ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

FAKULTA BIOMEDICÍNSKÉHO INŽENÝRSTVÍ

Katedra biomedicínské informatiky

Webová aplikace pro detekci somatických bodových variant u pacientů s leukémii

A web application for detection of somatic point mutations in patients with leukemia

Diplomová práce

Studijní program:	Biomedicínská a klinická informatika
Specializace	Softwarové technologie
Vedoucí práce:	Ing. Ondřej Klempíř, Ph.D.
Konzultantka:	Mgr. Pavla Suchánková

Jméno Příjmení Bc. Matěj Koudelka

Kladno 2023

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení:	**Koudelka**	Jméno:	**Matěj**	Osobní číslo:	**474368**
Fakulta:	**Fakulta biomedicínského inženýrství**				
Garantující katedra:	**Katedra biomedicínské informatiky**				
Studijní program:	**Biomedicínská a klinická informatika**				

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Webová aplikace pro detekci somatických bodových variant u pacientů s leukémii

Název diplomové práce anglicky:

A web application for detection of somatic point mutations in patients with leukemia

Pokyny pro vypracování:

Včasná diagnostika mutací v genech frekventovaně mutovaných u hematologických onemocnění může mít vliv na léčbu i následnou prognózu pacientů. V klinické praxi se pro detekci variant využívají individuální prahové hodnoty, získané statistickou analýzou dat kontrolní skupiny. Cílem projektu je návrh a vývoj webové aplikace pro stanovení individuální prahů při detekci bodových variant. Dílčí úkoly: • Seznámení se se základními formáty dat využívaných v sekvenování nové generace (NGS). • Studium problematiky genetické podstaty leukémii prostřednictvím metod NGS. - Implementace statistické metody pro stanovení individuálních prahů pro detekci bodových variant. • Návrh a implementace webové aplikace pro stanovení individuálních prahů včetně jednoduchého grafického rozhraní. • Testování na simulovaných a reálných datech.

Seznam doporučené literatury:

[1] Indra Neil Sarkar, Methods in Biomedical Informatics - A Pragmatic Approach, ed. 1. ed., Elsevier Inc., 2014, ISBN 978-0-12-401678-1
[2] Johns Hopkins University, Exploratory Data Analysis, 10 Oct 2016, https://www.coursera.org/learn/exploratory-data-analysis/
[3] Pallets Developer Community, Python Flask Web Development, 4/2020, https://flask.palletsprojects.com/en/2.2.x/tutorial/

Jméno a příjmení vedoucí(ho) diplomové práce:

Ing. Ondřej Klempíř, Ph.D.

Jméno a příjmení konzultanta(ky) diplomové práce:

Mgr. Pavla Suchánková

Datum zadání diplomové práce: **14.02.2023**
Platnost zadání diplomové práce: **20.09.2024**

doc. Ing. Zoltán Szabó Ph.D.
vedoucí katedry

prof. MUDr. Jozef Rosina, Ph.D., MBA
děkan

PROHLÁŠENÍ

Prohlašuji, že jsem diplomovou práci s názvem Webová aplikace pro detekci somatických bodových variant u pacientů s leukémii vypracoval samostatně a použil k tomu úplný výčet citací použitých pramenů, které uvádím v seznamu přiloženém k diplomové práci.

Nemám závažný důvod proti užití tohoto školního díla ve smyslu § 60 Zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů.

V Kladně dne 18. 5. 2023 …..……..……...

Bc. Matěj Koudelka

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu práce Ing. Ondřeji Klempířovi, Ph.D., za uvedené připomínky, cenné rady, trpělivost a celkový dohled nad průběhem této práce. Dále bych rád poděkoval zadavatelce a konzultantce Mgr. Pavle Suchánkové, za rady a obeznámení s problematikou daného tématu. Také bych rád poděkoval celé rodině a přítelkyni za neochvěnou podporu při studiu a při realizaci diplomové práce.

ABSTRAKT

Název práce: Webová aplikace pro detekci somatických bodových variant u pacientů s leukémii

Diplomová práce zkoumá RNA sekvence izolovaných leukocytů u pacientů s leukémií v genu kinázové domény fúzního genu BCR::ABL1. Zkoumá hlavně chronickou myeloidní leukémii a PH pozitivní akutní lymfoblastickou leukémii způsobenou filadelfským chromozomem. Včasné objevení leukémie vede k větší šanci na úspěšnou léčbu a prodlužuje přežití pacientů s tímto onemocněním.

Dále se zabývá studiem základních znalostí nutných k pochopení práce, a pokračuje návrhem a implementací webové aplikace, která analyzuje data pacientů s leukémií, a to jak po teoretické, tak i praktické stránce.

Výsledná aplikace byla vytvořena pomocí webových technologií HTML, CSS, Javascriptu a programovacího jazyku Python, který vytvořil RESTové API pro výpočetní vrstvu aplikace. Bylo zjištěno, že mutace na některých pozicích genu jsou při tomto druhu onemocnění častější, a proto výpočet prahových hodnot probíhá na základě normalizace dat a následné kontrole pozic zdravých dárců oproti nemocným. Tato kontrola používá Poissonovo rozdělení pro detekci málo častých jevů a následně kvantifikuje tato data. Dále probíhá analýza, která zkoumá, na jaké pozici mohly vzniknout mutace či záměny aminokyselin a tyto pozice označí.

Aplikaci se podařilo v plné míře implementovat, a hlavně zjednodušit její nasazení na všechny platformy.

Klíčová slova

NextDOM, webová aplikace, leukémie, detekce, analýza, BCR::ABL1, fúzní gen, filadelfský chromozom, DNA, sekvenování, kinázová doména

ABSTRACT

The title of the Thesis: A web application for detection of somatic point mutations in patients with leukemia

The thesis investigates RNA sequences of isolated leukocytes from leukemia patients in the kinase domain gene of the BCR::ABL1 fusion gene. It mainly investigates chronic myeloid leukemia and PH positive acute lymphoblastic leukemia caused by Philadelphia chromosome. Early detection of leukaemia leads to a better chance of successful treatment and prolongs survival rate of patients with this type of disease.

Thesis also discusses the knowledge base required to understand the work, and continues with the design and implementation of a web application that analyzes data from leukemia patients, both theoretically and practically.

The final application was created using web technologies HTML, CSS, Javascript and the Python programming language to create a REST API for the application's processing layer. It was found that mutations at certain positions of the gene are more common in this type of disease, and therefore thresholds were calculated by normalizing the data and then checks the positions of the healthy donors against the diseased ones. This check uses a Poisson distribution to detect rare occurrences and then quantifies the data. Furthermore, an analysis is performed to examine at which position mutations or amino acid substitutions may have occurred and mark these positions.

The application has been fully implemented and more importantly, simplified for deployment on all platforms.

