Pokyny pro vypracování

1) Nastudujte téma zpracování práškových záznamů z oblasti aplikované krystalografie, hlavně algoritmy pro výpočet kvality řešení tzv. figures of merit [1,2].
2) Navrhněte a implementujte algoritmus, která bude pro vybrané záznamy optimalizovat jejich kvalitu.
3) Na základě algoritmu z bodu 2) navrhněte metodu, která bude provádět tzv. indexaci práškových záznamů, kterou integrujete do balíku ParaCell.
4) Vyhodnocte účinnost metody z bodu 3) na datasetech dodaných vedoucím práce.
5) Porovnejte výsledky metody s ostatními metodami.

ALGORITMY
VÝPOČETNÍ KRYSTALOGRAFIE

Michal Dufek

Fakulta informačních technologií
Katedra teoretické informatiky
Vedoucí: doc. Ing. Ivan Šimeček, Ph.D.
11. května 2022
Obsah

Poděkování viii

Prohlášení ix

Abstrakt x

Seznam zkratek xi

1 Cíle práce 3

2 Teoretický základ 5
  2.1 Úvod do krystalografie 5
  2.2 Strukturní krystalografie 6
    2.2.1 Krystal 6
    2.2.2 Krystalové mřížky a základní buňka 6
    2.2.3 Millerovy indexy 7
    2.2.4 Krystalografické směry 8
    2.2.5 Reciproká mřížka 9
  2.3 Difrakce krystalu 9
    2.3.1 Braggův zákon 10
    2.3.2 Metody určení kvality hodnot (Figures of merit) 10
    2.3.3 Výpočetní metody indexace 12
    2.3.4 Oprava posunutí nuly 14

3 Implementace 17
  3.1 Oprava zeroshift 17
    3.1.1 Pomocná struktura theta_z_result 18
    3.1.2 Funkce get_closest() 19
    3.1.3 Funkce filter_theta_z() 19
    3.1.4 Funkce cluster_together() 19
    3.1.5 Funkce find_closest_range() 20
    3.1.6 Výstup programu 20
  3.2 Filtrování metod MGLS 20

4 Testování 21
  4.1 Optimalizace MGLS 22
    4.1.1 Ortorombické krystaly 22
    4.1.2 Monoklinické krystaly 23
    4.1.3 Triklinické krystaly 24
    4.1.4 Porovnání s ostatními algoritmy 24
  4.2 Zeroshift 25
    4.2.1 Praktické testy 25
    4.2.2 Teoretická oprava chyb 28
    4.2.3 Porovnání shlukovacích funkcí 29
Seznam obrázků

2.1 Diagram z díla Johannese Keplera se jménem Strena Seu de Nive Sexangula (O šestíúhelné sněhové vločce) z roku 1611. [1] ................................................................. 6
2.2 Fotka první rentgenové difrakce provedené na povrchu planety Mars, pořízené vozítkem Curiosity roku 2012. [2] ................................................................. 6
2.3 Bravaisova míříka a její základní vektory \( \vec{a} \) a \( \vec{b} \). Převzato z [3] ................................................................. 7
2.4 Značení parametrů \( a, b, c, \alpha, \beta \) a \( \gamma \) v základní buňce. Převzato z [3] ................................................................. 7
2.5 Hodnoty \( (hkl) \) pro 2 různé základní buňky. Převzato z [3] ................................................................. 8
2.7 Diagram rentgenové difrakce. Upraveno z [3] ................................................................. 10
2.8 Ilustrace grafu difraktogramu. Upraveno z [9] ................................................................. 11
4.1 Příklad grafu hodnot \( F^{20} \) a uplynulého času indexace v minutách pro krystal Forsterit. ................................................................. 22
4.2 Graf součtu pořadí metod ve vyhodnocení \( F^{20} \) pro ortorombické krystaly. (Čím nižší hodnota, tím lepší) ................................................................. 23
4.3 Graf poměru hodnoty \( F^{20} \) s časem indexace při použití dané metody pro ortorombické krystaly. ................................................................. 23
4.4 Graf poměru hodnoty \( F^{20} \) s časem indexace při použití dané metody pro monoklinické krystaly. ................................................................. 24
4.5 Příklad grafu hodnot \( F^{20} \) a uplynulého času indexace v minutách pro triklinický krystal Microcline. ................................................................. 25
4.6 Graf porovnávající indexační metody MGLS (značené method_steps), Dichotomy a Treor na datech monoklinického krystalu Deloryte při použití stejného omezení rozsahu. Metody označené písmenem V byly spuštěny s předem zadáným interвалem objemu základní buňky. ................................................................. 26
4.7 Graf výsledků testování opravy posunutí nuly pro triklinický krystal Microcline po vyfiltrování vzorce [4.1] ................................................................. 27
4.8 Graf poměru opravy zerohští chyby s grafem průměrného počtu prvků v difraktním záznamu. ................................................................. 28
4.9 Graf porovnání ideálního posunu opravy a opravy s nejnižší hodnotou err . ................................................................. 29
4.10 Graf úspěšnosti nejlepšího nalezeného opravného posunu za použití různých shlukovacích metod s různými parametry. Vypočítáno pro ortorombický krystal Berthierite. ................................................................. 30
4.11 Graf počtu nalezených kandidátů při použití různých shlukovacích metod s různými parametry a za použití obou filtrů s hodnotou 0.7. Vypočítáno pro ortorombický krystal Berthierite. ................................................................. 30
4.12 Graf hodnot \( F^{20} \) krystalu Cubanite s uměle vytvořeným posunutím nuly pro indexaci s nejlepší opravou, průměrnou opravou, nejlhorší opravou a žádnou opravou při vyfiltrování dat za pomocí vzorce [4.1] ................................................................. 31
Seznam tabulek

2.1 Tabulka výpočtu objemu základní buňky. Převzato z [5] . . . . . . . . . . . . . . 7
2.2 Tabulka relativní velikosti všech parametrů mřížky. Převzato z [3] . . . . . . . . 8
2.3 Tabulka výpočtu \(1/d^2_{hkI}\). Převzato z [3] . . . . . . . . . . . . . . . . . . . . 9
2.4 Tabulka metod algoritmu MGLS. (Výpis je pouze orientační, v programu jsou
          různé systémy implementovány trochu odlišně.) [12] . . . . . . . . . . . . . . . 15

4.1 Tabulka sestav, použitých k testování . . . . . . . . . . . . . . . . . . . . . . . . 21
4.2 Tabulka výsledků testu opravy posunu nuly monoklinického krystalu Uralborite
        bez šumu. (Vyfiltrované podle 4.1) . . . . . . . . . . . . . . . . . . . . . . . . . 26
4.3 Tabulka výsledků testu opravy posunu nuly pro ortorombický krystal Cubanite
        bez šumu. (Vyfiltrované podle 4.1) . . . . . . . . . . . . . . . . . . . . . . . . . 26
4.4 Tabulka výsledků použití opravy posunu nuly na data s uměle přidaným šumem. 27
A.1 Tabulka výsledků testování metod MGLS pro ortorombické krystaly . . . . . . . 38
A.2 Tabulka výsledků testování metod MGLS pro monoklinické krystaly . . . . . . . 39

Seznam výpisů kódu

3.1 Část kódu funkce main() pro výběr hlavní výpočetní metody. Kód zkrácen pro
       přehlednost. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17
3.2 Ukázka kódu struktury theta_z_results. Kód byl pro přehlednost zkrácen odebráním
       komentářů. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 19
A.1 Část kódu funkce zero_shift_dong() pro výběr hlavní výpočetní metody. Kód
       zkrácen pro přehlednost. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 35

Seznam algoritmů

1 ParaCell - Grid search, převzato z [11] . . . . . . . . . . . . . . . . . . . . . . . . 12
2 ParaCell - Parallel Treor search, převzato z [14] . . . . . . . . . . . . . . . . . . 13
3 ParaCell - Dichotomy search, převzato z [15] . . . . . . . . . . . . . . . . . . . . 14
4 ParaCell - MGLS [12] . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15
5 ParaCell - Zeroshift . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 18
6 ParaCell - Zeroshift - ClusterTogether
Tímto bych chtěl poděkovat vedoucímu této práce panu docentovi Ivanu Šimečkovi za nesmírnou trpělivost a velkou ochotu pomoci, či cokoliv vysvětlit. Dále bych rád poděkoval mým rodičům, sestře, přítelkyni a celé mé rodině za nepřestávající podporu, nikdy ne-konečnou trpělivost a za umožnění bezstarostného studia na této fa-kultě.
Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při 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. § 2373 odst. 2 zákona č. 89/2012 Sb., občanský zákoník, ve znění pozdějších předpisů, tímto uděluji nevýhodnou 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, avšak pouze k nevýdělečným účelům. Toto oprávnění je časově, teritoriálně i množstevně neomezené.

V Praze dne 11. května 2022

.................................
Abstrakt

Tato bakalářská práce se zabývá rozšiřováním balíčku ParaCell o novou výpočetní metodu přizpůsobenou k optimalizaci kvality indexace a to za pomocí opravy chyby posunutí nuly (zero shift error) před samotnou indexací difrakčních dat. Metoda byla poté otestována na řadě dat z každé krystalické soustavy a na testovacích datech dosáhla vysoké úspěšnosti, ale za možnou cenu vysoké výpočetní náročnosti. Dále se práce zabývá optimalizací indexační metody MGLS a na základě výsledků vytvoření její efektivnější varianty.

Klíčová slova   krystalická rentgenová difrakce, ParaCell, difraction zeroshift, MGLS, indexace krystalů, C++

Abstract

This bachelor’s thesis deals with extending the ParaCell package with a new computational method adapted to optimize the quality of indexing by correcting the zeroshift error before indexing the diffraction data. The method was then tested on a series of data from each crystal system and achieved high success on the test data but at a possible cost of high computational complexity. Furthermore, the work deals with the optimization of the MGLS indexing method and based on the results is created a more efficient variant.

Keywords   crystalic xray diffraction, ParaCell, diffraction zeroshift, MGLS, crystal indexation, C++
<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MGLS</td>
<td>MultiGrid Local Search</td>
</tr>
<tr>
<td>TREOR</td>
<td>Trial-and-error</td>
</tr>
<tr>
<td>SMT</td>
<td>Simultaneous Multithreading</td>
</tr>
</tbody>
</table>
Za celou dobu existence lidstva, nebylo nikdy jednodušší rozšiřovat a popularizovat vědu než dnes a mnoha oborům se daří dostávat do všeobecného povědomí, čímž nejen, že se o nich častěji píše a mluví, ale zároveň se o ně více lidí zajímá, více lidí je začne studovat a je poté větší šance, že ho někdo posune dále. I když tento začátek zní velice obecně, pomalu ale jistě mě navádí na obor Krystalografie, který podle mě patří právě do skupiny oborů, kterým se bohužel nedostalo takového štěstí a nejsou veřejností natolik známé. To je ovšem podle mě velká škoda, a to nejen proto, že se tato disciplína zajímá o tak nádherné objekty, ale také kvůli řadě velice zajímavých objevů, které by si tu pozornost zasloužily a díky kterým jsme dnes technologicky tam kde jsme. A tak doufám, že tato práce ponúží této velice zajímavé disciplíně k popularizaci mezi čtenáři tohoto textu a snad ji alespoň trošku posune vpřed.

