atom platiny, 2 atomy dusíku a 8 atomů vodíku.



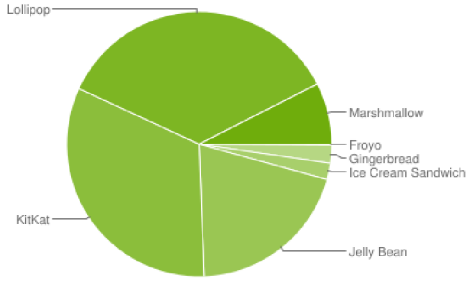
Obrázek 2.1: Hydroxid hlinitý [1]

[1] Zdroj: http://chem.sis.nlm.nih.gov/chemidplus/rn/14762-49-3

2.2 Vývoj pro OS Android

Android je operační systém založený na linuxovém jádře vyvíjený firmou Google. Je navržen primárně pro zařízení s dotykovou obrazovkou, jako jsou mobilní telefony či tablety. Uživatelské rozhraní je navrženo k ovládání dotykem pomocí různých gest určených k přiblížení obrazovky, přesunu objektů, výběru ovládacích prvků nebo zadávání vstupních dat pomocí softwarové klávesnice. Kromě výše zmíněných typů zařízení je také k dispozici Android TV pro televize, Android Auto pro automobily a Android Wear pro chytré hodinky. Každá z těchto platforem má specializované uživatelské rozhraní[1].

Přestože většina zařízení se systémem Android obsahuje také proprietární software, z nichž některý je potřebný například k přístupu ke Google službám, samotný operační systém je vydáván pod open source licencí. Také díky tomu existuje velká komunita vývojářů a nadšenců, která zdrojový kód Androidu využívá pro své vlastní projekty, které přináší nové možnosti pokročilým uživatelům nebo možnost instalace systému na zařízení prodávaná s jiným operačním systémem[7]. Absence centralizovaného systému aktualizací OS vede k velkému počtu stále používaných verzí, z nichž některé mohou být velice zastaralé[**?**].



Obrázek 2.2: Používané verze OS Android[2]

2.2.1 Struktura aplikací

Android aplikace se skládají z několika komponent. Základní je **Aktivita** (*Activity*), jež představuje obrazovku zobrazenou uživateli obsahující jednotlivé prvky grafického uživatelského rozhraní. Aktivita odpovídá jedné obrazovce. Poskytuje uživateli grafické uživatelské rozhraní určené k obsluze aplikace. Většinou jsou aplikace tvořeny vícero aktivitami, mezi kterými je uživatel schopen přepínat a přitom si aktivity mohou předávat informace formou **itentů**. Vzhledem k nutným operacím, které musí být provedeny při zahájení aktivity, je tento proces poměrně náročná záležitost. Pro každou aktivitu se musí vytvořit nový proces, alokovat paměť pro objekty uživatelských rozhraní, které se rozloží do layoutu obrazovky a na připravenou obrazovku vyvolat zobrazení. Aby nedocházelo ke zbytečnému plýtvání výpočetních prostředků např. při vzniku, zániku a opětovného vzniku aktivity – což se jednoduše může stát při stisku tlačítka zpět na zařízení – je zde *Activity Manager*, který zodpovídá za správu všech stavů životních cyklů aktivit. Activity Manager si ve formě zásobníku, na jehož vrcholu je aktuálně zobrazovaná aktivita, uchovává přehled o všech spuštěných aktivitách. Životní cyklus aktivity se může nacházet v těchto stavech:

- **Activity starts** – Počátek, kdy je aktivita inicializována.

[2]Zdroj:http://developer.android.com/about/dashboards/index.html

- **Activity is running** – Aktivita je zobrazena na displeji a může reagovat na uživatelovy podněty. V jediném okamžiku může být právě jedna aktivita v tomto stavu.
- **Process is killed** – Activity Manager byl donucen ukončit aktivitu, která není viditelná z důvodu nedostatku paměti. Další možnost není tak obvyklá – aktivita je viditelné, ale uživatel s ní nemůže navázat interakci (nastává například při dialogových hláškách).
- **Activity is shut down** – Activity Managerem ukončil activity a ta již nevyužívá žádnou paměť.

Služby (*Service*) jsou procesy běžící na pozadí, neobsahující vlastní grafické rozhraní. Většinou se používá k vykonávání dlouho trvajících úkolů nebo k přístupu k vzdáleným zdrojům, kde není známá doba odezvy (jako je připojení k serveru). Služby můžeme spustit dvěma způsoby a to pomocí metody *startService*. Potom se služba může ukončit sama nebo ji může ukončit jiná komponenta. Další způsob spuštění je pomocí metody *bindService*, kterou vyvolá jiná komponenta, tzv. klient, v tomto případě službu může ukončit pouze klient, který ji spustil. V jednom okamžiku může být k službě navázáno pomocí metody bindService i více komponent, potom je služba ukončena po odpojení všech klientů. Služba se může nacházet ve třech stavech:

- **Component** calls – Inicializace service pouhým zavoláním nebo navázáním komponenty na service.
- **Service is running** – Service vykonává na pozadí svou funkci.
- **Service is shut down** – Service byl ukončen sám nebo komponentou, záleží na formě spuštění service.

Poskytovatel obsahu (*Content provider*) umožňuje sdílení dat jak mezi různými aplikacemi, tak mezi jednotlivými aktivitami jedné aplikace. Aplikace může uchovávat data v souborech, SQL databázích nebo na webu, a přesto budou mít k těmto datům přístup – pokud je to povoleno – jiné aplikace. Content provider má relativně jednoduché rozhraní se standardními metodami (*insert, update, delete a query*), které mají stejnou funkci jako klasické databázové metody. Oddělení dat od uživatelského rozhraní nabízí možnost nahrazení výchozích aplikací novými. Například může jakákoliv aplikace využít uložených uživatelských kontaktů a nahradit tak defaultní aplikaci pro jejich zobrazování. Kromě content provideru mohou komponenty mezi sebou kooperovat pomocí zpráv, tzv. *intentů*.

Přijímač přenosů (*Broadcast receiver*) je komponenta sloužící k práci s oznámeními oznámení. Podle určení na ně reaguje, například výpisem na stavový řádek nebo spuštěním jiné komponenty. Aplikace mohou využívat broadcasty systémové nebo vytvářet své vlastní. Podobně jako service ani broadcast receiver nemá uživatelské rozhraní. Příklad použití broadcast recieveru může být reakce na oznámení o nízkém stavu baterie, o zachycení fotografie, doručení SMS zprávy nebo stažení dat.

Všechny tyto komponenty musí být definovány v souboru **AndroidManifest.xml**, uloženém v kořenovém adresáři projektu. Tento soubor také obsahuje seznam oprávnění potřebných k chodu aplikace[2].

2.2.2 Vývojové prostředky

Jako vývojové prostředí aplikací pro platformu Android lze v současné době využít Android Studio, oficiální IDE pro vývoj Android aplikací, vyvíjené společností Google, která zašti-

ťuje také vývoj Androidu samotného. Android Studio je postavené IntelliJ IDEA, k jehož kvalitnímu editoru zdrojových kódů a vývojovým nástrojům byla přidána spousta vlastností, které usnadňují vývoj aplikací a zvyšují produktivitu práce:

- Flexibilní sestavovací systém založený na Gradle.
- Šablony představující využití běžných vlastností aplikací.
- Editor grafického rozhraní s podporou změn rozvržení pomocí *drag and drop*.
- Nástroje k ošetření chyb spojených s výkonem, použitelností, kompatibilitou jednotlivých verzí a jinými. Lint tools to catch performance, usability, version compatibility, and other problems
- Vestavěnou podporou pro *Google Cloud Platform*, jednoduchou integraci *Google Cloud Messaging* a *App Engine*.

Android Studio je volně dostupné pod licencí Apache License 2.0 a klade si za cíl nahradit nejpoužívanější vývojové prostředí Eclipse s přidanými ADT (Android Deevelopment Tools). První stabilní verze byla vydána v prosinci 2014 a nyní se nachází ve verzi 2.1 (k květnu 2016)[4].

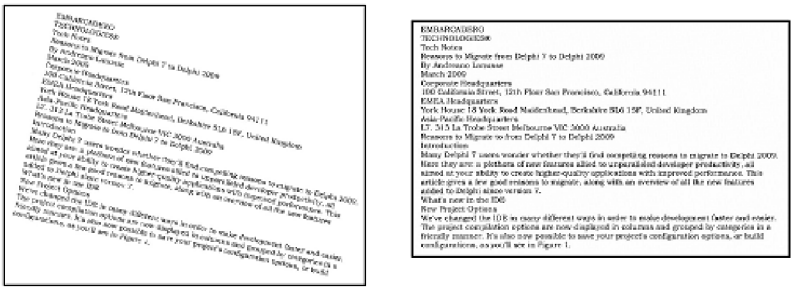
2.3 OCR

Optickým rozpoznáváním znaků (Optical Character Recognition) se rozumí proces umožňující digitalizaci tištěných textů získaných pomocí skeneru či fotoaparátu. Takto získaný text je poté možno dále zpracovávat jako standardní počítačový text. Rozpoznání znaků není bezchybné a především u nekvalitních podkladů určených k digitalizaci je zapotřebí důkladná korektura, protože OCR program nemusí rozeznat všechna písmena správně. Zpracování textu je možné využít u jakkoliv tištěného textu, včetně textů vytvořených laserovými, inkoustovými, termosublimačními a jehličkovými tiskárnami, psacími stroji či pomocí knihtisku. Avšak je-li předloha nekvalitní, např. slabě vytištěná jehličkovou tiskárnou nebo obsahuje mnoho dohromady slitých písmen, je z časového hlediska výhodnější přepis textu [10]. Rané období OCR je spojováno s technologiemi jako jsou telefgrafie a tvorba zvukových zařízení pro osoby s vadou zraku, Emanuel Goldberg již v roce 1914 vyvinul přístroj, který dokázal číst znak a převádět je ve standardní telegrafní kód[9].