Keywords

NextDOM, web application, leukaemia, detection, Analysis, BCR::ABL1, fusion gen, Philadelphia chromosome, DNA, sequencing, kinase domain

Obsah

Seznam symbolů a zkratek

Seznam zkratek

Zkratka	Význam
CML	Chronická myeloidní leukémie
ALL	Akutní lymfatická leukémie
Ph+ ALL	Ph pozitivní akutní lymfoblastická leukémie
CSV	Strukturovaný textový formát dat
XLSX	Excelovský sešit (soubor uložen v MS: Excel)
LoD	Limits of Detection
LoB	Limits of Blank
LoQ	Limits of Quantification
GIT	Verzovací systém
HTML	Jazyk webových stránek
CSS	Kaskádové styly (design webových stránek)
ÚHKT	Ústav hematologie a krevní transfuze
OOP	Objektově orientované programování
Open-source	Otevřený zdrojový kód
SNP	Jednonukleotidové polymorfismy
NGS	Next Generation Sequencing (Sekvenování nové generace)

Seznam tabulek

Seznam obrázků

1 Úvod

NextDOM 2.0 algoritmus je využíván pro detekci somatických bodových variant u pacientů s leukémií v genu kinázové domény fúzního genu BCR::ABL1 a zkoumá odolnosti těchto genů na léčbu. Tento postup řešení se zaměřuje na chronickou myeloidní leukémií (dále jen CML) a Ph pozitivní akutní lymfoblastická leukémie (dále jen Ph+ ALL). CML je typ rakoviny, která postihuje bíle krvinky (leukocyty) a má velmi pomalý postup, který se pohybuje v řádech několika let. Ph+ ALL je také typem rakoviny napadající bílé krvinky, která se projevuje především u lidí nad 50 let a má velmi nepříznivou prognózu. Oba tyto typy souvisí s charakteristickou translokací, kterou nazýváme filadelfský chromozom. Lidé s tímto chromozomem produkují onkoprotein BCR::ABL1. Bylo zjištěno, že vznik somatických mutací v kinázové doméně tohoto proteinu vede k rezistenci na léčbu. Z tohoto důvodu je velmi důležitá včasná a správná diagnostika těchto mutací pro zvýšení šance na lepší výsledky a účinnější léčbu u pacientů. Jiné druhy mutací jsou samozřejmě pro pacienty také důležité, ale nemají vliv na léčbu CML či Ph+ ALL. [1, 2]

Tento algoritmus používá Ústav hematologie a krevní transfuze (dále jen ÚHKT). Na vývoji algoritmu NextDOM 2.0 se podílela také 1. lékařská fakulta Univerzity Karlovy, Fakulta dopravní Českého vysokého učení technického v Praze (dále jen ČVUT) a Fakulta biomedicínského inženýrství ČVUT.

Hlavním důvodem vzniku této diplomové práce je zpřístupnění aplikace, kterou v současné chvíli využívá k diagnostikování a monitoringu somatických mutací oddělení molekulární genetiky na ÚHKT. Zároveň zvýšit také její uživatelskou přístupnost, která je v současné době nedostačující. Hlavním nedostatkem nynějšího řešení je menší variabilita, nemožnost včasných aktualizací a neschopnost provozovat a renderovat aplikaci na jiném operačním systému, než je samotný operační systém, na kterém je aplikace spuštěna.

Požadavek na tuto diplomovou práci přišel od Mgr. Pavly Suchánkové, která pracuje současně v ÚKHT i na FBMI a podílí se na této diplomové práci jako konzultantka.

Primárním cílem této diplomové práce je vytvořit webovou aplikaci, která nebude vyžadovat instalaci žádného dodatečného softwaru na stanici uživatelů a bude napsaná v programovacím jazyce, který není zatížen žádnými licenčními poplatky. Tímto se aplikace stane rozšířenější, variabilnější a pro většinu uživatelů dostupnější a udržitelnější pro její provozovatele.

2 Přehled současného stavu

Tato kapitola pojednává o jednotlivých částech teorie, které jsou potřeba pro uchopení základního konceptu této práce. Jedná se tedy o základy bioinformatiky, statistickém zpracování genomických dat, základech genetiky a základní informace o CML a Ph+ ALL. Další část kapitoly se věnuje současnému stavu aplikace, se kterou pracuje zadavatel této práce a popisuje její funkce a základní manipulaci s aplikací.

2.1 Základy genetiky

Tato podkapitola čtenáře stručně uvede do základů genetiky, seznámí je s jejím stručným vývojem, a především se základními pojmy.

Genetika je jednou z biologických věd, která se zabývá dědičností a proměnlivostí organismů. Genetika vysvětluje pravidla, jimiž se řídí přenos dědičných znaků z rodičovské generace na potomstvo. Předmětem jejího zkoumání je tedy podobnost rodičů a jejich potomků. Dědičnost neboli heredita je chápána jako schopnost organismu reagovat daným způsobem na určité podmínky vnějšího prostředí a na jejich změny. Jedním z klíčových procesů je přenos určitých metabolických procesů, které se nacházejí uvnitř buňky a jsou předávány z generace na generaci. Na tento přenos má určitý vliv i prostředí. Dědičnost je tedy schopnost organismů vytvářet potomky se stejnými či podobnými znaky, což zajišťuje zachování různých biologických druhů v průběhu mnoha generací. Proměnlivost neboli variabilita, pak zajišťuje odlišnost mezi jedinci jednoho druhu. Tyto proměnlivé projevy se rozšiřují působením faktorů vnějšího prostředí. [3, 4]

2.1.1 Základní pojmy genetiky

Nukleotid – Základní stavební blok nukleových kyselin, jimiž jsou DNA a RNA.

Nukleové kyseliny – Nukleové kyseliny, polynukleotidy, jsou tvořeny dlouhými řetězci (mono)nukleotidů, které jsou propojeny fosfodiesterovými vazbami. Monomerem nukleových kyselin jsou nukleotidy, které se skládají z heterocyklické dusíkaté báze. Pro DNA je to Adenin, Guanin, Cytozin a Thymin a pro RNA je to Adenin, Guanin, Cytosin a Uracil, dále z pentózy (ribóza či deoxyribóza) a kyseliny fosforečné.

Aminokyseliny – Základní stavební složka bílkovin.

Gen – Základní jednotka dědičnosti, která nese informaci o určité vlastnosti organismu. Podrobněji je rozebrán níže.

Exon – Kódující část genu.

Intron – Nekódující část genu.

Alela – Jedna z variant genu. Každý jedinec má dvě alely pro každý gen. Jeden od matky, druhý od otce.

Genom – Celý soubor genetické informace. Obsahuje kódující i nekódující oblasti.

Chromozom – Struktury obsažené v buněčném jádře, které nesou geny. Lidské chromozomy mají obvykle 23 párů, přičemž je 22 párů označeno jako autosomy a jeden pár je pohlavní neboli gonozomální (XX u žen a XY u můžu).

Genotyp – Genetická informace jedince. Může zahrnovat viditelné i neviditelné genetické vlastnosti.

Fenotyp – Jedná se o pozorovatelný projev genetické informace. Zahrnuje všechny viditelné vlastnosti organismu.

Karyotyp – Úplný popis sady chromozomů v jádře buňky.

Dominantní a recesivní alely – Rozdíl mezi dominantní a recesivní alelou je, že dominantní alela se projevuje ve fenotypu, i když je přítomna jen jedna alela. Recesivní alela se projevuje pouze pokud je v páru.

Dělení buněk – Dělení buněk může probíhat třemi způsoby.

- Mitóza – Dochází k replikaci a rovnoměrnému rozdělení genetického materiálu.
- Meióza – Způsob pohlavního buněčného dělení. Při každém dělené vzniknou 4 buňky. Dochází ke crossing-overu.
- Amitóza – Přímé dělení bez vytváření chromozomů a dělícího vřeténka. Většinou se jedná o degenerující nebo nádorové buňky.

2.1.2 Historie

První, kdo přišel s poznatky o genetice byl Johann Gregor Mendel v roce 1865, ten pracoval s existencí genů. Mendel vysvětlil základní zákonitosti genetiky. Mimo jiné zjistil, že se nedělí přímo znaky, ale vlohy pro ně. Výsledky jeho práce jsou známé jakožto Mendelovy zákony, které jsou celkem tři:

- zákon o stejnorodosti první generace kříženců,
- zákon o nestejnorodosti druhé generace kříženců,
- zákon o kombinovatelnosti vloh.