V rámci této práce se v kapitole Teoretický základ zabýváme základy strukturní krystalografie, principy fungování práškové difrakce krystalů a především algoritmy či výpočetními metodami používanými na určování naměřených záznamů. Tyto znalosti poté využíváme v kapitole Implementace při přidávání nových funkcionalit do balíčku ParaCell. Mezi tyto nové metody patří vypočítání a opravení chyb posunutí nuly při měření krystalu a optimalizaci poměrně nové metody pro indexaci dat MGLS. Na závěr je budeme v kapitole Testování testovat a hodnotit.

Toto rozšíření by mělo pomoci pro přesnější určování neznámých krystalů pro všechny instituce využívající tento balíček například pro zjišťování složení a struktury nově vytvořených léků, chemických přípravků, nanomateriálů či dalších krystalických látek.

Zadání jsem si vybral z několika důvodů. Jako první bych zmínil důvod „nešuplíkového“ zadání, tedy aby práce nebyla okamžitě po hodnocení někam navýšena, ale aby byl její výsledek někým využíván. Další důvod je můj zájem o fotografovaní a přírodní vědy, jejichž krása (a to nejen ta krystalická) je podle mě nezměrná a dnes až příliš nedoceněná. Vzhledem k tomu, že difrakce krystalů je prakticky fotografování poloh atomů krystalů, bylo mi jasné, že tohle téma je pro mě jak dělané.
Seznam zkratek
Kapitola 1
Cíle práce

Vypracování bakalářské práce je rozděleno na několik kapitol. V kapitole Teoretický základ se zabýváme úvodem do krystalografie, jejími základními pojmy, dále si řekneme něco o principu difrakce krystalů, a nakonec budeme řešit výpočetní metody a algoritmy využívané pro zpracování difrakčních dat.

V další kapitole Implementace se budeme nejdříve zabývat návrhem algoritmu, poté se budeme zabývat jeho samotnou implementací, včetně implementace metody do balíku ParaCell.

Nakonec se v kapitole Testování budeme zabývat účinností naimplementované metody na datech, které nám dodal vedoucí práce a porovnáme je s již existujícími metodami.

Tato práce bude prospěšná pro instituce, které využívají balík ParaCell, pro zjišťování struktury a složení krystalických látek.
Cíle práce
Kapitola 2

Teoretický základ

2.1 Úvod do krystalografie

O zájmu lidí o krystaly máme důkazy již z antického Řecka. Tyto minerály si získaly své jméno z řeckého *krystallos* (v překladu led), kterým nazvali jednu odrůdu křemene, protože je průhledná a připomínala Řeckem třpytivé kousky ledu (dnes tento nerost nazýváme horský křišťál). Krystaly také nejspíše ovlivnily stavitele pyramid, kteří se mohli inspirovat jejich geometrií. Té si všiml i Platón, jehož učení o vzniku kosmu se zabývalo pravidelnými mnohostěny. O něco později vzniklo dílo *Peri lithón* od řeckého autora Theofraste z Eresu, které změňovalo i určování nerostů podle jejich fyzikálních vlastností (především podle vzhledu). Sám autor konstatoval, že některé nerosty, i když stejné nazvané, jsou ve skutečnosti odlišné. A tím již stavěl základy disciplíny zajímající se touto problematikou.

Ale i přes všechny tyto přínosy muselo uplynout více než jedno tisíciletí, než se krystalografie (nebo tehdy přesněji mineralogie) začala rozvíjet. To se podařilo Georgius Agricolovi, lékaři, který trávil část života v Jáchymově a dnes je obecně uznáván jako "Otec mineralogie". Právě jako záliba sbírat různé nerosty inspirovala knihu Johanna Mathesia, který později sepsal dílo o určování minerálů. Nedlouho poté se minerálů dostalo v českých zemích ještě větší pozornosti, a to přímo od císaře Rudolfa II., který je vášnivě sbíral. Tím osvět spojil Čech s krystaly stále nekončí. Velmi důležité příspěvky vzešly od císařova dvorního matematika, Johanna Keplera, který se zabýval souměrností sněhových vloček (obrázek [2.1]), díky čemuž je považován za zakladatele vědeckého oboru krystalografie.

Na tomto objevu poté stavěli další vědci, jako například důnský astronom Nicolaus Stenois, který objevil stálost úhlů u horského křišťálu nebo o sto let později Jean Baptiste Rome de Isles, který ve svém díle *Krystalografie* shrnul, že úhly sklonu stěn krystalů jsou stálé a neměnné. Tyto objevy postupně vedly v 19. století k objevu rozdělení krystalografických soustav zároveň profesorem Friedrichem Mohsem a profesorem Samuellem Weissem na 7 různých kategorií (kubická, tetragonální, hexagonální, trigonální, monoklínická a triklinická).


---

1Pro ujasnění, minerál je takový krystal, který vznikl samovolně v přírodě, tedy nebyl vypěstován.
2Friedrich Mohs je také objevitelem Mohsovi stupnice tvrdosti.
2.2 Strukturní krystalografie

2.2.1 Krystal

Pro začátek bychom si měli definovat co to krystal přesně je a v čem se odlišuje od ostatních „nekrystalových“ nerostů. Veškeré nerosty rozdělujeme na 2 kategorie, amorfní a krystalické. Tyto 2 druhy rozlišujeme především podle uspořádání atomů v pevném stavu.

Amorfní látky se vyznačují náhodně uspořádanými atomy, které jsou mezi sebou různě vzdáleny (od toho také vznikl jejich název, který v řečtině znamená bezvářový). Toto se projevuje například na jejich teplotě tání, kde díky různé vzdálenosti atomů, tím pádem i různé síle vazeb mezi atomy, taje látka postupně, protože se nejprve poruší slabší vazby a na silnější je potřeba vyšší teplota.


2.2.2 Krystalové míříky a základní buňka


Prostorové míříky jsou charakterizovány za pomocí tzv. „základní buňky“. Tato buňka je charakterizována velikostmi třích vektorů (a, b, c) a třemi úhly (α, β, γ), viz. obrázek 2.4. Pro správnou definici si dále musíme říci, že souměrnost základní buňky musí být stejná jako souměrnost míříky a zároveň objem základní buňky musí být co nejmenší. Tato vlastnost objemu je užitečná pro porovnání 2 různých základních buňek stejné míříky, protože i když se jejich parametry

Strukturní krystalografie

Obrázek 2.3 Bravaisova mířžka a její základní vektory \( \vec{a} \) a \( \vec{b} \). Převzato z [3]

Obrázek 2.4 Značení parametrů \( a, b, c, \alpha, \beta \) a \( \gamma \) v základní buňce. Převzato z [3]

Mohou lišit, jejich objemy se musí rovnat. Celkem je možných 14 různých druhů Bravaisových prostorových mířžek podle souměrnosti, které jsou dále rozděleny do 7 různých soustav a to je kubická, tetragonální, ortorombická, hexagonální, trigonální, monoklinická a triklinická. [3] [5]

<table>
<thead>
<tr>
<th>Typ</th>
<th>V</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kubická</td>
<td>( a^3 )</td>
</tr>
<tr>
<td>Tetragonální</td>
<td>( a^2c )</td>
</tr>
<tr>
<td>Ortorombická</td>
<td>( abc )</td>
</tr>
<tr>
<td>Hexagonální</td>
<td>( a^2c(1 - \cos^2 120^\circ)^{\frac{1}{2}} )</td>
</tr>
<tr>
<td>Trigonální</td>
<td>( a^2(1 - 3 \cos^2 \alpha + 2 \cos^3 \alpha)^{\frac{1}{2}} )</td>
</tr>
<tr>
<td>Monoklinická</td>
<td>( abc \sin \beta )</td>
</tr>
<tr>
<td>Triklinická</td>
<td>( abc(1 + 2 \cos \alpha \cos \beta \cos \gamma - \cos^2 \alpha - \cos^2 \beta - \cos^2 \gamma)^{\frac{1}{2}} )</td>
</tr>
</tbody>
</table>

Tabulka 2.1 Tabulka výpočtu objemu základní buňky. Převzato z [5]

2.2.3 Millerovy indexy

Také definujeme tzv. mířžové roviny, to jsou roviny, které procházejí alespoň 3 mířžovými body, co nejsou v přímce. Posloupnost těchto rovin nazýváme osnova rovin. Vzdálenost mezi těmito rovinami značíme \( d \) nebo případně \( d_{hkl} \). Tuto vzdálenost lze za předpokladu naší znalosti parametrů krystalové mířžky vypočítat se vzorcem v tabulce 2.3 Orientaci této roviny, popisují tzv. „Millerovy indexy“ \((h,k,l)\), které udávají počet počet rovin, které vektory \( a, b \) a \( c \), protínají. Typický příklad je uveden v obrázku 2.5, kde vektor \( a \) protíná mířžku, v \( \frac{1}{2} \) své délky, \( b \) také v \( \frac{1}{2} \). Millerovy indexy pro tento příklad jsou tedy \((210)\). [3]
Typ | Relativní velikost
--- | ---
Kubická | \( a = b = c \), \( \alpha = \beta = \gamma = 90^\circ \)
Tetragonální | \( a = b \neq c \), \( \alpha = \beta = \gamma = 90^\circ \)
Ortorombická | \( a \neq b \neq c \), \( \alpha = \beta = \gamma = 90^\circ \)
Hexagonální | \( a = b \neq c \), \( \alpha = \beta = 90^\circ, \gamma = 120^\circ \)
Trigonální | \( a = b = c \), \( \alpha = \beta = \gamma \neq 90^\circ \)
Monoklinická | \( a \neq b \neq c \), \( \alpha = \gamma = 90^\circ, \beta \neq 90^\circ \)
Triklinická | \( a \neq b \neq c \), \( \alpha \neq \beta \neq \gamma \neq 90^\circ \)

Tabulka 2.2 Tabulka relativní velikosti vůči parametrům mřížky. Převzato z [3]

2.2.4 Krystalografické směry

Krystalografické směry jsou popsány symbolem \((uvw)\) a jsou to nesoudělná čísla, která se rovájí vektoru z počátku do bodu mřížky. Platí \( t = ua + vb + wc \). Hodnoty směru musí být označeny v hodnotách vzájemně nesoudělných, a aby určovali pouze jeden bod v mřížce. Validní krystalografický směr je tedy například \((111)\), ale směry \((222)\) či \((\frac{1}{2} \frac{1}{2} \frac{1}{2})\) při popisování krystalů již nejsou validní.