2.3.1 Předzpracování obrazu

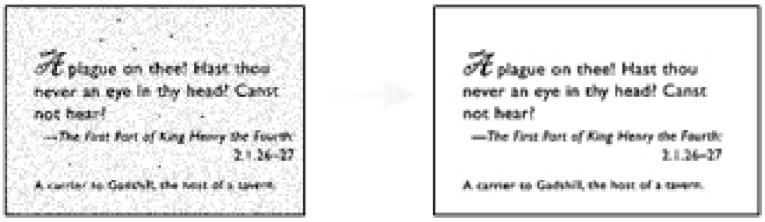
Správnost rozpoznávání znaků je silně ovlivněna kvalitou zpracovávaných podkladů. Proto je nutné obraz nejdříve upravit tak, aby byl co nejlépe strojově čitelný. Tento proces je rozdělen do několika dílčích úkonů.

- **Zarovnání** - je nutné provést, jelikož naskenovaný text nemusí být zarovnaný, je tedy potřeba provést mírné pootočení obrazu, tak aby byl řádně vodorovně i vertikálně srovnán. Obrázek 2.2 vlevo představuje obraz na vstupu, vpravo je pak výsledek tohoto procesu.



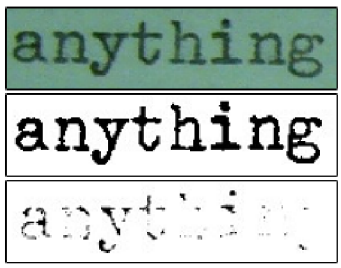
Obrázek 2.3: Zarovnání [3]

- **Vyhlazení** - odstranění šumu a vyhlazení hran. Vyhlazení lze rozdělit na dvě části- vyplňování a zužování. Vyplňování nám zacelí malé dírky v písmenech, tak aby písmeno bylo tvořeno souvislou plochou barvy. Zúžení pak ztenčí rozpoznávaný znak.



Obrázek 2.4: Vyhlazení [4]

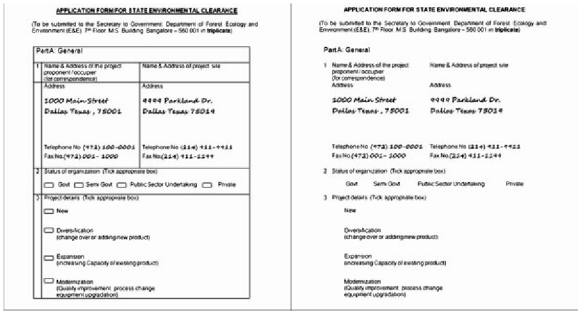
- **Prahovaní (Binarizace)** - převod barevného obrazu, nebo obrazu ve stupních šedi na čistě černobílý, tedy pouze dvoubitová data- černá a bílá (0 a 1), za účelem oddělení textu od pozadí. Obrázek 2.4 znázorňuje obraz před provedením prahování (nahoře), správně provedené prahování (uprostřed) a chybně provedené prahování (dole).



Obrázek 2.5: Prahování [5]

- **Oddělení textu** - odstranění částí obrazu, jež neobsahují text, tedy obrázky, grafy, čáry a podobné.

[3] Zdroj: http://scanningireland.com/editing-files-services/
[4] Zdroj: https://www.avision.com/motion.asp?menuid=10114
[5] Zdroj: https://www.nicomsoft.com/optical-character-recognition-ocr-how-it-works/



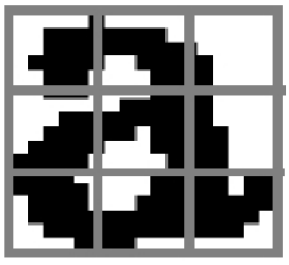
Obrázek 2.6: Oddělení textu [6]

- **Analýza rozvržení** - identifikuje sloupce, odstavce, nadpisy apod. jako odlišné bloky textu.
- **Segmentace znaků** - musí být provedeno rozdělení dvou nežádaně spojených znaků a složení znaku roztříštěných znaků. Tedy náprava chyb způsobených některým z předcházejících kroků.
- **Normalizace znaků** je posledním procesem přípravy zdroje k počítačovému zpracování. Upravuje sklon, rotaci a jednotkovou velikost znaků, tak aby byly tyto vlastnosti u všech znaků shodné.

2.3.2 Rozpoznávání znaků

OCR využívá dvou základních metod k rozpoznávání znaků.

- **Rozdělení do pásem** Políčko s lokalizovaným znakem je rozděleno na několik oblastí a zkoumá se histogram tmavých míst v jednotlivých oblastech znaku, jak je vidět na obrázku. Histogramy se pak porovnávají s rysy jednotlivých znaků, které vzejdou z tzv. trénovacích dat.
- **Průsečíky** Tato metoda je založena na počtu průsečíků předem zvolených vektorů v políčku se znakem. Metoda rozpoznávající na základě specifických rysů je nazývána strukturální analýzou, kdy jsou jednotlivé znaky popisovány geometrickou a topologickou strukturou znaků[10].



Obrázek 2.7: Rozdělení do pásem[7]

Obrázek 2.8: Průsečíky[7]

[6] Zdroj: http://www.xplorimaging.com/DocumentCleanUp.html
[7] Zdroj: http://geo3.fsv.cvut.cz/vyuka/kapr/SP/2008_2009/vymetalek_viktora/index.html

2.3.3 Tesseract

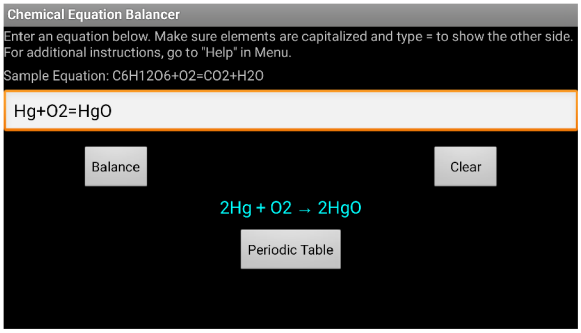
Tesseract je engine původně vyvíjený jako proprietární software firmou Hewlett Packard mezi lety 1985 a 1994 napsaný v jazyce *C/C++*. V roce 2005 byl engine firmou uvolněn jako open-source projekt a od roku 2006 je jeho vývoj sponzorován firmou Google. Již v roce 1995 byl mezi třemi nejlepšími OCR enginy podle přesnosti rozpoznávání znaků. Vzhledem k implementaci v jazyce *C/C++* je engine multiplatformní, k dispozici je tedy pro Windows, Linux i Mac OS X, vzhledem k omezeným financím je ovšem řádně testován jen pro Windows a linuxovou distribuci Ubuntu. Přestože k vývoji aplikace postačí schopnost Tesseractu rozpoznat anglické texty, je vhodné zmínit, že podporuje přes sto dalších jazyků včetně francouzštiny, němčiny, španělštiny, jazyků psaných zprava doleva a v neposlední řadě češtiny.

Výsledek procesu rozpoznávání je závislý na kvalitě vstupních obrazů, ty proto musí byt předzpracovány. Obraz musí být zvětšen tak, aby písmo mělo alespoň 20 pixelů vysoké. Samozřejmostí je úprava rotace a sklonu písma. Různá úroveň jasu obrazu musí být odfiltrována, aby nedošlo ke ztrátě dat binarizací Tesseractu. Dále je vhodné odstranit tmavé ohraničení, jinak může být zaměněno za text[3].

2.4 Existující podobná řešení

I přes celkem úzký okruh potencilních uživatelů je pro platformu Android několik aplikací pro vyčíslování chemických rovnic, avšak žádná neposkytuje možnost zpracování rovnice získané z pořízeného snímku. Následující aplikace byly vybrány podle počtu hodnocení.

- **Chemical Equation Balancer** - Jednoduchá aplikace očekávající zadání celé rovnice, kterou pouze vyčíslí a žádné další informace nevypisuje. Rovnice musí mít správně zapsaná malá a velká písmena, jinak vyčíslení nefunguje. Na několika ze zkušebních rovnic aplikace selhala a vyčíslila rovnici špatně[7].