Dalším průkopníkem byl William Bateson, který se jeho pokusy snažil zpopularizovat. Ten také přinesl slovo genetika, které se pro tuto vědu uchytilo a používáme jej dodnes.

Kolem roku 1919 začal Thomas Hung Morgan s pokusy na octomilkách, na nichž studoval chromozomy. Morganovy pokusy přinesly spoustu nových poznatků a informací o genech a genové vazbě.

Dalším významným průlomem pak bylo rozluštění struktury molekuly DNA, k němuž došlo v roce 1953. O tento objev se postarali James D. Watson a Francis H. Crick.

Je vhodné také zmínit pojem eugenika. Jedná se o obor, který se zajímá o ovlivňování genetického kmene lidstva ve snaze působit na faktory genetické i ostatní, jenž by mohly ovlivnit či změnit fyzický a duševní vývoj budoucích generací. Předchůdcem dnešní eugeniky byly ideje Francise Galtona, který chtěl zlepšit genetické složení lidské populace selekcí jedinců, potenciálních rodičů. Ta byla zaměřena na zlepšení vlastností potomků i k vyloučení anomálií. Tyto myšlenky pak byly zneužity, konkrétně ideologií nacistického Německa, kdy nacisté usilovali o vyhlazení určitých fenotypů lidí a jiné naopak vyzdvihovali.

2.1.3 DNA (kyselina deoxyribonukleová)

Jedná se biopolymer a univerzální látkový nosič genetických informací. Skládá se z řetězce nukleotidů, které obsahují 4 báze jmenovitě Adenin-A, Cytosin-C, Guanin-G, Thymin-T. V pořadí bází, tedy deoxyribonukleotidů, je zakódovaná informace o složení daného organismu. Tato kombinace tvoří genetický kód. Také rozhoduje o tom, čeho bude buňka schopná a jak bude vypadat. Primární struktura této genetické informace zůstává během celého života buňky stejná. DNA nalezneme ve všech živých buňkách schopných dělení, a i v některých virech.

Ústřední (centrální) dogma molekulární genetiky

Jedná se o dědičnou informaci uloženou ve dvoušroubovici DNA a přepisuje se pomocí transkripce na RNA jednošroubovici, podle které se dále syntetizují bílkoviny. Tento proces je součástí:

Transkripce

Před tím, než se DNA začne rozplétat na jednošroubovici RNA je na DNA přiveden enzym helikáza, který rozplete šroubovici. Při opisu se Thymin u DNA přepisuje na Uracil v RNA a opis této rozpletené šroubovice má na starost RNA polymeráza. RNA molekula se pak ještě upraví, jelikož obsahuje i nekódující oblasti tzv. introny. Výsledku tohoto přepisu se říká mRNA, která opustí jádro a přesune se do cytoplazmy. V cytoplazmě pak mRNA přejde do ribozomu, který ji bude používat k tvorbě bílkovin.

Translace

Translace je proces, kdy ribozom vezme sekvenci uloženou v mRNA a přepíše ji. V mRNA jsou nukleotidy seřazeny do trojic neboli kodonů. Každý kodon kóduje vždy právě jednu aminokyselinu. Ribozom čte tyto triplety a jakmile narazí na iniciační sekvenci AUG začne jednotlivým tripletům přiřazovat příslušné aminokyseliny. Toto přiřazení má na starosti tRNA, která dodá správnou aminokyselinu na základě antikodonu

odpovídající právě aktuálně čtenému kodonu. Většina aminokyselin je kódována více než jedním kodonem. Tento proces řadí tRNA na mRNA a sousední prvky se navzájem spojují peptidovou vazbou do doby, než narazí na STOP kodon, kterým je UAA či UAG nebo UGA. Výsledný protein na základě peptidových vazeb tvoří strukturu proteinu a na základě těchto spojení se protein specifickým způsobem zabalí do sebe.

2.1.4 Geny

Základními jednotkami dědičnosti jsou geny neboli vlohy. Jsou to konkrétní úseky molekuly DNA, tedy kyseliny deoxyribonukleové, které mají specifické pořadí nukleotidů. Toto pořadí určuje strukturu a funkci genového produktu. Jedná se o genetický záznam obsahující informaci k tvorbě konkrétního proteinu, který se pak podílí na vytváření konkrétního znaku a jsou stavebními kameny organismů. Geny můžeme dělit dvojím způsobem.

- Dělení dle účinnosti při realizaci dědičného znaku
  - Monogenní geny – Mají velký účinek. Na tvorbě znaku se jich podílí málo (nejčastěji jen jeden) a zpravidla se jedná o znak kvalitativní jako je např. krevní skupina nebo barva vlasů.
  - Polygenní geny – Mají menší účinek. Na tvorbě znaku se jich podílí více a velkou roli zde hraje i vliv prostředí. Nejčastěji ovlivňuje kvantitativní znaky jako je výška, váha či predispozice k nemocem.
- Dělení dle funkce
  - Strukturní geny – Kódují strukturu proteinů. Obsahují informace potřebné k vytvoření proteinů.
  - Regulační geny – Díky nim vytvořené proteiny regulují expresi strukturních genů a mají vliv na diferenciaci buněk. Určují, zda bude gen transkribován.
  - RNA geny – Podle nich se syntetizují tRNA a rRNA a jsou významnou složkou při translaci.

Geny jsou uloženy na chromozomech za sebou, a to ve specifickém a neměnném pořadí. Mají tedy své jedinečné místo na určitém chromozomu a jeho určité části. Tomuto místu se říká lokus genu. Geny, jež jsou uloženy na jednom chromozomu jsou spolu v genové vazbě. Geny ve vazbě se dědí společně, pokud mezi nimi nedojde ke crossing-overu, tedy výměně genetického materiálu mezi otcovským a mateřským chromozomem.

2.1.5 Mutace

Mutací se rozumí změna genetické informace v DNA. Mutace může ovlivnit funkci a vliv genu na organismus. Neplatí však, že mutace je vždy pro organismus špatná. Nyní se podíváme na různé dělení mutací, kterých máme hned několik.

Dělení dle typu

Mutace jsou změny genotypu jedince oproti jeho normálnímu stavu a můžeme je dělit následovně:

- Spontánní – vznikají náhodou při replikaci či transkripci tzv. bodová mutace. Četnost těchto chyb je díky korekční funkci DNA polymerázy velice nízká a pohybuje se kolem 10^{-7} na 1 nukleotid. Mezi další takovéto chyby dochází při dělení buněk. Při nestejném crossing-overu mohou vznikat i chromozomové aberace,
- Indukované – jsou vyvolány vnějšími mutagenními vlivy tzv. mutageny.

Dělení z pohledu evoluce

Z pohledu evoluce můžeme mutace považovat za užitečné a můžeme je rozdělit na:

- Letální – neschopné reprodukce,
- Nevýhodné – ztratí nebo přijdou o svoji funkci genu,
- Neutrální – neovlivňují reprodukční schopnosti,
- Výhodné – zvyšují reprodukční schopnosti. Jsou preferované.

Dělení dle mutagenu

Mutageny jsou látky nebo děje, jenž jsou schopny vyvolat mutace při působení na buňky. U člověka mohou mutageny vytvořit i karcinogenní buňky. Mutageny můžeme dělit na:

- Fyzikální – UV záření nebo ionizující záření,
- Chemické – organická rozpouštědla, tabákový kouř, toxické látky, herbicidy,
- Biologické – retroviry, papilomaviry, herpesviry.