Obrázek 2.5 Hodnoty \((hkl)\) pro 2 různé základní buňky. Převzato z [3]

---

3Viz. tabulka 2.1

4Na rozdíl od Millerových indexů, které určují natočení celé roviny.
Typ | \( \frac{1}{d_{hkl}^2} \)
--- | ---
Kubická | \( \frac{h^2+k^2+l^2}{a^2} \)
Tetragonální | \( \frac{h^2+k^2}{a^2} + \frac{l^2}{c^2} \)
Ortorombická | \( \frac{h^2}{a^2} + \frac{k^2}{b^2} + \frac{l^2}{c^2} \)
Hexagonální | \( \frac{\frac{h^2}{a^2} + \frac{k^2}{b^2} + \frac{l^2}{c^2}}{1+2\cos^2 \alpha - 3\cos^2 \alpha} \)
Trigonální | \( \frac{\frac{h^2}{a^2} \sin^2 \beta + \frac{k^2}{b^2} + \frac{l^2}{c^2} \sin^2 \gamma - \frac{2hl}{ac} \cos \beta \cos \gamma + 2\frac{hk}{ab} \cos \alpha \cos \beta - \frac{2h^2}{ac}}{1+2\cos^2 \alpha - 3\cos^2 \alpha} \)
Monoklinická | \( \frac{\frac{h^2}{a^2} \sin^2 \beta + \frac{k^2}{b^2} + \frac{l^2}{c^2} \sin^2 \gamma - \frac{2hl}{ac} \cos \beta \cos \gamma + 2\frac{hk}{ab} \cos \alpha \cos \beta - \frac{2h^2}{ac}}{1+2\cos^2 \alpha - 3\cos^2 \alpha} \)
Triklinická | \( \frac{\frac{h^2}{a^2} \sin^2 \alpha + \frac{k^2}{b^2} \sin^2 \beta + \frac{l^2}{c^2} \sin^2 \gamma + \frac{2hl}{ac} \cos \beta \cos \gamma - \frac{2hk}{ab} \cos \alpha \cos \beta - \frac{2h^2}{ac}}{1+2\cos^2 \alpha - 3\cos^2 \alpha} \)

Tabulka 2.3 Tabulka výpočtu \( \frac{1}{d_{hkl}^2} \). Převzato z [3]

Tyto násobky jsou ale využívány při vypočítávání rozměrů krystalu (součást tzv. „indexace“ o které si více povím v další kapitole) a aktivně se tak používají (čím lépe zlepší přesnost výpočtů).

### 2.2.5 Reciproká mřížka

Jiného způsobu popisu krystalu lze také dosáhnout za pomoci tzv. reciproké mřížky. Ta se používá, protože umožňuje snadněji výpočty s daty. Parametry reciproké mřížky jsou často značeny s hvězdíčkou, např. \( a^* \), aby byly rozlišovatelné od jejich hodnot v přímé mřížce.[3]

Pro konstrukci reciproké mřížky, vedeme k rovinám \( hkl \) přímé mřížky z počátku souřadnic kolmice, ke kterým naneseme vzdálenost \( \frac{1}{d_{hkl}} \). Získané body pak tvoří reciprokou mřížku. Z tohoto nám vyplývá, že vektor \( a^* \) je kolmé k rovinným určeným vektory \( b \) a \( c \).

### 2.3 Difrakce krystalů

Difrakce je fenomén, který nastává když se vlna střetává s předmětem, hranou předmětu či otvorem a při tomto blízkém setkání vlna zatačí a pokračuje dál od objektu v jiném směru. Tento jev se využívá například v analýze světelného spektra za pomoci difrakční mřížky nebo také může nastat při fotografování s velmi malou clonou (neboli s velmi vysokým clonovým číslem).

U krystalů, se vlastností difrakce využít, protože vlny o velikosti okolo 1.5 \( \AA \) změní svůj směr při blízkém setkání s atomy. Tím se nám prakticky „odrazí 2D mapa“ poloh atomů. Dnes se difrakce krystalů zaznamenává na tzv. „difraktometrech“, přístrojích, které jsou konstruovány, aby měřily odrazy záření s co největší přesností a tyto odrazy se zaznamenávají na tzv. „difraktogramy“. Tyto záznamy z digraktoramu poté musíme zpracovat, tedy vypočítat z naměřených hodnot parametry měřeného krystalu \( a, b, c, \alpha, \beta \) a \( \gamma \), tento proces nazýváme jako tzv. „indeces“ a v druhé části této kapitoly rozebíráme způsoby jak lze tuto indexaci provést.

5Ångström, jednotka vzdálenosti pojmenovaná podle švédského astronoma Anders Jonas Ångström (1814-1847), 1Å = 0.1nm
2.3.1 Braggův zákon

\[ 2d_{hkl} \sin \theta = \lambda \]  \hspace*{1cm} (2.1)

Ze záznamů „diffraktogramů“, naměřených s rentgenovým zářením o vlnové délce \( \lambda \), můžeme za pomocí tzv. „Braggovy rovnice“ vypočítat mezirovinnou vzdálenost \( d_{hkl} \), jejíž obrácená hodnota na druhou (tedy \( \frac{1}{d_{hkl}^2} \)) se nazývá \( Q_{hkl} \). Pokud známe parametry \( (a, b, c, \alpha, \beta, \gamma) \) lze pro vypočítání hodnoty \( Q \) využít vzorce z tabulky 2.3.

2.3.2 Metody určení kvality hodnot (Figures of merit)

Především dříve, kdy byly již zmíněné algoritmy na výpočet parametrů krystalu známé, bylo velmi drahé krystaly indexovat, zejména z důvodu, že jejich přesné indexování je výpočetně velmi náročné a na tehdejším hardwaru musely běžet velmi dlouho. Z toho důvodu vznikla potřeba vymyslet nějaké kritérium, díky kterému lze určit, zdali jsou vypočítané parametry dostatečně přesné.
2.3.2.1 De Wolffovo $M_N$ kritérium

První známé kritérium vzniklo v 70. letech, kdy ho do světa publikoval původem indonéský profesor Pieter Maarten de Wolff a nazval ho $M_N$. Výsledek vzorce 2.2 kde $Q_N$ je hodnota $Q\left(\frac{1}{2}\right)$ pro $N$-tu naměřenou hodnotu, $\lvert \Delta Q \rvert$ je průměrný absolutní rozpor mezi $N$ naměřenými hodnotami $Q$ a $N$ vypočítanými hodnotami $Q$. Hodnotu $N$ lze určit více způsoby, ovšem musíme poznamenat, že pro hodnoty $N > 30$ ztrácí metoda přesnost z důvodu šumu v naměřených hodnotách a z toho důvodu se nejčastěji používá $N = 20$. U výsledků tohoto kritéria platí, že čím vyšší výsledek, tím vyšší přesnost indexace. Obecně počítáme, že u výsledků o hodnotách od 20 do 60 je indexace dostatečně přesná pro rutinní práci. U výsledků s hodnotou nižší než 10 se může jednat o naprosto špatné parametry základní buňky. [8][7]

$$M_N = \frac{Q_N}{2 \times \lvert \Delta Q \rvert \times N_N} \quad (2.2)$$

2.3.2.2 $F_N$ kritérium

Poměrně krátece po vzniku metody $M_N$ se začaly projevovat její nevýhody, a to především z důvodu závislosti hodnoty 20. $Q$ na objemu výsledné buňky nebo souvislost s vysokými hodnotami a krystalické soustavy. Z toho důvodu vzniklo o několik let později nové kritérium, tzv. „$F_N$“ kritérium, které slibovalo tyto nedostatky napravit. [7]

$$F_N = \frac{1}{\Delta 2\theta} \left(\frac{N}{N_{\text{poss}}}\right) \quad (2.3)$$

Metoda $F_N$ byla roku 1978 publikována spolu se vzorcem 2.3 kde $\lvert \Delta 2\theta \rvert$ je průměrný absolutní rozpor mezi naměřenou a vypočítanou hodnotou, a $N_{\text{poss}}$ je počet možných difrákčních linií (záznamů v difraktogramu) až do $N$ této špatně linií. Výsledek $F_N$ funguje podobně jako hodnota $M_N$, tedy čím vyšší je, tím přesnější výsledky máme. Například pro hodnotu $F_N = 100$ je průměrný rozpor $2\theta \leq 0.01^\circ$, pro hodnotu $F_N = 50$ je průměrný rozpor $2\theta \leq 0.02^\circ$, atd. Pro naše potřeby uvažujeme, že hodnoty $F_N > 20$ jsou dostatečně přesné. U výběru hodnoty $N > 30$ nastává stejně jako v případě kritéria $M_N$ ztráta přesnosti a proto se nejčastěji používá hodnoty $N = 20$. [7]
2.3.3 Výpočetní metody indexace

Při měření krystalu se zaznamenávají hodnoty theta a intenzity přijímaného záření do hlavy difraktogramu. Nyní nám pro správnou indexaci zbývá ten nejtěžší krok, a to zjistit přesné rozměry stran \(a\), \(b\) a \(c\) a případně i úhly \(\alpha\), \(\beta\) a \(\gamma\) takové, aby pro danou krystalickou soustavu (viz. tabulka 2.3) vycházely správné rozměry po dosazení všech naměřených dat. Tím získáme jak již zmíněné rozměry, ale také parametry \(hkl\), kterými dokážeme určit směr roviny daného záznamu.

Metoda na indexaci krystalu je více, každá s různou přesností, výpočetní náročností či dokonce úspěšností řešení. V programu ParaCell se aktuálně nachází celkem 6 různých metod, Grid, TREOR, Dichotomy, MGLS, ITO a Rules. My si bližší popíšeme čtyři implementace, a to metody Grid, Treor, Dichotomy a MGLS.