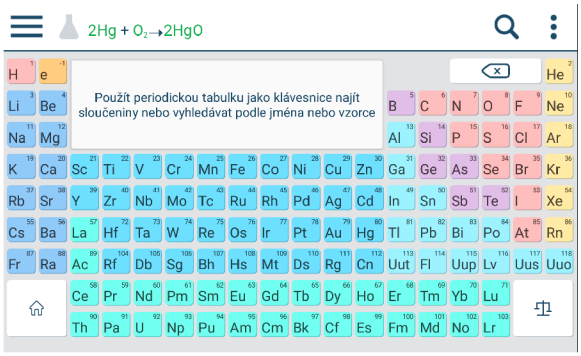


Obrázek 2.9: Chemical Equation Balancer

- **Chemik - super chemie nástroj** - Látky jsou zadávány výběrem z periodické tabulky prvků a následné volby počtu atomů prvku. Po výběru všech reaktantů jsou automaticky doplněny produkty z databáze a rovnice je vyčíslena. Aplikace navíc obsahuje také názvosloví sloučenin. Právě způsob zadávání rovnice je velmi zajímavý, jelikož na jednu stranu usnadňuje zápis sloučenin, kdy si uživatel vybere z nabídnutých a tím pádem nemusí složitě přecházet mezi jednotlivými bloky klávesnice, na stranu

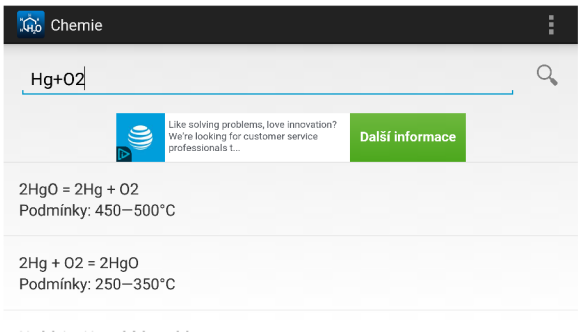
[7] Zdroj: https://play.google.com/store/apps/details?id=com.upas.eqbalancer

druhou je orientace v periodické tabulce zpočátku velice složitá a může zbytečně prodloužit dobu zapisování[8].



Obrázek 2.10: Chemik

- **Chemistry** - Aplikaci jsou formou textu zadány reagující látky, na základě nichž aplikace vypíše všechny chemické reakce, ve kterých se tyto látky společně vyskytují jak na straně reaktantů, tak na straně produktů. Výpis je vcelku nepřehledný, uživatel se musí naučit v něm vyhledat žádané informace[9].



Obrázek 2.11: Chemistry

[8]Zdroj: https://play.google.com/store/apps/details?id=com.bk.advance.chemik
[9]Zdroj:https://play.google.com/store/apps/details?id=com.chemistry

Kapitola 3

Systém vyčíslující chemické rovnice z obrazu

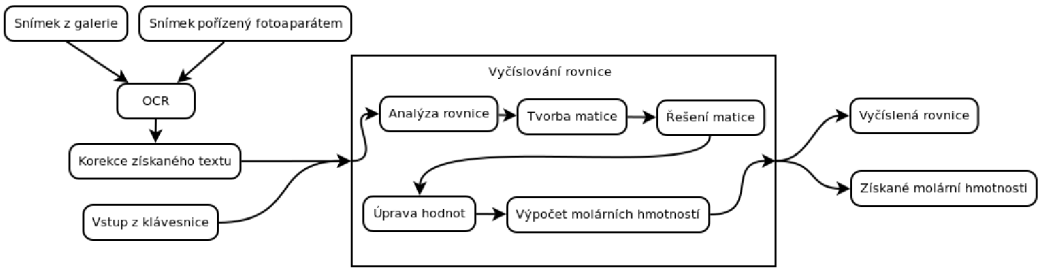
V této kapitole je čtenář detailněji seznámen s cíli práce a návrhem řešení aplikace. Je zde nastíněn postup práce a rozdělení aplikace do jednotlivých modulů, které dokáží fungovat bez ohledu na funkcionalitu ostatních modulů a jsou spojeny teprve finální aplikací do jednoho funkčního bloku. Těmito moduly je soubor algoritmů určených k postupnému vyčíslení aplikace, získání snímku reakce, jeho následná úprava a extrakce znaků a grafické uživatelské rozhraní umožňující zobrazující uživateli ovládací prvky, pomocí kterých se lze v aplikaci orientovat, určovat způsob využití aplikace. Tyto uživatelovi možnosti použití aplikace jsou v této kapitole taktéž popsány.

3.1 Rozbor a analýza

Zadáním této práce je tvorba aplikace pro mobilní zařízení užívající platformu Android, určené k automatizovanému vyčíslování chemických rovnic, již je uživatel schopen ovládat pomocí jednoduchého grafického rozhraní. Od uživatele je očekáván vstup ve formě nevyčísleného zápisu reakce, tedy reakčního schéma, vloženého buď přímo klávesnicí, nebo získaného ze snímku uživatelem vybraným z galerie či vyfoceným pomocí vestavěného fotoaparátu. Podle formy vstupních dat je uživatel vyzván k označení oblasti schématu na snímku a po rozpoznání textu případně umožněno editovat toto schéma za účelem korekce špatně rozpoznaných znaků. Poté je již spuštěn samotný vyčíslovací algoritmus, který, skončí-li bez chyby, zobrazí uživateli vyčíslenou rovnici. Kromě vyčíslené rovnice poskytne uživateli také vypočítané molární hmotnosti jednotlivých sloučenin chemické reakce.

3.2 Návrh systému

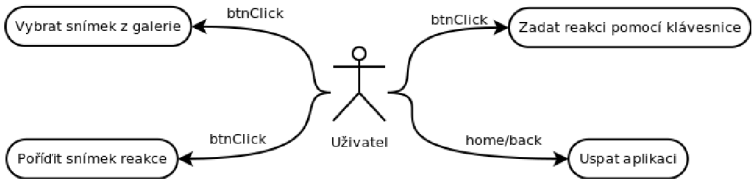
Jak již bylo řečeno, aplikace bude rozdělena do několika modulů, z nichž hlavní je modul pro samotné vyčíslení rovnice. Tento modul obsahuje především funkce na získání jednotlivých prvků a sloučenin v rovnici za účelem sestavení matice. Sestavená matice se bude počítat pomocí Gaussovy eliminační metody. Výsledky získané touto metodou je zapotřebí ještě upravit na celá čísla bez společného dělitele. Poté se získané konstanty dosadí k příslušným sloučeninám v reakci a tímto je reakce vyčíslena.



Obrázek 3.1: Schéma systému

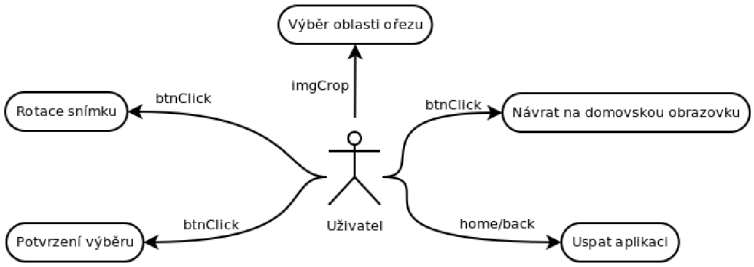
3.3 Popis případů užití

Aby bylo možno navrhnout co nejpřívětivější grafické uživatelské rozhraní, je zapotřebí zjistit, jaké úkony a v jakém sledu očekává uživatel, že bude muset udělat, aby získal požadovaný výsledek, tedy vyčíslenou rovnici. Toto je dokumentováno pomocí diagramů případů užití. Na obrázku 3.2 jsou znázorněny akce, které by měly být umožněny uživateli při spuštění aplikace. Zde uživatel očekává možnost výběru způsobu zadání reakce a dále samozřejmě možnost opuštění aplikace.



Obrázek 3.2: Usecase diagram po spuštění aplikace

Po zvolení formy vstupu je další postup zřejmý, zadat a potvrdit rovnici. V případě vstupu z klávesnice, již dále není nutný žádný mezikrok, ale pokud je reakce znázorněna na snímku, bude zapotřebí vybrat oblast snímku, kterou rovnice zaujímá. Na takovéto obrazovce by mělo být uživateli umožněno, kromě samotného vyznačení oblasti a potvrzení, nebo zrušení akce, také editovat snímek, který může být špatně pootočen.