Dělení dle mutovaných buněk

Mutace z hlediska mutovaných buněk může buňky dělit na:

- Gametické – postihuje pohlavní buňky a přenáší se na potomka,
- Somatické – pokud se neoddělí chromozom během mitotického dělení a jedinec má pak několik linií buněk, kde má každá jiný karyotyp.

Dělení dle umístění genotypu

Mutace z hlediska umístění v genotypu rozlišujeme na:

- Genové – jedná se o změnu sekvence nukleotidů,
- Chromozomové (strukturní chromozomové aberace) – zlomy a výměny jednotlivých genů či části chromozomů,
- Genomové (numerické chromozomové aberace) – jedná se o změnu násobku celé řady chromozomů či změnu počtu jednotlivých chromozomů jako třeba trizomie 21 chromozomu (známá jako Downův syndrom).

Mutace genu

Tyto mutace mění nukleotidovou sekvenci u vláken DNA. Při mutaci v kódující oblasti DNA a podle důsledku na proteosyntézu hovoříme o mutaci:

- Neměnící smysl – nevede ke změně aminokyselin, většinou zamění třetí nukleotid kodonu,
- Měnící smysl – dochází k přidání nové aminokyseliny do polypeptidového řetězce,
- Posunové – proběhne změna struktury a vznikne nová bílkovina,
- Nesmyslné – vznik terminálního kodónu, což vede k ukončení translace,
- Elongační – jedná se o mutaci terminálního kodónu, jenž vede k prodloužení translace.

Mezi základní typy operací, které způsobují zmíněné mutace patří:

- Inzerce – jedná se o vložení jednoho či více párů bází. Pokud se jedná o jiný násobek než tří, vzniká posun čtecího rámce a tím důsledky popsané výše. Pokud se bude jednat o násobek tří, vloží se celé aminokyseliny a prodlouží se tím peptidový řetězec,
- Inverze – převrácení pořadí jednoho či více párů nukleotidů. To může stejně jako substituce vést k nesprávné syntéze enzymu,
- Delece – odstranění jednoho či více párů nukleotidů. Účinek je podobný jako u inzerce, avšak při násobku tří se zde zkracuje polypeptidový řetězec.
- Substituce – substituce sice neposouvá čtecí rámec, ale záměna aminokyseliny může vést k syntéze nesprávného enzymu. Substituce je nahrazení původní sekvence nukleotidů jinou. K tomu může dojít dvěma způsoby:
    - Tranzice – záměna pyrimidinové báze za pyrimidinovou nebo purinové za purinovou (A za G nebo C za T),
    - Transverze – záměna pyrimidinové báze za purinovou a naopak.

Mutace chromozomů

Strukturní chromozomové mutace se označují jako aberace a dochází u nich k mutaci struktury chromozomu. Vznikají působením mutagenu, kdy pak vznikne zlom chromozomu. Tato mutace se zjišťuje pomocí vyšetření karyotypu. Mutace chromozomů způsobuje změnu tvaru a struktury chromozomu.

Odlomená část chromozomu může podlehnout:

- Deleci – smaže se konec raménka (termální delece),
- Inverzi – připojí se v opačném směru,
- Translokaci – připojí se na jiný pár chromozomu,
- Duplikaci – zdvojí se úsek chromozomu.

2.2 Základy bioinformatiky

Bioinformatika je vědní obor, který nejvíce spojuje biologii, informatiku, matematiku a statistiku. Bioinformatika je relativně mladý vědní obor. Využívá výpočetní technologie pro zpracování biologických dat, jako je například sekvenování DNA nebo RNA, analýza mikroskopických obrazů nebo snímků magnetické rezonance, dále také např. predikce vlivu některých léků či predikce chování DNA řetězců.

Bioinformatickými daty se rozumí jakékoli informace, které se týkají zdravotnických dat, obrazů či vlastností pozorovaného objektu. Tato data bývají objemově velmi obsáhlá, tudíž ruční zpracování takového množství dat je velmi nepraktické.

Níže se podíváme na jednotlivé kategorie tohoto vědního oboru, které jsou zajímavé pro tuto práci a které jsou využívány před či při práci s aplikací.[5]

2.2.1 Sekvenování

Sekvenování je biochemický proces, kterým získáváme informace o sekvenci aminokyselin či nukleotidů v molekule DNA, RNA či v proteinu. Získání této informace nám pomáhá pochopit strukturu a funkce těchto molekul.

Vývoj metodických přístupů k sekvenování genomů

Lidský genom se skládá z molekul DNA a ty se dělí na 46 chromozomů, 22 párů autozomů a jednoho páru gonozomů. Velikost lidského genomu činí 3,2 miliardy nukleotidů a obsahuje až 25 000 genů. Kódující oblasti, tvořené exony strukturních genů, tvoří přibližně 1,5 % našeho genomu. Další části tohoto celku tvoří introny, pseudogeny, mobilní elementy, repetitivní sekvence a mnoho dalších úseků, které jsme dosud úplně přesně nepochopili.

Nejranější snahy o sekvenování začaly v 60. letech 20. století, tedy necelých 10 let po objevení struktury DNA (1953, James Watson a Francis Crick). K této snaze byl využit elektronový mikroskop a nikoli jeden z níže vysvětlovaných postupů.

K metodickému pokroku v tomto směru došlo v dalším desetiletí, konkrétně v roce 1977, kdy laboratoř Fredericka Sangera zavedla sekvenační metodu, za kterou získal Nobelovu cenu. Tato metoda potřebuje na začátku připravit tzv. knihovnu DNA, což je námi zvolený úsek DNA, který je pak sestříhán na nedlouhé úseky, jenž lze pak klonovat do vektorů. Následuje amplifikace pozměněných vektorů, která probíhá v plazminech bakteriálních buněk, z nichž jsou extrahovány části nesoucí klonované úseky DNA. Tyto části (sekvence) jsou vloženy do směsi obsahující DNA polymerázu. Proces samotné sekvenace se děje ve čtyřech zkumavkách. Na konci sekvenace vzniknou různě dlouhé fragmenty, které jsou pak rozděleny dle velikostí na polyakrylamidovém gelu elektroforézou.

Další způsob DNA sekvenování zavedli Allan Maxam a Waltert Gilbert tzv. Maxam-Gilbertovu metodu. Principem této metody jsou chemické reakce založené na degradaci nukleotidových bází. Každý řetězec je na svém konci označen radioaktivním fosforem. Poté dochází ke štěpení označené báze a následně k elektroforéze označených DNA fragmentů v polyakrylamidovém gelu. Následně je tento gel vystaven rentgenovému záření, které pak osvětlí části DNA označeny fosforem.

Tyto metody byly na počátku schopny přečíst kolem 100 bází v jednom běhu. Později byly díky automatizaci a dalšímu zdokonalování schopny přečíst až 1000 bází. Díky tomu mohly být využívány jakožto diagnostické metody v běžné lékařské praxi. Konkrétně byly využívány pro analýzu fragmentů DNA (např. produktů polymerázové řetězové reakci, tedy PCR reakci). Avšak pro skenování celých genomů se stále nehodily, jelikož tento proces byl velmi zdlouhavý a finančně náročný. Z tohoto důvodu bylo stále potřeba hledat efektivnější metodu, která by práci zjednodušila a také zpřesnila. Navíc žádná metoda stále nebyla dosud schopna osekvenovat celý genom.