2.3.3.1 Grid

Grid je nejjednodušší a naivní metoda. Nejdříve si pro všechny hledané parametry určíme intervale na kterých budeme hledat výsledek. Tyto intervaly poté rovnoměrně rozdělíme a postupně všechny vyzkoušíme. Pokud s těmito parametry určí metoda určení kvality vysokou hodnotu kvality, jsou tyto parametry uloženy jako kandidáti na výsledek. Pro urychlení výpočtu je v balíčku ParaCell například také naimplementována možnost zadat interval očekávané velikosti základní buňky, pokud velikost buňky vypočítané z parametrů, které algoritmus aktuálně zkouší vyjde velikost mimo zadany interval, počítání kvality řešení se přeskočí. Toto urychlení je z v rámci jednoduchosti z pseudokódu vynecháno. [10][11][12]

**Algoritmus 1:** ParaCell - Grid search, převzato z [11]

1. for \(a\) in range \((aMin, aMax)\) by \(aStep\) do
2.   for \(b\) in range \((bMin, bMax)\) by \(bStep\) do
3.     { and so on for \(c, \alpha, \beta, \gamma\) }
4.     calculate reciprocal values \(a_{ij}\) from \(a, b, \ldots\)
5.     save the reciprocal values in buffer as a new cell
6.     if buffer is full or no more cells will be generated then
7.         evaluate it and put results in database

2.3.3.2 TREOR

První funkční verze programu (a algoritmu) TREOR (trial-and-error) vznikl již okolo roku 1964. V té době musel ještě být naprogramovaný ve strojovém jazyce a tak nebylo možné implementovat tak pečlivý výpočet. Až 10 let po vydání nové verze byla vytvořena nová verze, s novými funkcionalitami tentokrát naprogramovaná ve Fortranu. Od té doby bylo vydáno několik nových verzí s postupným rozšiřováním funkcionality pro zpřesnění výpočtů. [13][14][12]

Aktuální implementace tohoto algoritmu v balíčku ParaCell je založena na technologiích paralelních výpočtů OpenMP pro procesory a CUDA pro grafické karty Nvidia. [14][12]

Princip tohoto algoritmu tkví ve zkoušení různých kombinací Millerových indexů na výbrané naměřené hodnoty. Velikost souboru naměřených hodnot je závislá na počtu neznámých proměnných v dané krystalické soustavě (viz. tabulka 2.2). Pro všechny tyto soubory algoritmus nejdříve vypočítá teoretické hodnoty \(hkl\) a poté se s nimi porovná zbylé změřené hodnoty krysztalu. Pokud ostatní hodnoty vyjdou přibližně přesně, je základní buňka uložena. [14][12]

\(^{6}\)například přidání výpočtů \(M_{20}\)
Algoritmus 2: ParaCell - Parallel Treor search, převzato z [14]

Output: $DB = \text{database of proposed lattice parameters}$

1 $DB \leftarrow \{\}$
2 $DB_2 \leftarrow \{\}$
3 \textbf{for} $i \leftarrow 1, STEPS$ \textbf{do}
4 \hspace{1em} $L \leftarrow \text{set of } s \text{ distinct lines}$
5 \hspace{1em} \textbf{for} $j \leftarrow 1, no\_combs(n_{hkl}, s)$ \textbf{do}
6 \hspace{2em} $M_{\text{curr}} \leftarrow j$-th combination of Miller’s indicies
7 \hspace{2em} \textbf{if} $M_{\text{curr}}^{-1}$ \textbf{exists} \textbf{then}
8 \hspace{3em} $M_{\text{inv}} \leftarrow M_{\text{curr}}^{-1}$
9 \hspace{3em} \textbf{for} $l \leftarrow 1, no\_perms(L)$ \textbf{do}
10 \hspace{4em} $\vec{Y} \leftarrow l$-th permutation of $L$
11 \hspace{4em} $\vec{A} \leftarrow M_{\text{inv}} \cdot \vec{Y}$
12 \hspace{4em} \textbf{insert} $\vec{A}$ \textbf{info} $DB_2$
13 \textbf{end for}
14 \textbf{end for}

15 \textbf{for} $i \leftarrow 1, \lvert DB_2 \rvert$ \textbf{do}
16 \hspace{1em} $\vec{A} \leftarrow i$-th item of $DB_2$
17 \hspace{1em} $(i, err) \leftarrow \text{EvaluateError}(\vec{A})$
18 \hspace{1em} \textbf{if} $(i \geq \text{threshold}) \& (\text{error} < \text{threshold2})$ \textbf{then}
19 \hspace{2em} $\vec{A} \leftarrow \text{RefineCell}(\vec{A})$
20 \hspace{2em} $fom \leftarrow \text{CalculateFOM}(\vec{A})$
21 \hspace{2em} \textbf{if} $fom \geq \text{threshold3}$ \textbf{then}
22 \hspace{3em} \textbf{insert} $\vec{A}$ \textbf{into} $DB$
23 \hspace{2em} \textbf{end if}
24 \hspace{1em} \textbf{end if}
25 \textbf{end for}

26 \textbf{return} $DB$
2.3.3.3 Dichotomy

Mezi další pokročilejší algoritmy obsažené v programu ParaCell se řadí „Dichotomy“. Tento algoritmus je stejně jako Treor a Grid implementován paralelně a to za pomocí OpenMP. \[15\]

Dichotomy nejdříve vytvoří základní buňku podle intervalů hledání, které zadá uživatel. Tato buňka se poté otestuje jestli v ní mohou, či nemohou být řešení (na základě objemu). Pokud může, tak se rozdělí na menší buňky, které jsou poté zařazeny do fronty na vyšetření. Pokud v ní nemůže být řešení, tak je buňka vyřazena z hledání. Tento proces se opakuje do té doby, dokud se nenajde řešení nebo dokud nedojde vyšetřované základní buňky. Podrobnější popis algoritmu lze nalézt v pseudokódu \[15\].

```
Algoritmus 3: ParaCell - Dichotomy search, převzato z \[15\]

\[1\] push \( B \) onto stack
\[2\] while \( solution \) is not found AND stack is not empty do
\[3\] pop \( B \) from the stack
\[4\] \( Q_{calc} \leftarrow \text{Generate}(B, HKLset) \)
\[5\] if \( \text{Verify}(Q_{calc}, Q_{obs}) = \text{True} \) then
\[6\] if \( \text{SmallEnough}(B) = \text{True} \) then
\[7\] \( solution \leftarrow solution \cap B \)
\[8\] else
\[9\] \( B_{new} \leftarrow \text{Divide}(B) \)
\[10\] push all \( B_{new} \) onto the stack
\[11\] return \( solution \) or "solution not found"
```

2.3.3.4 MGLS

Algoritmus MGLS (MultiGrid Local Search) je navržený a implementovaný vedoucím mě práce, panem doc. Šimečkem. \[12\]

Pro vysvětlení algoritmu uvažme, že pro aktuálně indexovaný krystal musíme vypočítat \( n \) parametrů.\[7\] Algoritmus nejdříve vygeneruje po určených intervalech parametry krystalu z určeného rozsahu. Tyto hodnoty poté dosadí do rovnic výpočtu \( Q \) a pro jednotlivé řádky zjistí několik (v tuto chvíli vždy 2) nejbližší \( h, k, l \). Na základě vybrané kombinace \( h, k, l \) se sestaví soustava rovnic, která se vyřeší pro všechny neznámé parametry. Následuje výpočet \( Q \) pro všechny zbylé naměřené hodnoty. Dále po vypočetění kvality výsledků určí, o jak dobrý výsledek se jedná. Pokud je vypočtena kvalita vysoká, uloží si program tyto parametry do databáze možných kandidátů. \[12\]

Pro výběr prvků, které budou použity na výpočet je v programu implementováno 8 metod, každá s jiným přístupem k výběru (viz. tabulka \[2.4\]). My se ještě v kapitole Implementace setkáme s metodami 1, 2, 3, 4 a 8.

2.3.4 Oprava posunutí nuly

Jedna z nejčastějších systematických chyb při zachycování difrakce krystalu je tzv. chyba posunutí nuly. Ta nastává při nesprávné kalibraci nulového bodu, tedy nesprávné nastavení místa,\[7\]

\[8\]

---

\[7\] Podle krystalického systému se jedná o některé nebo všechny z parametrů \( a, b, c, \alpha, \beta \) a \( \gamma \). Viz. tabulka \[2.2\].

\[8\] Viz. tabulka \[2.3\].
Algoritmus 4: ParaCell - MGLS \[12\]

Input: SearchIntervals
Input: StepsNumber

1. \(\text{Steps} \leftarrow \text{SearchIntervals}/\text{StepsNumber} \) // For each unknown cell parameter

2. \textbf{foreach} \textbf{Step} \textbf{in} \text{Steps} \textbf{do}

3. \(\text{Cell} \leftarrow \text{set cell parameters from Step}\)

4. \(Q_{\text{list}} \leftarrow \text{GetMeasuredTheta(size of } P\text{)}\)

5. \text{Solve equations for } h, k \text{ and } l \text{ in } Q_{\text{list}}

6. \text{Calculate quality of base cell with } h, k, \text{ and } l \text{ substituted to all other measured elements}

7. \textbf{if} \ Quality \text{ of base cell} > \text{threshold} \textbf{then}

8. \text{Insert base cell into } \text{DB}

### Tabulka 2.4 Tabulka metod algoritmu MGLS. (Výpis je pouze orientační, v programu jsou různé systémy implementovány trochu odlišně.) \[12\]

| Metoda 1 | vybere vždy \(n\) nejbližších po sobě jdoucích řádek |
| Metoda 2 | řádky jsou seřazeny podle chyby, následně se vybere \(n\)-tice až \(m\)-tice po sobě jdoucích řádek |
| Metoda 3 | vybere vždy kombinaci \(n\) po sobě jdoucích řádek |
| Metoda 4 | funguje stejně jako metoda 2, ale řádky jsou seřazeny dle 2. nejmenší chyby |
| Metoda 5 | funguje stejně jako metoda 1, ale přeskanuje některé označené záznamy |
| Metoda 6 | funguje stejně jako metoda 1, ale přeskanuje některé označené záznamy a je použitý až 2. nejbližší |
| Metoda 7 | jako metoda 5, ale odspodu |
| Metoda 8 | podobná metodě 3, ale částečně jiný výběr kombinace |

kde je rameno v nulovém úhlu. Jedná se o relativně jednoduše opravitelnou chybu, ale dostatečně velká chyba nám může tuto snahu zcela znemožnit a tím i znehodnotí naměřené údaje. Na opravu tohoto problému bylo za mnoho let výzkumu vyvinuto více různých výpočetních metod, které tuto chybu opravovaly, ovšem drtivá většina je aplikována až po dokončení indexace, tedy v době, kdy již vímame samotné rozměry a \(hkl\) krystalu. My ovšem budeme používat jiný postup, který lze použít na opravu chyby již před samotnou indexací. \[16\]

Princip tohoto postupu, spočívá v nalezení 2 úhlů, \(\theta\) a \(\theta'\) o kterých se domníváme, že jejich \(hkl\) jsou v poměru \(m\), viz. rovnice \[2.5\]. Poté tyto hodnoty dosadíme do rovnice \[2.5\], jejíž výsledek je předpokládaná chyba pro tento úhel. Po vypočítání všech pravděpodobných kandidátů na chybu provedeme jejich průměr a odečteme chybu od celku. \[16\]

\[
d_{hkl} = md_{h'k'l'}, \text{ kde } m \in \mathbb{Z} \text{ \text{ \begin{align*} h' &= mh \\ k' &= mk \\ l' &= ml \end{align*}}} (2.4)
\]

\[
2\theta_z = 2 \arctan[(\sin \theta' - m \sin \theta)/(m \cos \theta - \cos \theta')]) \quad (2.5)
\]

Po konzultaci s vedoucím práce nebudeme ve výsledné implementaci algoritmu ve vzorce \[2.4\] uvažovat \(m \in \mathbb{Z}\) a \(m \in [1, 2, 3, 4, 5, 6]\). Výšší čísla nebudeme používat, protože takto vysoké násobky nejsou pro reálná data pravděpodobné.