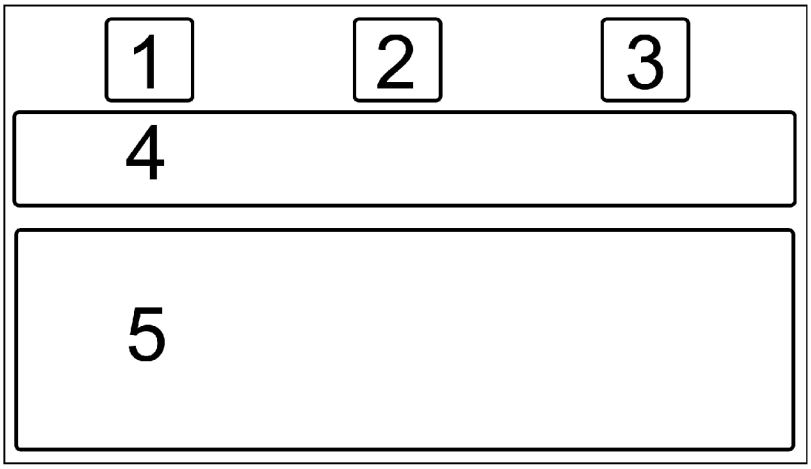
Obrázek 3.3: Usecase diagram obrazovky ořezu

Legenda use-case diagramů:

- **btnClick** - stisk tlačítka GUI
- **home/back** - stisk/zmáčknutí hardwarového tlačítka "domů"či "zpět"
- **imgCrop** - tažení mřížkou prvku pro výběr ořezové oblasti

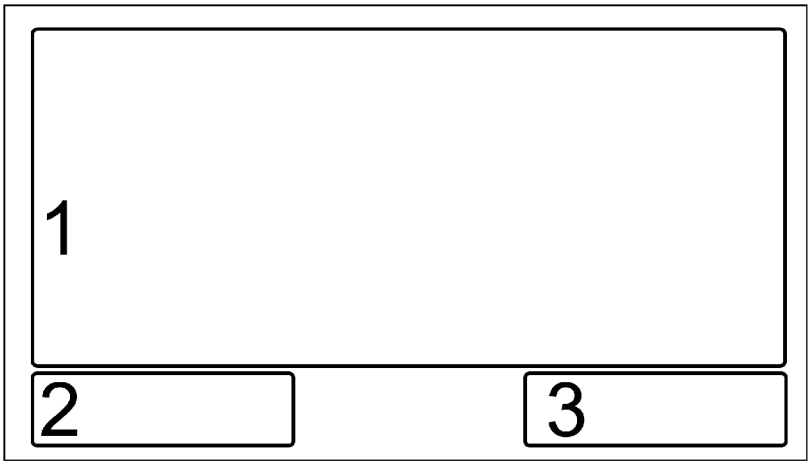
3.4 Návrh grafického rozhraní

Jelikož uživatel vyžaduje co nejjednodušší aplikaci se snadnou orientací, umožňující co nejrychlejší seznámení s poskytovanou funkcionalitou, bylo navrženo uživatelské rozhraní, které neupřednostňuje jeden způsob zadávání rovnice před druhým. Toto je provedeno pomocí rozcestníku ve formě domovské obrazovky (obrázek 3.2), z něhož uživatel ihned při prvním použití může rychle vyčíst, že jsou mu poskytovány tři (oblasti 1,2,3 na obrázku 3.4) rozdílné možnosti jak může vložit rovnici, kterou potřebuje vyčíslit. Tyto možnosti jsou rozlišeny pomocí piktogramů co nejvěrněji označujících danou činnost. Rozhraní fotoaparátu je bezpředmětné popisovat, jelikož bude využíváno některé již nainstalované aplikace určené k pořizování fotografií. Totéž platí pro výběr již existujícího snímku z galerie.



Obrázek 3.4: Návrh rozhraní aplikace

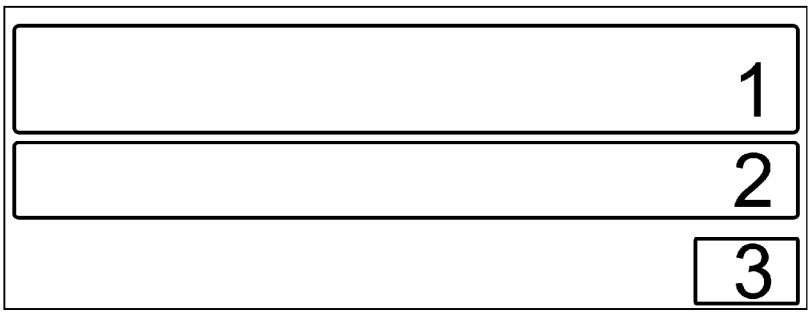
Obrazovka určená k výřezu reakčního schématu z fotografie je znázorněna na obrázku 3.5 a zahrnuje grafický prvek pro ohraničení oblasti obrazu (číslo 1), tlačítka pro rotaci o snímku 90 ° po i proti směru hodinových ručiček (2), potvrzení ořezu a zaslání výřezu dále k získání textu a také prvek pro zrušení činnosti (3).



Obrázek 3.5: Návrh rozhraní aplikace

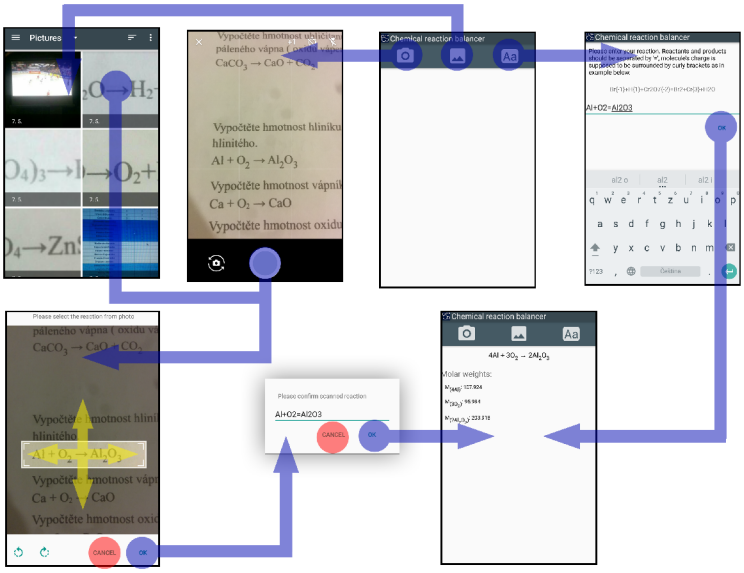
Obrázek 3.4 zobrazuje obrazovku pro zadávání reakčního schématu pomocí klávesnice,

jež obsahuje nápovědu týkající se očekávaného formátu reakce (1), pole pro zadání reakce (2) a tlačítko pro potvrzení reakce(3).Obdobně je řešen dialog pro potvrzení schématu získaného ze snímku, avšak již neobsahuje nápovědu.



Obrázek 3.6: Návrh rozhraní aplikace

Na obrázku 3.8 jsou znázorněné jednotlivé přechody uživatele mezi obrazovkami pomocí ovládacích prvků. Modře je vyznačeno jednoduché klepnutí na ovládací prvek aplikace, žlutě tažení ovládacím prvkem a červeně stisk ovládacích prvků rušících operaci s následných návratem na domovskou obrazovku. Vzhled a způsob ovládání fotoaparátu a galerie se může lišit dle použité systémové aplikace.



Obrázek 3.7: Znázornění přechodů mezi obrazovkami

3.5 Návrh vyčíslení rovnice rovnice

Modul pro vyčíslování chemické rovnice je zapotřebí také rozdělit do dílčích bloků, které mezi sebou budou postupně komunikovat. Těmito bloky jsou analýza rovnice, zisk matice potřebné k vyčíslení, řešení matice, zjištění váhy získaného řešení a výpočet molárních hmotností matice. Aplikace tohoto návrhu je poté dále vysvětleno v kapitole 4.

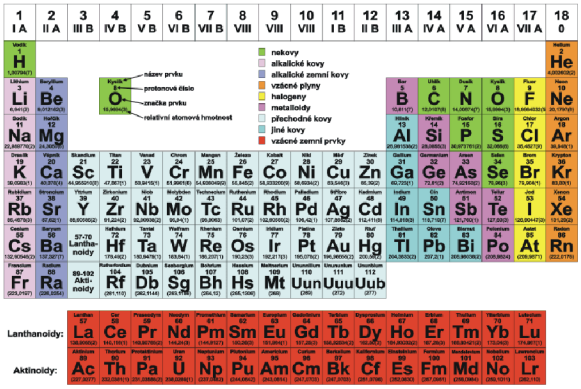
Jako první je zapotřebí ověřit **platnost rovnice**. Jestliže totiž schéma nesplňuje několik požadavků, není možné jej vyčíslit.

- Základním požadavkem na reakční schéma je možnost jednoznačně oddělit levou a pravou stranu rovnice, tedy reaktanty od produktů. Toto rozdělení tvoří znak „=".
- V rovnici dále mohou být zastoupeny pouze určité kombinace znaků. Tyto kombinace tvořící prvky jsou určeny periodickou soustavou prvků. Jednotlivé prvky mohou být ve sloučenině zapsány ihned za sebou, nebo mohou být spojeny číslicí. Sloučeniny jsou pak odděleny znakem „+", „=" či „→".
- Jsou-li získány jednotlivé prvky zastoupené v reakci, je potřeba ověřit, zda množina prvků na straně levé je rovna množině prvků na straně pravé.

Jestliže jsou splněny tyto požadavky, je možno postoupit k vyčíslení rovnice.

Algoritmus určený k **získání prvků zastoupených v reakci** prochází řetězec, v němž jsou uloženy reaktanty či produkty a porovnává jednotlivé podřetězce s periodickou tabulkou prvků, pokud je tento podřetězec zastoupen v této tabulce a zároveň se ještě v řetězci nevyskytl, je následně zapsán do pole obsahujícího všechny prvky, jež reakce obsahuje. V případě, že dalším znakem není číslice, oddělující znaky nebo další platná kombinace, je reakční schéma vyhodnoceno jako chybné a uživatel je o tomto informován. Toto neplatí, byl-li podřetězec na konci analyzovaného řetězce.

Následně je takto získané pole pro stranu reaktantů porovnáno se získaným polem strany produktů. Pokud se obsažené prvky neshodují, je rovněž schéma vyhodnoceno jako neplatné.