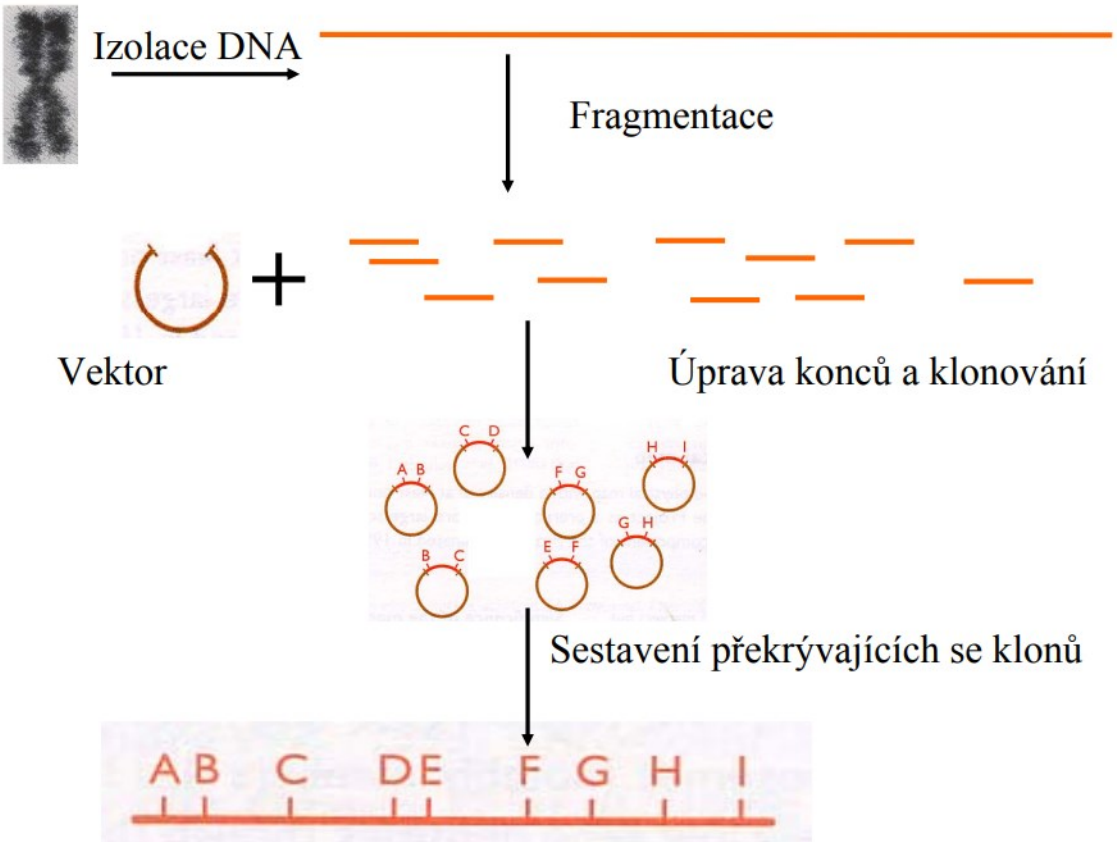
První kompletní genomy byly osekvenovány až v polovině 90. let a jednalo se o genom bakterie Haemophilus Influenzae. K tomuto úspěchu došlo v „The Institute For Genomic Research (TIGR), který byl založen Craigem Venterem. Venter o pár let později stál u zrodu společnosti „Calera Genomics“, která pokračovala ve výzkumu genomového sekvenování. Venter navrhl vlastní metodu sekvenování, tzv. „shotgun“ sekvenování, kdy dochází ke štěpení analyzované DNA na krátké úseky. Ty jsou dále klonovány do vektoru a poté čteny z obou konců. Získané úseky se pak vzájemně překrývaly, a tak bylo možné sestavit celou sekvenci.

K osekvenování kompletního lidského genomu došlo v únoru 2001. Výsledky těchto sekvenací prezentovali současně dva týmy – Venterův tým se společností Celera a International Human Genom Sequencing Consortium (se kterým byl Venter také nějaký čas spojen). Oba projekty poskytly obdobný odhad celkového počtu lidských genů, jenž se pohyboval mezi 26 000-30 000 transkriptů kódujících proteiny. Dalším objevem byly i údaje o struktuře genomu (1,1 % exony, 24 % introny a 75 % intergenové oblasti). Výsledky sekvenací (ač v současné době upřesněné a doplněné) vedly k dalšímu rozvoji molekulárně-biologických technologií a umožnily začátek nové éry genetiky člověka.

Sekvenování genomů lze tedy dělat dvěma odlišnými způsoby, nicméně Sangerovo metoda sekvenování je pokládaná za zlatý standard, i když je v posledních letech stále více nahrazována sekvenací nové generace, která se dostává do popředí kvůli jejímu rychlejšímu a levnějšímu přístupu. [6–10]

Enzymová (Sangerova) metoda

První enzymovou metodou je náhodné sekvenování, kde se z celého genomu náhodně vybere několik sekvencí (částí) o optimální velikosti (1300-2000 bp). Po úpravě zastřižených konců jsou nahodile naklonovány do vhodného vektoru. Tento postup předpokládá, že celková informace o vystřižené sekvenci odpovídá původnímu genomu. Pomocí univerzálních primerů připojujících se k vektoru v blízkosti klonovacího místa jsou stanoveny krátké sekvence s minimální délkou 500 bází. Tyto sekvence jsou poté použity k vytvoření jedné souvislé sekvence tzv. kontigů. Celý tento proces je popsán na Obr. 1. [11]



Obr. 1 Postup náhodného sekvenování

Druhou metodou je uspořádané sekvenování sousedních úseků. Tato metoda je vhodná pro malé fragmenty genomů a doplnění zbývajících mezer po náhodném sekvenování. Doplnění těchto mezer vyžaduje znalost vystřižené sekvence, ke které se bude připojovat primer, jenž umožní prodloužení řetězce. Tento proces nevyžaduje následné klonování a minimalizuje nadbytečné stanovení sekvencí. Přesto by správnost sekvencí měla být ověřena sekvenováním obou řetězců.

V současné době se používá automatická varianta tohoto sekvenování. Syntéza probíhá v jedné reakci a ke značení konců se využívají různé fluorescenční značky. Detekce probíhá během elektroforézy za pomoci laserového detektoru napojeného na

počítač, který vyhodnocuje označenou sekvenci a přiřazuje jednotlivé barvy jednotlivým nukleotidům. [8, 12, 13]

2.2.2 Sekvenace nové generace

Důvodem vzniku sekvenování nové generace, v anglickém jazyce Next Generation Sequencing (dále jen NGS) byl nedostatečně rychlý způsob sekvenace dat, jelikož lidský genom je obrovský a tradičními sekvenačními metodami by zkoumání lidského genomu zabralo příliš dlouho. Tato snaha se stala hlavním projektem pro výzkumníky na poli lidské genetiky a vznikl tak projekt, který je v angličtině znám jako Human Genome Project. NGS se stává stále častější formou sekvenace ve výzkumných laboratořích. Svoji popularitu získala hlavně díky rychlejšímu i finančně výhodnějšímu generování velkého množství dat. Díky NGS lidstvo dokázalo zmapovat celý lidský genom a již se získávají další a další typy genomů živočichů či genomů rostlin.