V implementaci také nebudeme používat jako finální výsledek čistý průměr \(\theta_z\) všech výsledků, protože někdy může vyjít velmi vysoký posun, který není možné aby nastal a ze zbylých výsledků
budeme vybírat možné kandidáty podle 2, námi implementovaných metod. Podrobnosti si povíme v kapitole 3.1, která zabývá samotnou implementací.
Kapitola 3
Implementace

Praktická část této práce se věnuje rozšíření balíčku ParaCell, napsaný v jazyce C++ a využívá technologie jako je CUDA či OpenMP. V této práci ovšem budeme používat pouze čisté C++, případně součásti knihovny std jako je \texttt{std::vector}, \texttt{std::string} nebo \texttt{std::sort}. Implementace je provedena ve stejném stylu jako je aktuální stav programu, tedy velmi funkcionální a žádné objektové prvky.

### 3.1 Oprava zeroshift

První část se týká vytvoření nové metody, která vypočítává předpokládané posunutí nuly na vstupních datech. Implementace této funkce je možná dvěma způsoby:

- Předzpracování dat po kterém by se následně spustila indexace pomocí předem zvoleného algoritmu.
- Samostatné vypočítání chyby a následně vygenerování opravených dat do datového formátu programu.

Po diskuzi s vedoucím byla vybrána druhá možnost, tedy aby program pouze počítal zeroshift, čímž se tato funkcionalita stává v rámci programu ekvivalentní s metodami indexace, tj. hlavní účel běhu programu. Pro tento důvod bylo potřeba přidat do parseru konfiguračního souboru novou možnost hodnotu pro parametr \texttt{METHOD}, a to \texttt{ZEROSHIFT}. Aktuální implementace volby metody je provedena přes proměnnou datového typu \texttt{int} ve struktuře \texttt{configuration}. Při příležitosti přidávání této nové hodnoty byl také vytvořen pro lepší přehlednost do souboru \texttt{common.h} nový \texttt{enum}, který obsahuje všechny aktuálně implementované metody s jejich celočíselnými hodnotami, aby se nemusel měnit starý kód, viz. výpis kódu 3.1.

#### Výpis kódu 3.1
Část kódu funkce \texttt{main()} pro výběr hlavní výpočetní metody. Kód zkrácen pro přehlednost.

```c++
switch (cfg.method1){
  ... skipping other indexing methods
  case method1::ITO: // ITO search
    ito_search(mea, &(DB[0]), &cfg);
    break;
  case method1::RULES: // RULES search
    rules_tri(mea, &(DB[0]), &cfg);
}
```
Jak lze dále vidět ve výpisu kódu 3.1, program se po dokončení metody `zero_shift_dong()` ukončuje, protože po dokončení běhu této funkce je vše dopočítáno a nemusí se dále data nijak zpracovávat. Zjednodušený běh celé této funkce je znázorněn v pseudokódu 5.

**Algoritmus 5: ParaCell - Zeroshift**

1. `CalculatedShifts ← {}`
2. `Candidates ← {}`
3. `for i ← 1, DATA do`
4. `for o ← 1, DATA do`
5. `q1 ← Q[i]`
6. `q2 ← Q[o]`
7. `ratio ← √(q1 / q2)`
8. `m ← GetClosest(ratio)`
9. `err ← |q2 - q1 * m2|`
10. `θz ← 2 \arctan(\sin(θ' - m \sin θ)/(m \cos θ - \cos θ'))`
11. `Push θz a err to CalculatedShifts`

12. `FilterThetaZ(CalculatedShifts, FILTER_LIMIT)`
13. `Sort(CalculatedShifts) // by θz`
14. `Candidates ← ClusterTogether(CalculatedShifts)`
15. `Candidates ← FindClosestRange(CalculatedShifts)`
16. `FilterThetaZ(Candidates, SECOND_FILTER_LIMIT)`
17. `Sort(Candidates) // by err`
18. `return Candidates`

V algoritmu se vyskytuje proměnná `err`, což je odhadovaná chyba parametru `m`, použitím ve výpočtu `θz` zobrazeném ve vzorce 2.5 nebo v algoritmu 5 na řádku 10. Tato chyba se vypočítává za pomoci vzorce |q2 - q1 * m2|. Tento vzorec byl navržen vedoucím práce.

V příloze A.1 je zkrácená ukázka (byly odebrány části sloužící k výpisu průběžných výsledků, části sloužící k ukládání výsledků do souboru a další nedůležité součásti kódu) celé implementace této funkce.

### 3.1.1 Pomocná struktura `theta_z_result`

Pro přehlednost bylo potřeba v implementaci vytvořit strukturu, která by umožňovala jednodušší práci s více vlastnostmi jednoho vypočteného záznamu. Tato struktura byla nazývána `theta_z_result` a jak lze vidět i ve výpisu kódu 3.2 její proměnné jsou následující:

- **double theta_z** tato proměnná je určena k uložení samotného výsledku vzorce v algoritmu 5 na řádku 10.

- **double used_ratio_m** je proměnná, která je výsledek volání funkce `get_closest()` a byla použita pro výpočet `theta_z`. Tato proměnnou ukládáme, protože se v programu spolu s dalšími parametry vypisuje.
VÝPIS KÓDU 3.2  Ukázka kódu struktury theta_z_result. Kód byl pro přehlednost zkrácen odebráním komentářů.

```c
struct theta_z_result {
    double theta_z = 0;
    double used_ratio_m = 0;
    double err = 0;

    bool is_valid = true;

    enum source_enum {
        RANGE, CLUSTERING, CALC
    };

    source_enum source = source_enum::CALC;
};
```

- double err je určena k udržení hodnoty vypočtené chyby, aby s ní bylo poté možno seřadit výsledky od nejpravděpodobnější po nej nepravděpodobnější.

- enum source_enum je množina obsahující různé způsoby vzniku instance této struktury, při vytvoření ve funkcí FindClosestRange() bude uchovávat hodnotu RANGE, atd.

3.1.2 Funkce get_closest()  

V implementaci opravy zeroshift celkem voláme 4 funkce (jak již bylo naznačeno v algoritmu 5), ta první je vyobrazena na osmém řádku se jménem get_closest(). Tato funkce přijímá pomoc (tedy desetinné číslo) a vrací nejblížší číslo ze seznamu. Aktuálně implementované čísla jsou již vidělí v kapitole 2.3.4.

3.1.3 Funkce filter_theta_z()  

Další funkce je zde filter_theta_z(), zde se z vypočítaných záznamů odeberou nesmyslné výsledky větší než přednastavená konstanta. Aktuálně jsou v programu nastavené definové konstanty FILTER_LIMIT na 0.7 a SECOND_FILTER_LIMIT na 0.3.

3.1.4 Funkce cluster_together()  

Poté se v algoritmu použije funkce cluster_together(), zde se vyfiltrovaná a seřazená data shromáždí dohromady způsobem popsaným v algoritmu 6. Zjednodušené, funkce projde všemi dvojicemi dat a spočítá nejmenší rozdíl $\theta_z$ mezi těmito záznamy. Tyto 2 záznamy poté nahradí jedním, jehož hodnota jak $\theta_z$ tak i $err$ je jejich průměr. Toto se opakuje tak dlouho, dokud nejmenší rozdíl je větší než předem definovaný limit, v programu definovaný konstantou MAX_CLUSTER_DIFFERENCE. Tuto operaci provádíme, protože v této fázi máme s největší pravděpodobností až přibližně mnoho vypočítaných hodnot a potřebujeme jejich počet zredukovat. Po diskuzi s vedoucím práce je v tuto chvíli v programu nastavený limit 0.2. Celá tato metoda sluhování lze také v kódu jednoduše vyhnout za pomocí zakomentování definice pro preprocesor #define CLUSTER_ENABLE.
Algoritmus 6: ParaCell - Zeroshift - ClusterTogether

Input: data
// array of θ z and err, sorted by θ z
1  clusteredData ← {} 
2  i_saved ← {} 
3  o_saved ← {} 
4  difference_saved ← {} 
5  while TRUE do 
6    minDifference ← MAX 
7    for i ← 1, data do 
8      // search 2 closest entries 
9      for o ← i + 1, data do 
10     difference ← data.θ z [o] - data.θ z [i] 
11     if difference < minimalDifference then 
12      save difference i and o 
13     if minDifference > LIMIT then 
14      break 
15     replace the 2 saved θ z and err values with their means 

3.1.5 Funkce find_closest_range()

Poslední volaná funkce v algoritmu o které jsme se ještě nezmínili, je find_closest_range(), tato metoda má stejný účel jako cluster_together(), tedy zmenšuje počet hodnot, které budeme považovat za možné kandidáty na správnou hodnotu posunu. Tato funkce hledá všechny různé intervale hodnot θ z z předem zadaného rozsahu (v programu definovaném v konstantě MAX_RANGE_SIZE) a vrátí jejich průměrnou hodnotu θ z a err. Aktuální hodnota v programu je po diskusi s vedoucím práce 0.5. Tuto funkci shlukování lze stejně jako tu předchozí jednoduše v kódu vypnout za pomocí zakomentování řádku #define RANGE_ENABLE.

3.1.6 Výstup programu

Při běhu programu se postupně do příkazové řádky vypisují průběžné mezivýsledky a nakonec i finální výsledky. V programu je implementovaná i možnost uložit si n posunů s nejmenším err. Tuto hodnotu lze jednoduše přenastavit za pomocí konstanty pro preprocesor RESULT_PRINT_LIMIT.

3.2 Filtrování metod MGLS

V druhé části přidáváme do implementace algoritmu MGLS filtr, který vypíná vybrané metody hledání možných parametrů základní buňky. Tento filtr se nastavuje v konfiguračním souboru programu za pomocí nového paramtru FILTER_METHOD s celočíslnou hodnotou. Tato hodnota představuje bitovou masku, která podle přítomnosti jedničky či nuly zapne či vypne danou metodu. Ve výchozím nastavení je její hodnota 255.