Obrázek 3.8: Periodická soustava prvků

Před samotným vytvořením matice je zapotřebí **rozdělit reakci** na jednotlivé sloučeniny rozdělených znaky „+", „=" či „→". Poté jsou tyto sloučeniny procházeny a porovnávány s množinou prvků. Bude-li prvek obsažen ve sloučenině, zapíše se jeho počet na příslušný index matice, v případě, že bude látka obsahující prvek na straně produktů, bude zapsaný počet vynásoben -1. Pokud prvek v dané sloučenině není zastoupen, je na daný index zapsána nula. Pro každý prvek je takto vytvořen jeden řádek matice, ve sloupcích jsou pak počty prvků v jednotlivých látkách.

Vytvořená matice bude následně vyřešena pomocí Gaussovy eliminační metody, která prochází matici sloupec po sloupci a postupně upravuje hodnoty, čímž přepisuje matici do schodovitého tvaru. Řešení matice je poté získáno pomocí zpětné substituce.

Jelikož je řešením matice pouze jakýsi poměr mezi prvky, který může být vyjádřen i pomocí desetinných čísel, je potřeba tento výsledek upravit na celá čísla. K tomuto účelu je využita **váha řešení**, kterou myšlena konstanta, jejímž vynásobením řešení dostaneme výsledek v požadovaném tvaru.

S takto získaným výsledkem je možno dále pracovat za účelem získání celkové **molární hmotnosti látky** v dané chemické reakci. Pro každý prvek ve sloučenině je potřeba zjistit molární atomovou hmotnost, tu vynásobit počtem prvku v látce a tyto hodnoty následně sečíst. Takto vypočítanou molární hmotnost dále vynásobíme s výsledkem řešení matice a následujících kroků, jež byly zmíněny výše.

3.6 Zpracování obrazu a rozpoznání znaků

Snímek s rovnicí určenou k vyčíslení je pořizován vně této aplikace pomocí systémové aplikace fotoaparátu, nebo jiných aplikací třetích stran. Tento přístup je zvolen z toho důvodu, že tyto aplikace mohou mít pokročilejší funkce zaměřené na kvalitu pořízených fotografií. Výběr fotografie z galerie je také vhodné vyřešit pomocí systémové aplikace. Dále je potřebný nástroj na ořez získané fotografie, k tomu je použita vhodná knihovna[1].

Samotná problematika zpracování obrazu a jeho úpravy za účelem rozpoznání textu je implementována pomocí knihovních funkcí enginu Tesseract. Poté je zapotřebí provést korekci rozpoznaných znaků, aby dávaly dohromady platné, a tudíž vyčíslitelné rovnice. Pro tento účel je vypracován algoritmus, jenž prvotně převede jednoduché neplatné kombinace znaků do akceptovatelné podoby. Následně je použito složitějších záměn na základě množiny očekávatelných sloučenin.

[1] Zdroj: https://github.com/ArthurHub/Android-Image-Cropper

Kapitola 4

Mobilní aplikace ChemEval

Mobilní aplikace je složena z několika modulů řešících jednotlivé podproblémy spojené s vyčíslováním chemických rovnic z obrazu. Rozdělení systému do jednotlivých modulů je nastíněno v předcházející kapitole a je tedy následující - zpracování obrazu a rozpoznání textu, samotné vyčíslení rovnice a grafické uživatelské rozhraní. Implementace modulu je postavena na modulu **tess-two**, který bude popsán dále v tomto textu[1].

Vyčíslování rovnice je provedeno vyřešením matice sestavené podle zastoupení prvků v jednotlivých látkách. Základem funkce pro vyřešení matice je funkce pro **Gausovu eliminační metodu se zpětnou substitucí**[2].

4.1 Vyčíslení rovnice

Modul určený k vyčíslování chemických rovnic je nejdůležitější komponentou aplikace a je implementovaný na základě poznatků zmíněných v kapitole 2. Jak již bylo v této kapitole nastíněno, pro zdárné vyčíslení chemické reakce je zapotřebí znát veškeré prvky v reakci obsažené a jejich přesné počty, tedy provést **rozbor reakce**. Jako první je tedy zapotřebí získat prvky na obou stranách rovnice. Toho je dosaženo zaprvé rozdělením rovnice na levou a pravou stranu a následně funkcí *getElements()*, která je volána pro obě strany. Tato funkce porovnává podřetězce jednotlivých látek s množinou prvků periodické soustavy a postupně rozpoznané prvky zaznamenává. Je-li nalezen neplatný znak, či posloupnost znaků (tedy neexistující zkratka prvku), je rovnice označena jako neplatná, obdobně je postupováno i v případě, vyskytují-li se na obou stranách rovnice jiné prvky, nebo objevuje-li se některý prvek pouze na jedné straně rovnice.

Dalším krokem je upravit reakci tak, aby neobsahovala molekuly zapsané ve tvaru $Al(OH)_3$, tedy odstranit závorky a správně doplnit počty, výsledek poté bude vypadat takto AlO_3H_3. Tato úprava je prováděna funkcí *removeBracket()*, která prochází reakci a v případě, že narazí na závorku, přečte číslo následující po závorce, závorku smaže a číslo zapíše za každý jednotlivý prvek, jenž se vyskytoval v závorce. Byl-li prvek zastoupen v této skupině vícekrát, je jeho počet adekvátně vynásoben.

Z takto upravené rovnice je již možno započít **sestavení matice**. Jednotlivé prvky reakce zapsané pomocí funkce *getElements()* do struktury ArrayList<> jsou ve funkci *getMatrix()* porovnávány postupně s každou reagující látkou, obsahuje-li látka prvek, daný počet

[1] Zdroj:https://github.com/rmtheis/tess-two

[2] Zdroj: http://introcs.cs.princeton.edu/java/95linear/GaussianElimination.java.html

molekul prvku je zapsán na náležité místo matice (sloupec dle pořadí zkoumané látky v rovnici, řádek podle pořadí obsaženého prvku ve struktuře obsahující prvky reakce), nachází-li se látka na pravé straně chemické reakce je počet před zapsáním do matice vynásoben -1.

Jak již bylo uvedeno výše, **výpočet matic** je proveden pomocí Gaussovy eliminační metody kterou ovšem bylo nutné mírně poupravit, algoritmus Gaussovy eliminační metody je implementován funkcí *gaussianElimination()*. Pro $m \times n$ matici A a vektor b platí následující pseudokód. Algoritmus prochází po sloupcích matici a naléza prvek ve sloupci s nejvyšší hodnotou, poté je provedena záměna řádků tak, aby byl tento nalezený řádek na řádku téhož pořadí, jako je procházený sloupec. Od nižších řádků matice je poté tento řádek odečítán pomocí konstanty *alpha* za účelem získání schodovitého tvaru matice. Z niž lze jednoduchou zpětnou substitucí získat řešení matice. Implementovaný algoritmus se od uvedeného v některých částech liší, například v tom, že dojde-li k detekci singulární matice, jsou provedeny prohození sloupců za účelem získat matici, jež má řešení.

Kód 4.1: Pseudokód Gaussovy eliminační metody

```
1  for(i=0;i<m;i++){
2
3   //nalezeni prvku sloupce s~nejvyssi hodnotou
4   max=i;
5   for(j=i+1;j<n;j++){
6    if(A[j][i]>A[max][i])
7     max=j;
8   }
9
10  //zamena radku za radek s~prvkem s~nejvyssi hodnotou sloupce
11  swapRows(i,max);
12
13  //matice je singularni
14  if(A[i][i]==0)
15   error "Matrix is singular";
16
17  //odpocet radku od radku s~prvkem s~nejvyssi hodnotou
       sloupce
18  for(j=i+1;j<m;j++){
19   alpha = A[j][i] / A[i][i];
20   b[j] -= alpha * b[i];
21   for (k~= i; k~< n; k++) {
22    A[j][k] -= alpha * A[i][k];
23   }
24  }
25 }
```

Tímto algoritmem jsme schopni získat schodovitý tvar matice, z níž je posléze velmi jednoduché vypočítat jednotlivé koeficienty pomocí zpětné substituce.

Kód 4.2: Pseudokód zpětné substituce

```
1  for(i=m;i>=0;i--){
```

```
2     sum=0;
3     if(i<n){
4      for(j=i+1;j<m;j++){
5       sum+=A[i][j]*x[j];
6      }
7      x[i]=(b[i]-sum)/A[i][i];
8     }
9     else{
10     x[i]=1;
11    }
12   }
```

Výsledné koeficienty látek zastoupených v reakci získané pomocí funkce *gaussianElimination()* jsou sice řešením, ale ne úplným. Jelikož se pro vyčíslení rovnic používají pouze přirozená čísla, je třeba získané výsledky upravit. Toho je docíleno funkcí *getValues()*, která nejprve převede všechny desetinná čísla na zlomky, ty jsou posléze vynásobeny nejnižším společným násobkem jmenovatelů a vyděleny nejvyšším společným dělitelem takto získaných čísel. Výsledkem jsou tedy přirozená čísla, která se ve funkci *getReaction()* doplní před jednotlivé látky reakce, výjimkou jsou čísla 1 a 0, kdy 1 se před sloučeninu nedoplňuje a v případě 0 je sloučenina z reakce vymazána.