Technologie sekvenování NGS muší řešit dvě zásadní otázky. Zaprvé fyzické oddělení jednotlivých fragmentů nukleových kyselin a zadruhé pak samotné čtení sekvence. Fyzické oddělení jednotlivých fragmentů nukleových kyselin je řešeno emulzní PCR nebo amplifikací ve shlucích. Čtení sekvence je řešeno sekventováním syntézou a sekventováním založené na ligaci. Pro sekvenování je často užíván přístup „mate-pairs“, který umožňuje částečnou eliminaci problému repetitivních sekvencí tím, že se na začátku určí vzdálenost DNA fragmentů. Všechny níže zmiňované technologie s tímto přístupem pracují.

Od prvního osekvenování kompletního lidského genomu do současné doby se postupně vyvíjely následující technologie: ABI370XL, 454 Genome Sequencer FLX od společnosti Roche, Genome Analyser od firmy Illumina, SOLiD (tedy Sequencing by Oligo Ligation and Detection) od firmy Applied Biosystems či Helicos Biosciences „HeliScope Single Molecule Sequencer“ od firmy Pacific Biosciences. Jednotlivé technologie se liší ve spoustě parametrech, ať už se jedná o délku sekvencí nebo o počet zpracovaných fragmentů či době běhu sekvenačního procesu. [6–8, 14, 15]

2.2.3 Moderní metody zpracování NGS

Podkapitola Moderní metody zpracování NGS, jak název napovídá, vysvětlí některé technologické principy NGS.

Technologie firmy Illumina (Genome Analyzer IIX)

Je založena na metodě amplifikace ve shlucích, kterou využívá ke čtení sekvencí, jež jsou získány sekvenováním syntézou s detekcí pomocí reverzibilně terminačních fluorescenčně značených nukleotidů. Amplifikace i sekvenování probíhají na povrchu speciální destičky (flow-cell).

Na fragmenty nukleových kyselin jsou navázány adaptéry, a to na oba jejich konce. Tyto fragmenty jsou náhodně zachyceny na povrchu destičky a následně amplifikovány a sekvenovány buď pouze z jednoho konce (single-reads), nebo z obou konců (paired-end reads). Technologie Illumina vyprodukuje až 18 gigabáze za 9,5 dne, což vychází na zhruba 1,9 gigabáze za den. [6, 14, 15]

2.2.4 Softgenetics NextGENe®

Americká firma Softgenetics se zaměřuje na vývoj softwarových řešení pro genetickou analýzu. Jejich produkt NextGENe software, tedy Next Generation Sequencing Software je software pro analýzu genetických dat. Software obsahuje velké množství modulů, které umožňují uživatelsky přívětivou analýzu od primárních sekvenačních dat po vizualizaci výsledků a jejich anotaci a interpretaci. Oddělení molekulární geneticky ÚHKT, má tento software zakoupen, používá ho k analýze ve Standardních operačních postupech, proto jsou výstupy (mutační reporty) z Analýzy genetických variant (Variant Calling) využity jako vstupy do analýz NextDOM.[16]

2.3 Statistické zpracování genomických dat

V této podkapitole si vysvětlíme statistické zpracování na základě článku [17], který se zabývá genomem a snaží se porozumět genetickému příspěvku k onemocnění lidského organismu. Článek pracuje s daty z 1092 genomů a ze 14 populací (Evropa, Amerika, východní Asie a sub-saharská Afrika). Tato data byla zkonstruována na základě nízkého pokrytí celého genomu a exomů, to znamená částí genomu, které jsou v pozadí lidského genomu a mají tudíž menší vliv na genotyp jedince. Článek ukazuje, že jednotlivé ukázky genomů z různých geografických oblastí mají odlišné vzácné i běžné profily a variace. Mezi základní metody patří genotypování, což je proces identifikace genetických variant jako jsou SNP a krátké vložení či delece. Těchto metod získání dat je spousta a jsou klíčové pro získání informací o variantách jedince.

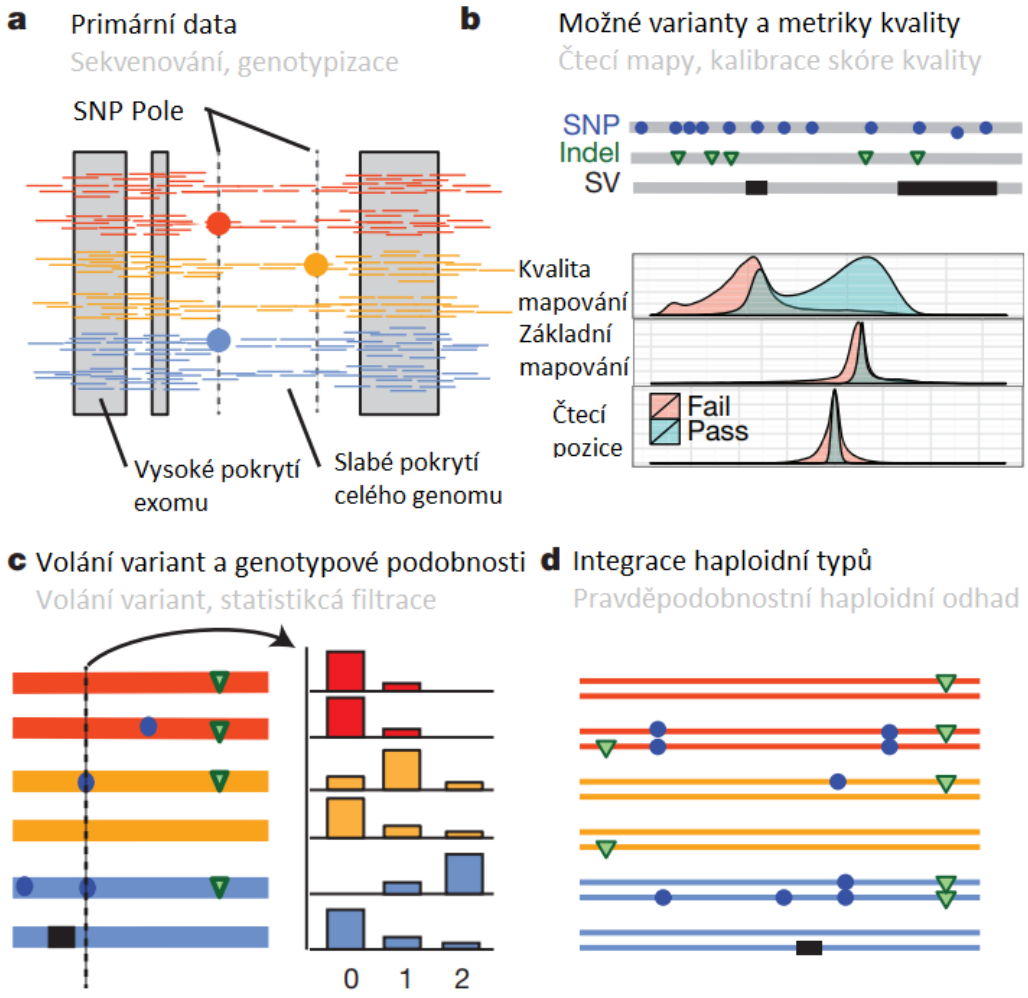
Přestože 95 % variant bylo objeveno už při první fázi projektu 1000 Genomes Project, málo časté varianty, a to převážně ty mimo kódující oblasti exomů jsou málo popsané. Tyto oblasti mohou mít vliv na potencionální mutace způsobené slabým výběrem a je vhodné je studovat. Protože tyto varianty tíhnou ke svým kořenům (genům v populaci) je pravděpodobné, že varianty budou větší či výraznější u různých populací. Charakterizace těchto variant vede ke zjištění funkčních částí, které budou důležité pro individuální sekvenci genomu, jenž pomůže oddělit sdílené varianty např. s rodinnými příslušníky.