Pro potřeby testování bylo dále potřeba přidat do programu způsob měření času. Na tento problém byla použita knihovna stl, chrono. Časovač byl vložen do funkce main() tak, aby obsahil pro nás nejvíce stěžejní část programu, tedy celé volání indexace MGLS.
Kapitola 4

Testování

V rámci této práce budeme ještě po částech testovat a hodnotit nově přidané funkcionality. Pro tyto testy byl vedoucím mé práce dodán odkaz pro získání testovacích dat na „American Mineralogist Crystal Structure Database“ [17], tím pádem všechna použitá data pocházejí z této internetové databáze, zároveň se také budou nacházet v digitální příloze této práce. Pro úplnost, ve všech testovacích souborech (tedy soubory končící na .dat) je na prvním řádku uvedeno jméno minerálu a odkaz na originální data. Použitá data nebyla nijak od původních záznamů měněna, případně byla pouze zkrácena až již z důvodu testovacích či z důvodu kompatibility s programem (ParaCell aktuálně podporuje indexaci s až 64 záznamy). Pro testování bylo použito více sestav (pro hardwarové parametry viz. tabulka 4.1), ale vzhledem k povaze testů, kde budeme porovnávat časové výsledky stejného krystalu spuštěného s různými metodami, bylo dodrženo použití stejného stroje pro všechny testy daného krystalu. Z tohoto důvodu také nebudeme u časových testů moc srovnávat přesné časové výsledky, ale budeme se soustředit na postupné pořadí od nejrychlejšího po nejpomalejšího.

<table>
<thead>
<tr>
<th>Sestava 1</th>
<th>OS</th>
<th>Windows 10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CPU</td>
<td>AMD Ryzen 7 5800H</td>
</tr>
<tr>
<td></td>
<td>Jádra/Logické procesory</td>
<td>8/16 (SMT)</td>
</tr>
<tr>
<td></td>
<td>GPU</td>
<td>Nvidia RTX 3070m</td>
</tr>
<tr>
<td></td>
<td>Velikost RAM</td>
<td>32GB</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sestava 2</th>
<th>OS</th>
<th>Ubuntu 20.04.4 LTS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CPU</td>
<td>Intel Xeon Gold 6254</td>
</tr>
<tr>
<td></td>
<td>Jádra/Logické procesory</td>
<td>18/36 (HyperThreading)</td>
</tr>
<tr>
<td></td>
<td>GPU</td>
<td>Nvidia Tesla V100 32GB</td>
</tr>
<tr>
<td></td>
<td>Velikost RAM</td>
<td>64GB</td>
</tr>
</tbody>
</table>

Tabulka 4.1 Tabulka sestav, použitých k testování
4.1 Optimalizace MGLS

Jako první byla testována optimalizace implementace algoritmu MGLS, protože v dalším testování opravy posunutí nuly budeme využívat právě tu nejlepší kombinaci metod MGLS zjištěnou na základě těchto testů.

Rámec tohoto testování se zajímá rozdíly různých implementací funckce, která v pseudokódu představuje funkci GetMeasuredTheta() a byly více probány v kapitole 2.3.3.4. V našem testování se budeme především zaměřovat na kvalitu výsledku a na rychlost indexace.

4.1.1 Ortorombické krystaly

Jako první budeme testovat nejjednodušší nativně implementovaný krystalický systém v algoritmu MGLS, a to ortorombický. Pro testování byly mým vedoucím práce zvoleny pouze metody 1, 2, 3 a 4, při čemž ve verzi, do které byla tato práce implementována, byly původně aktivní právě tyto 4 metody. Zbylé metody 5, 6, 7 a 8 byly v této verzi přítomny, ovšem jejich kód byl nedosažitelný. Další důležitou charakteristikou tohoto testu, kterou je nutno zmínit, je ze pro všechny použité testy bylo až na parametr výběru metody použito identické nastavení, především nastavení rozsahu hledání a počet kroků. Pro testování této skupiny dat byla použita sestava 1, znázorněná v tabulce 4.1.

Výsledky testování, které lze detailně prohlédnout v příloze A.1, byly shrnuty za pomoci součtu hodnot pořadí. Pro vysvětlení shrnutí si můžeme uvést krystal Orientite, který při použití metody 1 dosáhl hodnoty $F_{20}$ 84.4, metody 2 120.7, metody 3 0 a při použití metody 4 126.2. Pro tyto výsledky by shrnutí nabývalo hodnoty 3 pro metodu 1, 2 pro metodu 2, 4 pro metodu 3 a 1 pro metodu 4. Při nalezení stejné hodnoty se počítá pouze hodnota té lepší pro oba záznamy. Jak můžeme vidět v obrázku sloupcového grafu 4.2 především metoda 3 a metoda 2 nebyly ve vypočítání $F_{20}$ oproti ostatním metodám velmi úspěšné. Po podrobnější analýze dat můžeme o metodě 3 dokonce říci, že nebyla schopna nalézat žádný výsledek (ani jednou nenašla $F_{20}$ vyšší 6) a to i po téměř stejně době hledání parametrů jako metoda 1.

Pro praktický příklad se můžeme podívat na obrázek grafu grafu 4.1, kde jsou vyobrazeny i přesné časy a nalezené hodnoty pro krystal Forsterit. V tomto příkladu si můžeme povšimnout

---

1 Krystaly jednodušších krystalických systémů by byl algoritmus také schopný detekovat, ale pro testování je není důvod používat.
opět velice nízké hodnoty nalezené pomocí metody 3, ale dále i obrovské časové náročnosti metody 2, která i za cenu nejdélšího běhu nedosáhla nejlepšího výsledku. Dále při pohlednutí na výsledky metody 4 vidíme, že i když dosáhla nejlepších výsledků, čas hledání je téměř dvojnásobný oproti metodě 1.

A naše finální srovnání lze vidět na obrázku grafu [4.3]. Zde je vyobrazen poměr hodnoty F20 s časem indexace při čemž všechny záznamy s vyřešenými daty na maximální hodnotu F20 byly odebrány. Jak lze na první pohled vidět, metoda 1 naprosto překonala všechny ostatní metody. Po podrobnějším nahlédnutí do změněných dat opravdu můžeme konstatovat, že metoda 1 je nejlepší, téměř vždy měla v porovnání s ostatními metodami velmi přijatelný výsledek F20, a i když nebyl nejlepší, tak o to byla metoda 1 rychlejší.

Za zmínku stojí data krystalu Bridgmanite (viz. příloha A.1), pro který byly vytvořeny celkem 3 druhy testu, test se středním počtem kroků, s nižším a s vyšším. Pro test s vyšším a středním počtem kroků byly pro data nalezeny nejlepší parametry metodami 1, 2 a 4, ale metoda 3 nenalezla přijatelný výsledek ani v jednom případě i když čas indexace pro test s více kroky probíhal více než 4krát více než pro test se středním počtem kroků a více než 3.7krát delší čas než za který se povedlo metodě 1 celý krystal vyřešit.

### 4.1.2 Monoklinické krystaly

Pro monoklinické krystaly byla procedura testování prakticky stejná jako v minulém případě. Opět byla použita sestava 3 a detailní výsledky lze nalézt v příloze A.2.

Výsledky testování 16 krystalů této soustavy vyneslo podobné výsledky jako pro ortorombickou soustavu, ovšem mutno podotknout několik rozdílů. Metoda 1 byla stále nejlepší na nalezení rychlého výsledku, ale v hodnotě F20 nebyla nikdy nejlepší, buď měla nejlepší výsledek spolu s jinou metodou (toto nastávalo i když krystal ještě nebyl zcela vyřešen), nebo patřila na pozdější příčky. Jak lze ale vidět v obrázku grafu [4.4] stále dává za daný čas nejlepší výsledky. Metoda 4 byla opět na druhé příčce, tentokrát již tolik nezaostávala jako v případě ortorombické soustavy, stále se její využití časově nevyplatí, mnohokrát našla i lepší výsledek než metoda 1, ale při zvýšení počtu kroků nebo druhém, více omezeném vyhledávání by metoda 1 tyto výsledky zcela jistě našla také. Metoda 3 opět jako v předchozím případě nenašla žádný výsledek s hodnotou kvality F20 > 7, což znamená, že tyto výsledky, ani nemusí být správné a metoda je tím pádem
zcítíte výsledek. Překvapivě, metoda 2 je v tomto porovnání podobně neúspěšná jako metoda 3, to protože i když několikrát našla velmi dobrý výsledek, vždy za cenu velmi velkého času stráveného hledáním tohoto výsledku, někdy i více než 4krát delší čas než metoda 4.

4.1.3 Triklinické krytiny

Pro ten nejsložitější krytinický systém, tedy ten triklinicky, probíhá testování trochu jinak od předchozích testů. Tentokrát budeme testovat 5 různých metod (1,2,3,4 a 8) a to protože v době před implementací testovacích pomůcek do programu, byla pro tuto soustavu použita pouze metoda 8.

Pro tyto testy byla použita soustava 2 na školním clusteru „star.fit.cvut.cz“. Ale i přes velmi vysoký výkon této soustavy nebylo možno zaindexovat všechny krytiny, které byly v plánu. To především z důvodu obrovské časové náročnosti. Například indexace krytiny Inesita metodou 8 trvala téměř 5 hodin a dosáhla hodnoty 172 F20. Ostatní metody běžely více než 7.5 hodin a ani jedna nedosáhla nenulového F20, tento trend se poměrně často opakoval a pokusy o indexace trvaly někdy i více než den čas a dosáhly nenulového výsledku. Z toho důvodu si uvedeme pouze omezený počet krytalin.

Pro příklad si uvedeme triklinický krystal Microcline. Výsledky tohoto testu, které jsou vyobrazeny v obrázku grafu 4.5 jsou poměrně překvapivé. Metody 1,2 a 4 nikdy nenašly žádný výsledek, a to nejen v tomto jednom testu, ale v mnoha dalších provedených testech na tomto krystylu, které pro přehlednost budou vynechány z grafu. Další překvapení je metoda 3, pro kterou jsou v grafu uvedeny 2 různé testy jeden pro 20 kroků a druhý pro 25. Tato metoda se ukázala jako schopná najít řešení, ale zdaleka nedosahuje takové efektivity jako metoda 8, která při testování triklinických soustav se opakovaně zdálá být nejefektivnější nebo dokonce i jediná, která řešení vůbec nalezla.

4.1.4 Porovnání s ostatními algoritmy

Při porovnání s ostatními algoritmy implementovanými v programu na sestavě 2 bylo zjištěno několik zajímavých informací. Jak lze vidět v obrázku grafu 4.6 všechny testované algoritmy 6 bez

---

6Ostatní metody byly také naimplementovány, ale jejich kód byl narozen od metody 8 nedosažitelný.

5Viz. tabulka [4.1]

6Pro přehlednost zde byly zobrazeny pouze tyto 2 nejlepší metody.
zadaného specifického rozsahu objemu [7] nalezly dobré výsledky. Toto platí i pro metody MGLS, ale v porovnání s ostatními algoritmy nebyly tak ani tak přesné, ani tak rychlé. Zajímavý jev ovšem nastal při zadání přesnějšího intervalu objemu základní buňky, všechny algoritmy mimo MGLS s tímto neměly problémy, ovšem MGLS pro nižší počty kroků správné parametry jakoby přeskočil. Tento jev nejspíše nastával i při pokusech indexace triklinických krystalů v předchozí části, kdy z důvodu co většího urychlení indexace byly zadány co nejužší intervy parametrů a algoritmus i po velmi dlouhé době nic nenašel.