K získání nejvyššího společného dělitele za pomoci *Eukleidova algoritmu* je využita funkce greatestCommonDivider()[3].

Kód 4.3: Ekleidův algoritmus

```
1  while (b != 0) {
2   int temp = b;
3   b = a % b;
4   a = temp;
5  }
6  return a;
```

Výpočet molárních hmotností látek je implementován ve funkci *getWeights()*, která přijímá list prvků rovnice a matici získanou funkcí *getMatrix()*. Funkce *getWeights()* je znázorněna kódem níže, kde v poli $atomWeight$ jsou uloženy molární hmotnosti prvků, seřazených abecedně v poli *elements*, i-tý prvek *elements* má tedy molární hmotnost $atomWeight[i]$.

Kód 4.4: Získání molárních hmotností

```
1  int index;
2  for (int i = 0; i < matrix.length; i++) {
3   for (int j = 0; j < matrix[0].length; j++) {
4    //najdi i-ty prvek v~periodicke tabulce a ziskej protonove
       cislo
5    proton = elementsTable.indexOf(elementsArray.get(i));
6
```

[3]Zdroj: https://cs.wikipedia.org/wiki/Eukleidův_algoritmus

```
7     //pricti k~dosud ziskane vaze j-te latky reakce vahu prvku
          na i-te pozici v~poli prvku reakce ziskanou dle
          protonoveho cisla a vynasobenou poctem atomu tohoto
          prvku
8     weights[j] += Math.abs(atomWeight[proton] * matrix[i][j]);
9    }
10  }
```

Jelikož takto získané molární hmotnosti platí pro nevyčíslenou rovnici, je zapotřebí provést editaci vynásobením získaných hmotností koeficienty získanými vyčíslením rovnice. To je provedeno ve funkci *getSolvedWeights()*.

4.2 Rozpoznání znaků a samoopravný modul

Rozpoznávání znaků je provedeno za využití forku knihovny Tesseract pro platformu Android, modulu **tess-two**[4]. Ten obsahuje nástroje pro kompilaci knihoven Tesseract a Leptonica k využití na platformě Android a zprostředkovává API těchto knihoven ve formě Java API. Tesseract ke kategorizaci znaků využívá natrénovaná jazyková data, v této aplikaci je využita anglická sada *eng.traineddata*, jelikož není potřeba rozpoznávat diakritiku.

Jelikož celková množina znaků, jež se mohou objevit, je velmi omezená, je využito možnosti zvolit znaky, z kterých Tesseract vybírá nejlepšího kandidáta na pozici rozpoznávaného znaku pomocí řetězce *whitelist*, který obsahuje právě znaky, jež se mohou vyskytnout v rovnici.

Kód 4.5: Volání rozpoznávání textu

```
1  TessBaseAPI baseApi = new TessBaseAPI();
2  baseApi.setDebug(true);
3  baseApi.init(DATA_PATH, lang);
4  baseApi.setImage(image);
5  String whitelist = "
      ABCDEFGHIKLMNOPRSTUVWXYZabcdefghiklmnoprstuvwxyz0123456789
      +()[]->";
6  baseApi.setVariable(TessBaseAPI.VAR_CHAR_WHITELIST, whitelist
      );
7  recognizedText = baseApi.getUTF8Text();
```

Text rozpoznaný pomocí Tesseractu je posléze předán funkci *editOCRedText()*, ve které je nejprve pomocí soustavy regulárních výrazů upraven na tvar, se kterým dále pracuje vyčíslovací algoritmus. Je tedy provedena například záměna několika po sobě jdoucích *"-"* a následného znaku *">"* za znak *"="*. Další záměny jsou jednoduché neplatné kombinace znaků plynoucí ze špatného rozpoznání znaků *"I,l,1"*. Kombinace *"A1,AI,A[,A]"* jsou zde nahrazeny *"Al"*, obdobně jsou zaměněny kombinace *"F1,F[,F],TI,T[,T],T1"* a naopak všechnypro všechny znaky po nichž nemůže následovat *"l"*, ale je takto rozpoznáno, je *"l"* nahrezeno za *"I"*. Také jsou vymazány mezery. Dále je také vyřešena složitější problematika špatného rozpoznávání znaků *"*

",

[4] Zdroj:https://github.com/rmtheis/tess-two

″ a další časté chybně rozpoznané znaky.

Kód 4.6: Část programu sloužící ke korekci chyb při OCR

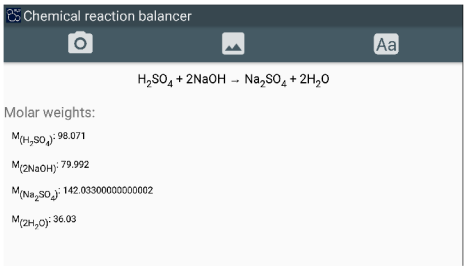
```
recognizedText=recognizedText
.replace(" ", "").replaceAll("O","O")
.replaceAll("[-=]+[^A-Z]*", "=")
.replace("+5","+S")
.replace(":", "=")
.replaceAll("AI|A1|A[|A]","Al")
.replaceAll("TI|T1|T[|T]","Tl")
.replaceAll("C1|CI", "Cl").replace("()","O")
.replaceAll("([^TACF])l","$1I")
.replaceAll("F1", "FI");
```

4.3 Struktura aplikace a uživatelské rozhraní

Aplikace je tvořena několika *aktivitami*. Každá aktivita obsahuje vlastní kód *(.java)* a popis grafického uživatelského rozhraní *(.xml)*. Hlavní obrazovka *MainActivity.java* má GUI popsáno pomocí *activity_main.xml* a dle uživatelovy interakce volá další aktivity s očekáváním výsledku průběhu volané aktivity metodou *startActivityForResult(intent, requestCode)*, kde v *intent* je popis spouštěné aktivity a *requestCode* je očekávaný kód, který je hlavní aktivitě předán spolu s návratovým kódem a výsledky spuštění volané aktivity po jejím ukončení. Volající aktivita na základě tohoto kódu je schopna určit, která jí volaná aktivita byla ukončena a přichází od ní data. Důležitým prvkem aplikace je také *AndroidManifest.xml*, v němž jsou deklarovány verze systému s kterými je aplikace kompatibilní, dále aktivita spouštěná při spuštění aplikace a především požadavky zisku práv k funkcím systému, jako jsou například přístup k fotoaparátu, umožnění čtení/zápisu na úložiště a podobně.

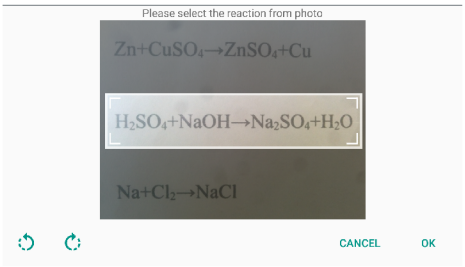
Souhrn aktivit, dialogů a jejich GUI

- **Domovská obrazovka** je tovřena aktivitou *MainActivity.java* , z níž jsou volány další aktivity a dialogy, GUI je popsáno *activity_main.xml*. Obsahuje textové pole, **TextView**, v němž je vyplněna vyčíslená rovnice, naformátovaná pomocí *HTML* značek, stejně tak další textové pole s výčtem molárních hmotností. Dále zde jsou samozřejmě umístěny ovládací prvky pro výběr způsobu zadání reakce.



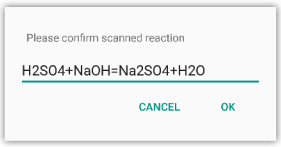
Obrázek 4.1: Domovská obrazovka s vyčíslenou rovnicí

- **Ořez snímku**, *CropImageActivity.java* je aktivitou, jež je volána po získání snímku určeného k rozpoznání rovnice. Uživatel v GUI vybere na snímku oblast, kterou zaujímá rovnice, toto je implementováno pomocí nástroje **Cropper**, který umožňuje vykreslení upravitelného ořezového pole nad nastaveným snímkem. Po nastavení této oblasti a potvrzení je volána metoda *getCroppedImage()*, vracející bitmapu označenou ořezovým oknem. Rozhraní obsahuje také prvky pro případnou rotaci snímku.



Obrázek 4.2: Obrazovka umožňující ořez snímku

- **Klávesnicový vstup** - aktivita *TextInputActivity.java* určená k zadání rovnice pomocí klávesnice. GUI (*text_input_activity.xml*) obsahuje pouze nápovědu, týkající se předpokládaného formátu rovnice, pole pro zadání rovnice a tlačítka pro potvrzení nebo návrat na domovskou obrazovku.
- **Vyfocení snímku** -je implementováno, pomocí spuštění již nainstalované aplikace poskytující uživateli fotoaparát. V případě většího počtu takovýchto aplikací je uživateli umožněno vybrat preferovanou aplikaci. Pro vyfocení snímku je samozřejmostí povolený přístup k fotoaparátu a zápisu na úložiště telefonu.
- **Potvrzení reakce** je řešeno pomocí pohledu, **View**, *confirmReactionDialogView* s vlastním grafickým rozhraním *activity_text_confirm.xml*, které obsahují, podobně jako aktivita pro přímé zadání rovnice z klávesnice, pouze tlačítka pro potvrzení či zrušení reakce a pole pro zadání rovnice, které je v tomto případě předvyplněno znaky rozpoznanými ze zadaného snímku.