Na základě těchto variant a dat v článku je vidět, že se vzácné varianty vyskytují jen u některých jedinců, a to většinou i u různých populací, zatímco většinu běžných genů má celá populace stejnou. Analýza těchto vzácných variant je nedílnou součástí přiřazení k určitým nemocem či fenotypům.

Analýza těchto genomů probíhala pomocí kombinace nízkého pokrytí (2-6), celého genomu zaměřeného na hluboké části (50-100), části exomů a hustých SNP genotypových dat. V úvodní fázi projektu se tento design ukázal jako velmi silný a levný nástroj v odhalování krátkých inzercí, delecí v indelových variantách a v genotypizaci SNP s výjimkou těch nejvzácnějších typů. Tento design byl dále vylepšen statistickými metodami pro výběr velmi kvalitních variant získaných vícero algoritmy k integraci SNP, indelových a větších strukturních variant v jednom celku. Postup je popsán na Obr. 2, kde:

a) primární struktura byla testovaná na skupinách po až 100 vzorcích ze stejné populační skupiny. Primární data jsou generována z každého vzorku a je patrné, že v exomech je mnohem více dat než mimo,
b) následuje čtecí zarovnání a vícero algoritmů identifikuje vhodné kandidáty. Pro každou z variant byla použita vhodná metrika pro získání kvalitních dat,
c) pomocí strojového učení byla natrénovaná data na velmi kvalitních známých variantách, které nám dovolí s jistotou určit skupiny a prahové hodnoty pro nízkou detekci prvků, které jsou falešně pozitivní (False Detection Rate),
d) protože je obvykle důkaz na pokrytí v genotypu s nízkým pokrytím dost slabý a jeho variabilita může být v exomech velmi vysoká, byly použity statistické metody pro využití informací ze vzorů vazebné nerovnováhy.

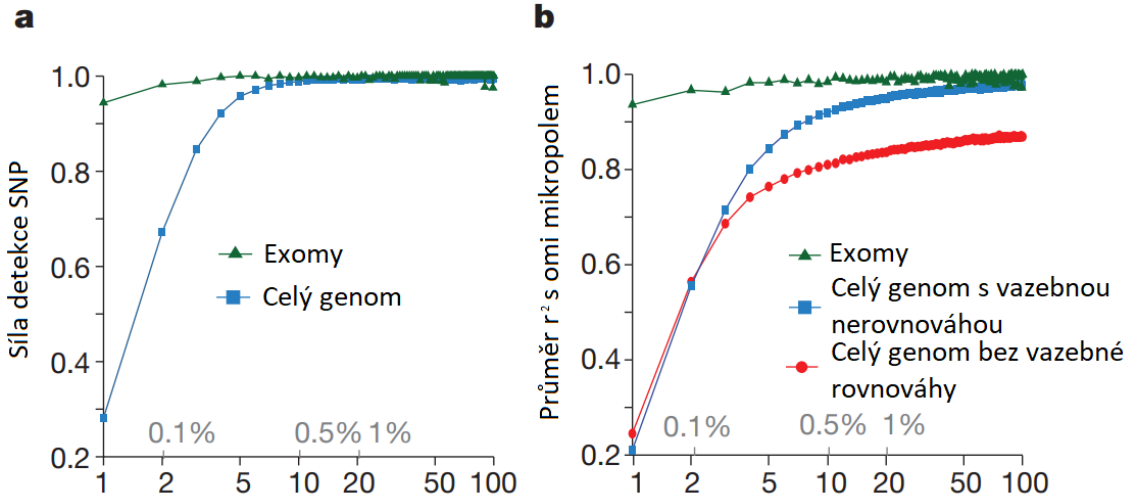
Celkově bylo objeveno a genotypováno 38 milionů SNP, 1,4 milionu dvou-alelických indelů a 14 000 velkých delecí. Bylo použito mnoho technologií pro validaci a kontrolu falešně pozitivních výsledků (FDR). Výsledky byly velmi jasné, avšak 3 ze 185 exomů, 5 z 281 nízko pokrytých oblastí a 72 z 3 415 velkých delecí nešlo validovat. Při nezdarech na začátku projektu byly přidány další filtry, které snížily detekci falešných výsledků na hodnotu 5,4 %. K tomu se ještě přidaly problémy s nízko pokrytým SNP (2,1 %) a 18 % indelových oblastí, které měly nekonzistentní nebo nejednoznačné výsledky. To vede k dalším výzvám, které je potřeba překonat při zkoumání těchto částí genomu. [17]



Obr. 2 Průběh algoritmu první fáze

Na základě Obr. 3, je vidět, že při použití vazebné nerovnováhy jsou genotypy z nízkého pokrytí skoro stejně přesné jako u exomů s hlubokým pokrytím.

Nicméně nemá cenu používat tuto metodu pro získávání genotypů, jelikož je výsledek méně přesný.



Obr. 3 Síla a přesnost

2.3.1 Genetické variace uvnitř a mezi populacemi

Asi 94 % variant bylo již objeveno před tímto projektem a jejich haploidní struktury byly zmapovány. Oproti tomu pouze 62 % variant v krajních mezích od 0,5 až 5 % a také 13 % variant menších než 0,5 % bylo prozkoumáno.

Pro analytické účely jsou populace seskupeny podle převažujících složek původu jedince na: Evropu, Afriku, východní Asii a Ameriku. Pokud bychom nastavili hranici významnosti pro varianty na více než 10 %, měly by informace obsaženy v této oblasti všichni jedinci. Změny jsou častější a silnější u jedinců žijící v daném místě s lokálními předky, kteří se adaptovali na podmínky daného podnebí. Bylo pozorováno, že lidé z Afriky mají až 3krát více variant pod 5% hranicí než například lidé z Evropy či Ameriky.

2.3.2 Využití 1000 Genomes Projektu

Data z tohoto projektu se hojně používají při detekci variant objevených exomů a genetických nemocí či rakovinných genomových projektech. Využitím dat z tohoto výzkumu se velmi zlepšila síla detekce, a to hlavně u vzácných a málo častých běžných variant nalezených u všech jedinců z různých částí světa.

2.3.3 Statistické metody

Statistické metody by se zde použily především k identifikaci jednotlivých genů k dané skupině onemocnění. Dále by se statistické metody daly využít k predikci toho, jaký dopad by mohly mít dané varianty na strukturu proteinů, regulaci genové exprese a dalších biologických funkcí.

Statistické metody hrají klíčovou roli v analýze genomických dat. Tyto metody nám umožní identifikovat genetické varianty, porozumět vazbám mezi geny a chorobami a odhalit funkční důsledky jednotlivých variant na různých oblastech genomu. Celkově lze tedy říci, že nám statistické metody pomáhají zobrazit genetické informace a jejich vliv na různé oblasti v lidském životě a statistickým způsobem tyto data interpretovat.

V této práci bylo použito konkrétní statistické rozložení, a to normální a Poissonovo rozdělení, které napomohlo definovat různé jevy a pravděpodobnosti.

Normální rozdělení se většinou používá pro detekci špatně či chybně naměřených dat, což může být například chyba měření z důvodu možné závady na přístroji či poklesu napětí v elektrické síti. Toto rozdělení předpokládá rovnoměrné rozložení dat tzv. Gaussovo rozdělení, které by mělo být kolem střední hodnoty a mělo by se řídit Centrálním limitním teorémem, který říká že: „součet nezávislých vzorků libovolného rozdělení s konečnou střední hodnotou a rozptylem konverguje k normálnímu rozdělení s rostoucí velikostí vzorku do nekonečna“, neboli čím více máme dat tím spíše se dá říct,

že se hodnoty ustálí na nějaké střední hodnotě. Což dovoluje odhalit odchylky od očekávaného chování a identifikovat potenciální problémy.