4.2 Zeroshift

Pro testování opravy chyby zeroshift byla použita data stejná jako pro test optimalizace MGLS a pro indexaci takové nastavení algoritmu MGLS, které jsme zjistili, že je pro indexaci nejefektivnější a nastavili ho tak, aby krystalovou soustavu kompletně vyřešil. Pro otestování schopnosti opravy chyby byla data uměle posunuty od $-0.1^\circ$ do $0.1^\circ$ po $0.005^\circ$ (tedy celkem 40) a do další série testů byl zanesen posun i s umělým řádem. Tato upravená data byla dále otestována 2 způsoby, praktickým vypočítáním hodnoty $F_{20}$ na opravených datech a porovnáním opravy s uměle uvedeným posunem bez indexace.

4.2.1 Praktické testy

Při praktických testech nebyly vypočítány všechny možné výsledky, byly vypočítány pouze ty, kde platí následující podmínka:

$$|\theta_z + artificialShift| \leq \left| artificialShift \cdot \frac{2}{3}\right|$$  \hspace{1cm} (4.1)

Tím pádem na reálných datech, tedy tam kde neznáme posun bylo potřeba výsledky buď vyfiltrovat nebo vypočítat všechny, což by ovšem u složitých soustav mohlo trvat velmi dlouho.

Pro příklad úspěšnosti opravy chyby si uveďme monoklinický krystal Uralborite. Pro tento krystal bylo pro 40 posunů (bez řádu) vypočítáno celkem 2702 různých oprav (to udává 67.5

\footnote{Objem byl nastavený na $\langle 30,1500 \rangle$}
Obrázek 4.6 Graf porovnávající indexační metody MGLS (značené method_steps), Dichotomy a Treor na datech monoklinického krystalu Deloryte při použití stejného omezení rozsahu. Metody označené písmenem V byly spuštěny s předem zadaným intervalem objemu základní buňky.

Tabulka 4.2 Tabulka výsledků testu opravy posunu nuly monoklinického krystalu Uralborite bez šumu. (Vyfiltrováno podle 4.1)

<table>
<thead>
<tr>
<th></th>
<th>Původní $F_{20}$</th>
<th>Průměrný opravený $F_{20}$</th>
<th>Nejhorší opravený $F_{20}$</th>
<th>Nejlepší opravený $F_{20}$</th>
<th>Průměrná hodnota $-\text{ooprava/posun}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uralborite</td>
<td>205.7</td>
<td>101.0</td>
<td>55.6</td>
<td>229.7</td>
<td>0.999</td>
</tr>
</tbody>
</table>

Tabulka 4.3 Tabulka výsledků testu opravy posunu nuly pro ortorombický krystal Cubanite bez šumu. (Vyfiltrováno podle 4.1)

<table>
<thead>
<tr>
<th></th>
<th>Původní $F_{20}$</th>
<th>Průměrný opravený $F_{20}$</th>
<th>Nejhorší opravený $F_{20}$</th>
<th>Nejlepší opravený $F_{20}$</th>
<th>Průměrná hodnota $-\text{ooprava/posu}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cubanite</td>
<td>138.8</td>
<td>88.2</td>
<td>33.3</td>
<td>140.3</td>
<td>0.988</td>
</tr>
</tbody>
</table>
oprac pro každý posun), z toho jsme pro ušetření času vyfiltrovali vzorcem 4.1 možné opravy, které byly až příliš daleko od uměle nasazeného posunu. Výsledky tohoto testu jsou vyobrazeny v tabulce 4.2 ve které můžeme vidět velmi dobré výsledky, ve kterých lze vidět, že se dokonce podařilo opravit posun nuly obsažený i v samotných původních datech. Za tyto dobré výsledky můžeme určit poděkovat již zmíněnému filtrování, bez něj by průměrná hodnota a minimální hodnota byla určitá daleko. Ale jak si ještě ukážeme v další kapitole, v tuto chvíli není znám žádný způsob jak určit, které opravy jsou nejpravděpodobnější.

Jako další příklad si můžeme uvést ortorombický krystal Cubanite. Pro tento krystal bylo vypočítáno 2317 záznamů ty byly vyfiltrované na 235. Obecné výsledky, které jsou zobrazeny v tabulce 4.3 nám ukazují, že výsledky tohoto testu jsou podobně dobré jako u předchozího krystalu. Ale stále trpí již zmíněným problémem s počtem výsledků.

Jako poslední krystal na tyto testy si uvedeme triklinický Microcline, jehož výsledky jsou uvedeny v obrázku grafu 4.7. V něm lze spatřit linii rozdílů mezi nejlepší nalezenou opravou a umělým posunem. Tyto hodnoty představují skutečnou chybu posunutí nuly, kterou data krystalu obsahovala ještě před uvedením umělého posunu. Dále si lze všimnout „schodů“ v grafu průměrné hodnoty $F20$, ty jsou způsobeny filtrací dat.

**Tabulka 4.4** Tabulka výsledků použití opravy posunutí nuly na data s uměle přidaným šumem.
Testování

Test se šumem byl proveden pro posuny $-0.005$, $0.0$ a $0.005$, kde šum nabýval náhodných hodnot pro každý prvek z intervalu hodnot $-0.01$ až $0.01$. Takto upravená data jsme dále stejně otestovali jako v předchozích testech. Výsledky těchto testů, které jsou k vidění v tabulce 4.4, nám říkají, že průměrně nejsou dobré, často díky opravě vychází i nulové nebo velmi nízké $F$ a $L$, ale i tak lze díky opravě nalézt výsledky, které hodnotu zvýší. Obecně se tedy použití algoritmu na opravu posunu vyplatí i na tyto data, koneckonců sami nevíme jestli nějaký šum nebyl již v datech na kterých jsme testovali posun bez šumu.

4.2.2 Teoretická oprava chyby

Další metoda testování této opravy byla provedena bez opravdové indexace, v tomto případě spoluříkame, že v samotných testovacích datech se tato chyba nachází co nejméně a algoritmus nám najde co nejbližší opravu na námí umělé dodaným posunu. Hodnocení úspěšnosti poté měříme stejným zlomkem jako v tabulce 4.4 tedy $-oprava/posun$. Tím pádem čím blíže se tato hodnota nachází k $100\%$ tím blíže se algoritmus trefil k umělému posunu.

\begin{figure}
\centering
\includegraphics[width=\textwidth]{graf.png}
\caption{Obrázek 4.8 Graf poměru opravy zeroshift chyby s grafem průměrného počtu prvků v difrakčním záznamu.}
\end{figure}

Porovnání různých krystalických systémů lze vidět na obrázku grafu 4.8, kde je vyobrazen již zmíněný poměr nejlepší odhadnutí chyby, průměrné počtydifrakčních záznamů pro krystaly z daného systému a poměry nejlepších odhadů chyby podle parametru $err$ změněný v algoritmu 5. Tento parametr bohužel úplně nenaplňuje očekávání a jeho přesnost je velmi nízká. Bylo otestováno jestli jeho úspěšnost při vypočítávání kubického systému má nějaké dočinění s nízkým počtem zaznamenaných difrakčních záznamů, ale bohužel nikoliv. Zdá se, že tento výpočetní parametr je úspěšný pouze pro jednodušší krystalické soustavy.

Při podrobnějším prohlédnutí těchto statistik si lze všimnout u většiny krystalů malého rozdílu nejlepší opravy zeroshift a námi provedeným umělým posunem, to je opět oprava této.

\footnote{Které jsou dostupné v elektronické příloze této práce.}
chyby v původních datech. Nutno ale podotknout, u tohoto testu nastává jedna nevýhoda a to zejména u výpočtu kvality malých posunů. Při přesnějších datech se nejedná o nic hrozného, ale čím více se posun nachází v originálních datech, tím více tyto hodnoty jsou opravené mimo původní posun a tím je i výsledek horší. Dále lze z dat zjistit, že na existenci nejlepší možné opravy nemá počet prvků žádný vliv, sem tam se vyskytne nějaká anomálie, ale za to nejspíše mohou specifická data krystalu.

4.2.3 Porovnání shlukovacích funkcí

Jak jsme si již řekli v praktické části, v algoritmu jsou naimplementovány 2 funkce určené pro zredukování možných kandidátů shlukování. Každá z těchto 2 funkcí má v kódu svůj vlastní parametr pro nastavení míry shlukování a také svůj vlastní „vypínač“, díky kterému lze zako-menováním nebo odkomentováním jednoho řádku funkci vypnout či zapnout. Pro zjištění, jaká funkce je lépeš, byl proveden test, jehož výsledky jsou v obrázku 4.10. V obrázku si můžeme všimnout, že nastavení parametrů těchto shlukovacích funkcí se skutečně do výsledků promítne. Při výběru našich parametrů si nejlépe vedla funkce RANGE s nastavením na 0.2. Tím dosáhla velmi dobrých výsledků, které se v průměru rovnali hodnotě 0.9992. Oproti tomu funkce CLUSTER si v ukázkách v grafu nevedla tak dobře, ale v testech se ukázalo, že ji její nejlepším nastavení, které leží okolo hodnoty 0.01, dokáže být téměř stejně přesná jako metoda RANGE. Problém ale poté nastává v množství kandidátů, protože při takovém velkém množství je prostě zaručené, že se nějaký kandidát do námí uvedeného posunu „trefí“. A z toho důvodu je výběr funkce a nastavení parametrů shlukování či filtrování při použití této metody zásadní.

4.2.4 Porovnání indexace opravených a neopravených dat

V poslední krátké části testování se zajímáme o porovnání nově naimplementované metody indexace s dalšími již naimplementovanými metodami, tedy porovnáváme data s posunem nuly s opravou a bez. Jak lze vidět v obrázku grafu 4.11 při použití námi vyfiltrovaných dat se chybu

---

9 které lze vidět v obrázku grafu 4.11
Obrázek 4.10 Graf úspěšnosti nejlepšího nalezeného opravného posunu za použití různých shlukovacích metod s různými parametry. Vypočítáno pro ortorombický krystal Berthierite.

Obrázek 4.11 Graf počtu nalezených kandidátů při použití různých shlukovacích metod s různými parametry a za použití obou filtrů s hodnotou 0.7. Vypočítáno pro ortorombický krystal Berthierite.
podařilo pro každý posun opravit. V praxi sice nebudeme indexovat 40 různých posunů jako v případě testování, ale nebudeme mít možnost tyto data vyfiltrat tak, aby zahrnovala pouze posun když ho nebudeme znát. Jeden z teoreticky možných postupů by v takovém případě mohlo být rozdělení všech nalezených oprav do několika skupin a postupovat v hledání správné opravy jako v binárním stromě (případně jako v (a,b) stromě). Tuto teorii i podporují naše výsledky zobrazené v již zmíněném obrázku [12] kde můžeme pozorovat možné hodnoty F20 při zkoušení různých posunů na černém grafu.