Obrázek 4.3: Dialog sloužící k potvrzení rozpoznané rovnice

4.4 Testování a experimenty

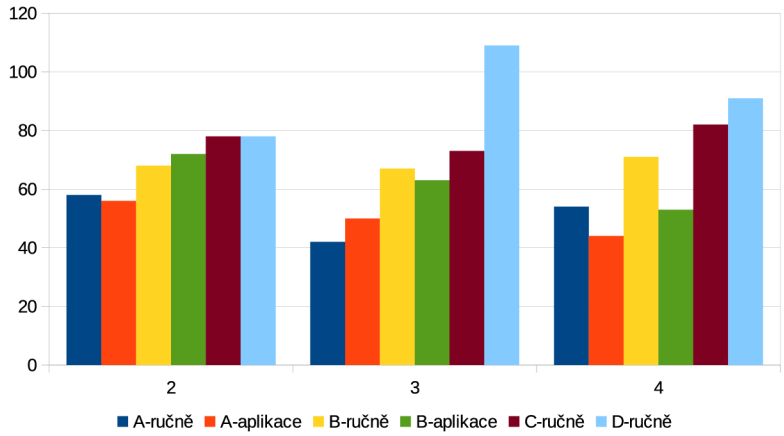
Testování je nedílnou součástí vývoje softwaru, proto ani v této nemůže chybět. Vzhledem k návrhu a implementaci grafického uživatelského rozhraní přímo na základě konzultací s uživateli a pozdějšími testery nás ovšem nebude příliš zajímat rychlost sžití se s aplikací, ale spíše porovnání rychlosti aplikace vůči klasické *"ruční"* metodě a validnost výstupů aplikace.

Vzhledem k vcelku specifickému okruhu potenciálních uživatelů, byli jako testovací uživatelé vybráni lidé, kteří již mají zkušenosti s vyčíslováním rovnic a tudíž je u nich nejlépe

demonstrovatelné, nakolik je výsledná aplikace užitečná v běžných případech užití. Byli tedy osloveni 4 testovací uživatelé, jimž byly poskytnuty podklady, jenž jsou vloženy k tomuto dokumentu jako v příloha v části Testovací data. Tato data lze rozdělit do dvou částí a to na úkoly 1 až 10, které se věnují nejen vyčíslování rovnic, ale také následnému vypočtení hmotnosti jedné z reagujících látek na základě hmotnosti jiné látky. Pro tyto úkoly byly zvoleny většinou jednodušší tipy rovnic, které lze vyčíslit i *z hlavy*, úlohy 2 až 4 nebylo dokonce ani nutné vyčíslovat. Naopak příklady 1 a 10 byly složitější. Úlohy 11 až 15 pak byly podobného rázu, ale nebylo nutné počítat hmotnosti látek, tudíž v případě ručního počítání odpadla nutnost hledání molárních atomových hmotností v periodické tabulce prvků. Ovšem vzhledem k vyšší náročnosti se již testování účastnili pouze uživatelé sběhlejší ve vyčíslování rovnic. Kromě samotného porovnání časů, lze tyto testy brát také jako validaci výstupů aplikace, kdy tyto byly porovnávány s vypočítanými výsledky.

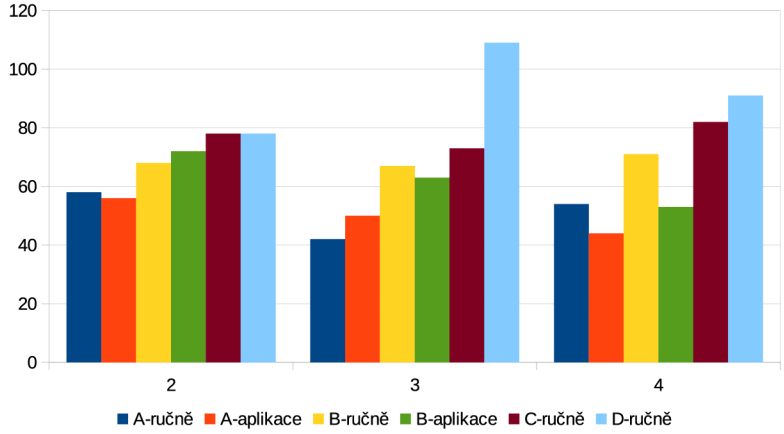
Úkol	A		B		C		D	
	ručně	aplikace	ručně	aplikace	ručně	aplikace	ručně	aplikace
1	4:23	1:13	5:41	01:32	9:20		8:42	
2	0:58	0:56	1:08	01:12	1:18		1:18	
3	0:42	00:50	1:07	01:03	1:13		1:49	
4	0:54	00:44	1:11	00:53	1:22		1:31	
5	1:42	00:53	1:49	01:05	1:49		2:04	
6	0:49	01:29	1:04	00:51	1:36		2:18	
7	0:57	00:58	1:31	00:55	1:25		1:59	
8	1:17	01:06	1:34	00:59	2:04		2:29	
9	1:19	01:08	1:50	01:03	2:53		2:27	
10	3:29	01:08	4:32	01:13	9:16		10:48	
11	3:50	0:32	5:08	0:34				
12	3:37	0:32	5:06	0:41				
13	3:36	0:28	3:54	0:33				
14	0:43	0:34	1:30	0:29				
15	3:37	0:46	4:18	2:30				
16	3:49	0:33	4:26	0:29				

Tabulka 4.1: Výsledky měření



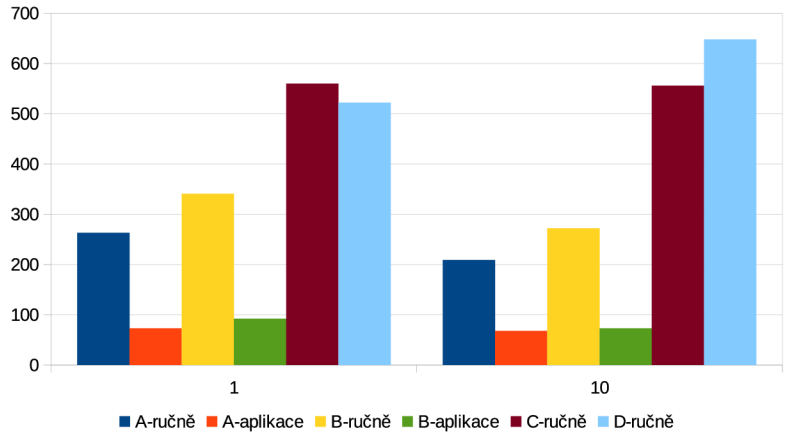
Obrázek 4.4: Dosažené časy u již vyčíslených rovnic v úloze s výpočtem hmotnosti

Jak lze vidět z grafu uvedeného na obrázku 4.4, u již vyčíslených reakcí je rychlost srovnatelná a rozdíl tvoří především doba hledání molárních atomových hmotností v periodické tabulce prvků. V těchto případech tedy lze předpokládat, že uživatel aplikaci potřebovat nebude.



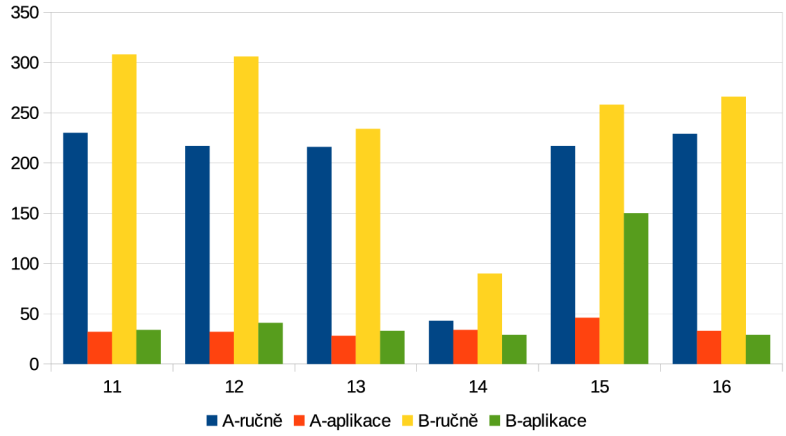
Obrázek 4.5: Dosažené časy u jednodušších rovnic v úloze s výpočtem hmotnosti

Na výše uvedeném grafu stále nelze sledovat výrazné zrychlení procesu vyčíslení a výpočtu hmotností na základě použití aplikace, především z důvodu triviality rovnic.



Obrázek 4.6: Dosažené časy u složitějších rovnic v úloze s výpočtem hmotnosti

Na grafu vyobrazujícím složitější rovnice se zadáním obsahující výpočet hmotností již lze vidět velký rozdíl mezi dobou vypočtu ručně a pomocí aplikace, totéž lze říci o časech dosažených u složitých rovnic (obrázek 4.7), kdy čas potřebný k výpočtu pomocí aplikace je téměř konstantní a daleko nižší než u ruční metody.



Obrázek 4.7: Dosažené časy u složitějších rovnic

Z výsledků měření lze vyčíst předpokládané, a to, že aplikace nemusí být zcela vždy potřebná k rychlému vyčíslení rovnice a vypracování navazujících úloh, především je-li rovnice velmi triviální, nebo v některých případech dokonce není potřebné rovnici vyčíslit. V těchto případech se ovšem hodí funkce aplikace, která poskytuje molární hmotnosti látek, uživatel je tedy nemusí hledat v tabulce periodických prvků, ve které může být pro někoho složitější se orientovat.

Naopak výraznou úsporu času je možné sledovat u složitějších úloh, kdy aplikace dosahovala až *10krát* nižších časů v případě zručnějších počtářů, u uživatelů, kteří vyčíslují pomaleji, je očekávaná úspora času ještě vyšší. Samotné vyčíslení rovnice je velmi rychlé a pohybuje se kolem půl minuty, u příkladů 1-10 je vyšší čas způsoben nutností dopočítat výsledné hmotnosti. Výrazné odchylky jsou potom způsobeny chybným rozpoznáním textu a nutností složitější úpravy rozpoznané rovnice.

Naměřené hodnoty mohou být trochu zkreslené vzhledem k zapojení testovacích uživatelů ve vývoji a tudíž dobré znalosti aplikace. Předpokládá se ovšem, že potenciální uživatelé by podobných hodnot měli dosahovat i po minimálním počtu použití aplikace.

Kapitola 5

Závěr

Cílem práce bylo navrhnout a implementovat mobilní aplikaci pro vyčíslení chemických rovnic pořízených fotoaparátem telefonu či zadaných textovým vstupem. Řešení bylo rozděleno na 2 podcíle a to získání textového tvaru rovnice z obrazových dat a dále vyčíslení takto získané rovnice. Bylo potřebné nastudovat teorii týkající se této problematiky, dále navrhnout řešení, to posléze realizovat a testovat správnost řešení. Dalším úkolem bylo zjištění vhodnosti navrženého uživatelského rozhraní a vyhodnocení výsledků. Realizace získání textu ze vstupních obrazových dat byla provedena pomocí open-source knihovny Tesseract, určené k optickému rozpoznávání znaků ze snímku, a následné korektury rozpoznaných znaků.

Druhý podcíl,tedy samotného vyčíslení chemických rovnic je řešeno pomocí několika algoritmů, mezi nimiž nechybí algoritmus *Gaussovy eliminační metody* pro získání řešení matice sestavené ze soustavy rovnic či *Euklidův algoritmus*, určený k získání nejvyššího společného dělitele dvojice čísel. Oba dílčí podcíle jsou spojeny ve formě aplikace pro mobilní platformu Android, pro kterou bylo navrženo jednoduché, avšak přehledné grafické uživatelské rozhraní.

Na začátku vytyčené cíle týkající se funkcionality aplikace se podařilo implementovat, včetně vyfocení snímku či výběru fotografie z galerie, rozpoznání tištěné rovnice a umožnění zadání rovnice pomocí klávesnice a samozřejmě vyčíslení aplikace. Aplikace oproti prvotnímu záměru obsahuje také výpočet a zobrazení molárních hmotností reagujících látek, což může být ve většině případů užití aplikace pro uživatele velice výhodné, jelikož tím ušetří spoustu času.

V průběhu testování byla potvrzena správnost algoritmu použitého k vyčíslování rovnic, avšak rozpoznávání textu z fotografie je závislé na mnoha faktorech ovlivňujících kvalitu snímku, jako je osvětlení, rozlišení fotoaparátu či rozostření vzniklé špatným vyfocením a nemusí tedy dojít ke správnému rozpoznání. Tomuto se částečně dá předejít modulem určeným ke korekci získaného textu a následným předložením uživateli pro potvrzení správnosti rovnice určené k vyčíslení. Aplikace bohužel není příliš spolehlivá v rozpoznávání rukou psaného textu, je zde tedy možnost dalšího vývoje, tak aby naplno pokryla i tyto vstupy. Dále by mohla být pro některé uživatele přínosná volba způsobu ukládání fotografií pořízených přímo v aplikaci, které je řešeno přemazáním starého snímku při vyfotografování nového. V dialogu potvrzení reakce získané z fotografie by také mohl být zobrazen výřez reakce vytvořený uživatelem v předchozím kroku. Některé z navržených úprav však nemusí mít žádaný přínos pro uživatele a naopak mohou způsobovat přílišnou robustnost aplikace, která byla vyvíjena s důrazem na jednoduchost a srozumitelnost.

Literatura

[1] *Android (operating system)*. [Online; navštíveno 15.5.2016].
URL https://en.wikipedia.org/wiki/Android_(operating_system)

[2] *Android (operační systém)*. [Online;navštíveno 15.5.2016].
URL https://cs.wikipedia.org/wiki/Android_(operacni_system)

[3] *Tesseract (software)*. [Online;navštíveno 15.5.2016].
URL https://en.wikipedia.org/wiki/Tesseract_(software)

[4] Android Developer: *Android Studio*. [Online;navštíveno 15.5.2016].
URL http://developer.android.com/tools/studio/index.html

[5] Břížďala, J.: *Chemické rovnice - vyčíslování a výpočty*. [Online;navštíveno 15.5.2016].
URL http://www.e-chembook.eu/chemicke-rovnice-vycislovani-a-vypocty

[6] Farhad, M.: *A Murky Road Ahead for Android, Despite Market Dominance*. Květen 2015, ISSN 0362-4331.

[7] Gargenta, M.: *Learning Android : Building Applications for the Android Market*. O'Reilly Media, 2011, ISBN 978-1-4493-9050-1.

[8] IUPAC: *Compendium of Chemical Terminology, 2nd ed.* Recognition Technologies Users Association, ISBN 0-9678550-9-8.

[9] Schantz, H. F.: *The history of OCR, optical character recognition*. Recognition Technologies Users Association, 1982, ISBN 9780943072012.

[10] Viktora, J.; Vymetálek, P.: *OCR*. Červen 2008, [Online;navštíveno 15.5.2016].
URL http://geo3.fsv.cvut.cz/vyuka/kapr/SP/2008_2009/vymetalek_viktora/index.html

Přílohy

Seznam příloh

Příloha A

Testovací data

1. Při reakci jodu s kyselinou dusičnou vzniká kyselina jodičná, oxid dusnatý a voda.

   $$I_2 + HNO_3 \rightarrow HIO_3 + NO + H_2O$$

   Kolik gramů kyseliny dusičné potřebujeme, aby vzniklo 6 gramů kyseliny jodičné? (Molární hmotnosti: M (HNO3) = 63 g/mol, M (HIO3) =175,9 g/mol)

2. Vypočtěte hmotnost oxidu siřičitého, která vznikl spálením 8g síry.

   $$S + O_2 \rightarrow SO_2$$

3. Reakcí železa s kyselinou sírovou vzniká vodík a síran železnatý. Vypočtěte hmotnost železa, kterou potřebujeme k přípravě 20g vodíku.

   $$Fe + H_2SO_4 \rightarrow H_2 + FeSO_4$$

4. Vypočtěte hmotnost uhličitanu vápenatého, kterou potřebujeme k výrobě 112kg páleného vápna (oxidu vápenatého).

   $$CaCO_3 \rightarrow CaO + CO_2$$

5. Vypočtěte hmotnost hliníku a hmotnost kyslíku potřebnou k přípravě 51g oxidu hlinitého.

   $$Al + O_2 \rightarrow Al_2O_3$$

6. Vypočtěte hmotnost vápníku potřebného k oxidaci na 112g oxidu vápenatého.

   $$Ca + O_2 \rightarrow CaO$$

7. Vypočtěte hmotnost oxidu fosforečného, který vznikl spálením 31g fosforu.

   $$P + O_2 \rightarrow P_2O_5$$

8. Vypočtěte hmotnost bromu, který se vyloučí z 206g bromidu sodného s přebytkem chlorové vody.

   $$NaBr + Cl_2 \rightarrow Br_2 + NaCl$$

9. Vypočtěte hmotnost chloridu hlinitého, který vznikl reakcí 105g chloru s práškovým hliníkem.

$$Al + Cl_2 \rightarrow AlCl_3$$

10. Vypočtěte hmotnost kyseliny fosforečné, která vznikla reakcí 15g kyseliny sírové s fosforem.

$$H2SO4 + P \rightarrow H3PO4 + SO2 + H2O$$

11. Vyčíslete následující chemické rovnice.

$$Al + NaOH + H2O \rightarrow Na[Al(OH)4] + H2$$

$$Ba(NO3)2 + Al2(SO4)3 \rightarrow BaSO4 + Al(NO3)3$$

$$CaO + NH4Cl \rightarrow CaCl2 + H2O + NH3$$

$$KCl + AgNO3 \rightarrow AgCl + KNO3$$

$$H2SO4 + NaOH \rightarrow Na2SO4 + H2O$$

$$C3H8 + O2 \rightarrow CO2 + H2O$$

Příloha B

Obsah CD

- Tato technická zpráva ve zdrojových souborech LaTeXa formátu .pdf ve složce /TZ
- Plakát ve složce /Poster
- Video ve složce /Video
- Instalační .apk soubor aplikace ve složce /apk
- Zdrojové kódy ve složce /ChemEval
- soubor s informacemi o projektu /README.txt

Příloha C

Plakát

CHEMEVAL

ALL YOU
NEED FOR
BALANCING
CHEMICAL
EQUATIONS