**Obrázek 4.12** Graf hodnot F20 krystalu Cubanite s uměle vytvořeným posunutím nuly pro indexaci s nejlepší opravou, průměrnou opravou, nejhorší opravou a žádnou opravou při vyfiltrování dat za pomocí vzorce [1,1].
Kapitola 5
Závěr

V této práci jsme se seznámlí se základy strukturní krystalografie a principem difrakcí krystalů, dále jsme se zabývali metodami k určení kvality vypočítaných parametrů krystalu spolu s různými indexačními algoritmy, které jsou nainstalované v balíčku ParaCell a nakonec problematikou opravy chyby posunutí nuly. V následující části jsme se zabývali konkrétní implementací a návrhem nové metody pro opravu chyby zero-shift a úpravou již existujícího algoritmu v programu za účelem jeho optimalizace.

Tyto dvě rozšíření jsme poté v další kapitole otestovali na dodaných datech od vedoucího práce a vyhodnotili. Výsledky ukázaly silné i slabé stránky metod algoritmu MGLS, kde jsme určili nejefektivnější metody pro indexaci nainstalovaných krystalických soustav. V porovnání s ostatními nainstalovanými algoritmy si algoritmus MGLS nevedl velmi dobře, ale stále se nachází v experimentální fázi.

Poté jsme testováním metody pro optimalizaci záznamů objevili vysokou úspěšnost pro hledání posunu v datech, i když z důvodu velkého množství kandidátů pro tyto posuny se metoda bez správného filtrování kandidátů může vyznačovat vysokou výpočetní náročnost. Dále jsme testováním různých parametrů algoritmu zkoumali promítání těchto změn v datech a určili jsme jejich přibližně nejlepší parametry pro nejprv pravděpodobnější vypočítání chyby. Na závěr jsme porovnali na posunutých datech opravy vytvořených kandidátů s výsledky bez opravy, kde jsme zjistili, že pro všechny posuny existuje v množině kandidátů nejlepší možné řešení, nakonec jsme naznačili metodu, díky které bychom měli být schopni nalézt v množině nejlepší možný výsledek.
Závěr
Příloha A

Přílohy

■ Výpis kódu A.1 Část kódu funkce zero_shift_dong() pro výběr hlavní výpočetní metody. Kód zkrácen pro přehlednost.

```cpp
bool zero_shift_dong(struct measured* mea,
                     struct database* DB,
                     struct configuration& cfg) {

    std::vector<theta_z_result> calculated_th_z(0);
    std::vector<double> fractions =
        {1, 1.2, 1.25, 1.33333, 1.5, 1.66667, 2, 2.5, 3, 4, 5, 6};

    for (int i = 0; i < mea->ndat; ++i) {
        for (int o = i + 1; o < mea->ndat; ++o) {

            double theta1 = mea->th_center[i];
            double theta2 = mea->th_center[o];

            double q1 = mea->q_center[i];
            double q2 = mea->q_center[o];

            if (theta1 < 10) {
                // angle too small
                continue;
            }

            double ratio = sqrt(q2 / q1);
            double m = get_closest(fractions, ratio);

            double err = abs(q2 - q1 * m * m);

            calculated_th_z.push_back({
                zero_shift_calculate(theta1, theta2, m, ratio),
                m,
            });
        }
    }

    return true;
}
```

err});
}

/* Filters theta_z results */
calculated_th_z =
    filter_theta_z(calculated_th_z, FILTER_LIMIT);

/* Sorts vectors by their theta_z value */
std::sort(calculated th_z.begin(), calculated th_z.end(),
    [](theta_z_result a, theta_z_result b) {
        return a.theta_z < b.theta_z;
    });

std::vector<theta_z_result> theta_z_candidates =
    find_closest_range(calculated_th_z, MAX_RANGE_SIZE);

std::vector<theta_z_result> theta_z_candidates_cluster =
    cluster_together(calculated_th_z, MAX_CLUSTER_DIFFERENCE);

theta_z_candidates.insert(
    theta_z_candidates.end(),
    theta_z_candidates_cluster.begin(),
    theta_z_candidates_cluster.end());

if (theta_z_candidates.size() == 0) {
    return false;
}

/* Sort candidates by err */
std::sort(theta_z_candidates.begin(),
    theta_z_candidates.end(),
    [](theta_z_result a, theta_z_result b) {
        return a.err != b.err ?
            abs(a.err) < abs(b.err) :
            a.theta_z < b.theta_z;
    });

... // Prints results

return true;
<table>
<thead>
<tr>
<th>Krystal</th>
<th>Metoda</th>
<th>F20</th>
<th>Uplynulý čas</th>
</tr>
</thead>
<tbody>
<tr>
<td>Orientite</td>
<td>1</td>
<td>84.4</td>
<td>6600262</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>120.7</td>
<td>67951500</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0</td>
<td>5998218</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>126.2</td>
<td>10787017</td>
</tr>
<tr>
<td>Dorfmanite</td>
<td>1</td>
<td>101.3</td>
<td>6770774</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>109.9</td>
<td>91817897</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0</td>
<td>6104789</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>128.3</td>
<td>11706939</td>
</tr>
<tr>
<td>Graphite</td>
<td>1</td>
<td>89.7</td>
<td>801890</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>84.0</td>
<td>1407330</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1.7</td>
<td>632502</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>93.8</td>
<td>1730218</td>
</tr>
<tr>
<td>Bridgmanite</td>
<td>1</td>
<td>319.6</td>
<td>6122643</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>319.6</td>
<td>104314001</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>3.4</td>
<td>5249064</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>319.6</td>
<td>13019180</td>
</tr>
<tr>
<td>Bridgmanite</td>
<td>1</td>
<td>319.6</td>
<td>437891664</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>319.6</td>
<td>450921281</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>4.0</td>
<td>22803884</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>319.6</td>
<td>61816093</td>
</tr>
<tr>
<td>Bridgmanite</td>
<td>1</td>
<td>295.5</td>
<td>439063</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>256.5</td>
<td>4744830</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>2.8</td>
<td>413120</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>315.6</td>
<td>782356</td>
</tr>
<tr>
<td>Yuanfuliite</td>
<td>1</td>
<td>148.6</td>
<td>3084679</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>124.4</td>
<td>20389002</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>3.7</td>
<td>2793031</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>182.3</td>
<td>5783620</td>
</tr>
<tr>
<td>Zincolivenite</td>
<td>1</td>
<td>178.3</td>
<td>6532525</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>262.4</td>
<td>96513433</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>5.2</td>
<td>6000622</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>289.1</td>
<td>11834362</td>
</tr>
<tr>
<td>Marcasite</td>
<td>1</td>
<td>205.2</td>
<td>2821438</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>212.8</td>
<td>17732033</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1.9</td>
<td>2131869</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>212.8</td>
<td>6012338</td>
</tr>
<tr>
<td>BrN2C4O4</td>
<td>1</td>
<td>318.7</td>
<td>6573637</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>360.0</td>
<td>77956564</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.0</td>
<td>6084630</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>356.0</td>
<td>10972611</td>
</tr>
<tr>
<td>Montroseite</td>
<td>1</td>
<td>297.3</td>
<td>5803912</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>296.3</td>
<td>66418727</td>
</tr>
<tr>
<td>Tabulka A.1 Tabulka výsledků testování metod MGLS pro ortorombické krystaly</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------</td>
<td>------------------</td>
<td></td>
</tr>
<tr>
<td><strong>Krystal</strong></td>
<td><strong>Metoda</strong></td>
<td><strong>F20</strong></td>
<td><strong>Uplýnulý čas</strong></td>
</tr>
<tr>
<td>Arrojadite,KFe</td>
<td>1</td>
<td>27.2</td>
<td>7669282</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>10.2</td>
<td>71170342</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>6.5</td>
<td>5552757</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>36.6</td>
<td>24581611</td>
</tr>
<tr>
<td>Co3P2O8</td>
<td>1</td>
<td>167.1</td>
<td>4030283</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>6.4</td>
<td>66044577</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>3.8</td>
<td>3395157</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>167.1</td>
<td>13538615</td>
</tr>
<tr>
<td>Gorgeyite</td>
<td>1</td>
<td>24.2</td>
<td>1245313</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>7.0</td>
<td>16222022</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>5.0</td>
<td>1013374</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>24.2</td>
<td>3698074</td>
</tr>
<tr>
<td>Nickelboussingaultite</td>
<td>1</td>
<td>31.1</td>
<td>3117175</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>31.1</td>
<td>32495302</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.0</td>
<td>2786145</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>283.4</td>
<td>9781969</td>
</tr>
<tr>
<td>NiTi</td>
<td>1</td>
<td>141.9</td>
<td>4074335</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>12.3</td>
<td>46849045</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>2.4</td>
<td>2787256</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>------------------</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
</tr>
<tr>
<td><strong>O_Ti</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>45.8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>4.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>2.4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>61.6</td>
<td></td>
</tr>
<tr>
<td><strong>Reinhardbraunsite</strong></td>
<td>1</td>
<td>171.2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>46.8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>4.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>347.6</td>
<td></td>
</tr>
<tr>
<td><strong>Studenitsite</strong></td>
<td>1</td>
<td>113.6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>67.4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>113.6</td>
<td></td>
</tr>
<tr>
<td><strong>Thalenite_Y</strong></td>
<td>1</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>229.2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>113.6</td>
<td></td>
</tr>
<tr>
<td><strong>Ti50.75Ni47.75Fe1.50</strong></td>
<td>1</td>
<td>136.1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>136.1</td>
<td></td>
</tr>
<tr>
<td><strong>Uralborite</strong></td>
<td>1</td>
<td>6.2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>70.6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>169.8</td>
<td></td>
</tr>
<tr>
<td><strong>Uranophane_alpha</strong></td>
<td>1</td>
<td>40.2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>23.7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>4.4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>142.2</td>
<td></td>
</tr>
<tr>
<td><strong>Zwieselite</strong></td>
<td>1</td>
<td>65.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>5.9</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>3.2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>65.5</td>
<td></td>
</tr>
</tbody>
</table>

*Tabulka A.2* Tabulka výsledků testování metod MGLS pro monoklinické krystaly


Obsah přiloženého média

readme.txt................................. stručný popis obsahu média
exe................................. adresář se spustitelnou formou implementace
data
data.zip.......................... archiv ve formátu ZIP obsahující testovací data a jejich výsledky
src
impl........................................... zdrojové kódy balíku ParaCell
thesis..................................... zdrojová forma této práce ve formátu \LaTeX
thesis.pdf.............................. text práce ve formátu PDF