Poissonovo rozdělení se používá pro detekci ojedinělých či málo pravděpodobných jevů. Takovým jevem může být právě pravděpodobnost mutace, která by u zdravého člověka měla být velmi malá. Díky tomuto rozdělení je možné vyhodnotit, zda se vyskytuje neobvyklé či nadměrné množství těchto jevů. Tato rozdělení jsou užitečným nástrojem pro kvantifikaci a analýzu jevů.

Dále byla data normalizována a to z důvodu odstranění zkreslení dat, jejichž rozsah se může výrazně měnit při každém výpočtu. Normalizace tyto rozdíli převede na stejný rozsah, čímž odstraní zkreslení či odlehlé hodnoty tzv. outliery a umožní přesnější porovnání a analýzu. [18, 19]

2.4 Ph+ ALL a CML

Tato podkapitola se bude zabývat základními informacemi o zkoumané nemoci. Jedná se hlavně o Ph pozitivní akutní lymfoblastickou leukémii a chronickou myeloidní leukémii.

2.4.1 Ph pozitivní akutní lymfoblastická leukémie

Ph+ ALL je vzácným poddruhem nejčastější dětské rakoviny, jíž je akutní lymfatická leukémie (dále jen ALL). Stejně jako ALL a CML jedná se o typ rakoviny bílých krvinek. Oproti ALL je Ph+ ALL známá tím, že má předem známou mutaci ve svém genetickém kódu, která spojuje dva geny dohromady. Ph+ ALL je podskupinou ALL a ta se od ostatních podskupin liší přítomností fúzního genu BCR::ABL1. Tento gen je také znám jako filadelfský chromozom a ten může způsobit rakovinu bílých krvinek. Přestože je filadelfský chromozom velmi vzácný v pediatrii, mnohem častěji se vyskytuje u dospělých. Jedná se také o hlavní příčinu vzniku CML. [20–22]

2.4.2 Chronická myeloidní leukémie

CML je rakovinné onemocnění, které postihuje bíle krvinky a má velmi pomalý postup nemoci, jenž může trvat i několik let. Toto postižení se obvykle objevuje až ve starším věku (kolem 60 let), ale může postihnout kohokoli. CML je jen jedním typem leukémie. Tento typ není tak nebezpečný jako akutní myeloidní leukémie (dále jen AML), avšak 15 až 30 % pacientů s diagnostikovanou CML v chronické fázi se postupně vyvine do CML v blastické fázi, která má pro většinu pacientů smrtelné následky. Toto onemocnění postihuje kostní dřeň, která se vyskytuje převážně ve velkých kostech jako je např. kost lebeční, pánevní nebo páteř.

Nemoc se projevuje tím, že kostní dřeň produkuje velké množství bílých krvinek. Těchto krvinek je hodně a produkují se velmi rychle, což vede k tomu, že krvinky nejsou

zcela vyvinuté, a proto nemohou plně plnit svojí funkci. Nemoc se obvykle neprojevují v raném stádiu, ale až v pozdější fázi. Většinou se na CML přijde náhodou.

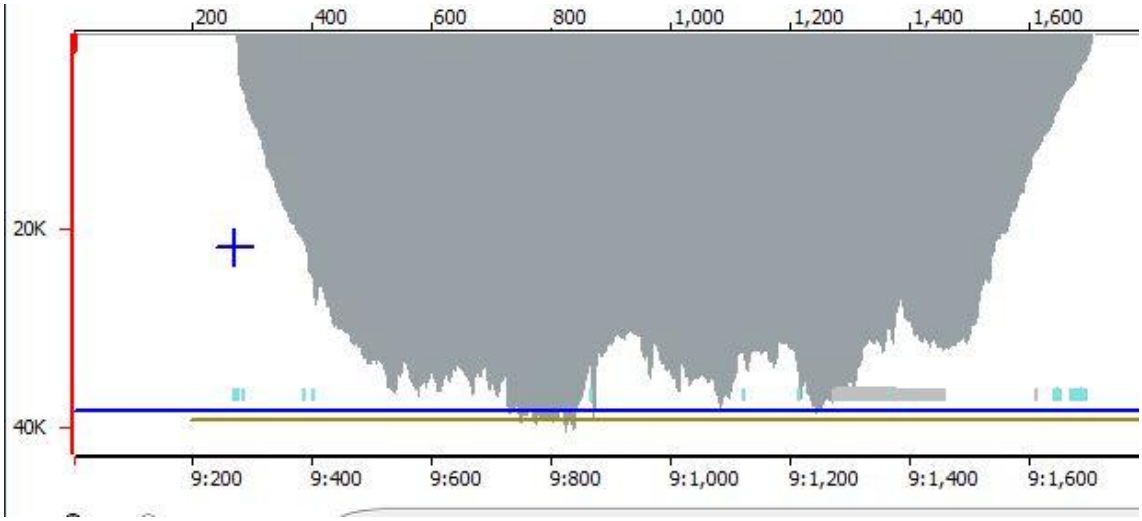
Léčba se zahajuje prakticky okamžitě při zjištění, aby se zpomalil postup nemoci. Hlavním lékem jsou inhibitory tyrozinkinázy, které ve většině případech dokáží nemoc udržet pod kontrolou, avšak pacient je medikován po zbytek svého života. V současné době se průměrná doba dožití pacienta s diagnostikovanou CML výrazně neliší od běžné populace. [1, 23]

2.5 Současný stav aplikace NextDOM2

Současný stav aplikace využívané na ÚHKT je funkční aplikace napsaná v programovacím jazyce MATLAB a je využívána pravidelně. Software pracuje s aplikačním rozhraním a obsahuje dvě části. První se věnuje výpočtu prahových hodnot a ta druhá hodnocení statistické významnosti jednotlivých variant zjištěných u pacientů na základě vypočtených mezních hodnot.

2.5.1 Výpočet prahových hodnot

První část aplikace obsahuje výpočet prahových hodnot tzv. thresholdů, které se dají využít pro analýzu získaných dat. Z dřívější práce bylo zjištěno, že při hlubokém amplikonu sekvencí provedené pomocí Next Generation Sequencing (dále jen NGS) vznikají záměny a mutace na utrčitých pozicích v nukleotidech DNA sekvence. Základní myšlenkou výpočtu prahových hodnot spočívá v nalezení individuálních prahových hodnot pro jednotlivé pozice v nukleotidu a detekci mutací přes celou kinázovou doménu fúzního genu BCR::ABL1.



Obr. 4 Počet pozic při měření

Pro statisticky významný výpočet těchto prahových hodnot je nutné přečíst jednotlivé pozice minimálně na 1000 hodnot viz Obr. 4. Při sekvenování obecně bývají začátky a konce čtených sekvencí méně čtené. Proto nejsou pozice, na kterých je čtení nižší zahrnuty do výpočtů, aby nezkreslovaly vypočítané prahové hodnoty. Tyto části jsou většinou odlehlými hodnotami, které se dají považovat za statisticky nevýznamné ne-li chybné. Z tohoto důvodu musí být tyto hodnoty odstraněny, aby nevnášely do výpočtu chyby. Čtení každého nukleotidu je normalizováno na 3000 čtení a jednotlivé prahové hodnoty jsou vypočteny na hladině významnosti 95% Tento postup stále plně neošetří falešnou pozitivitu či negativitu. Z tohoto důvodu je precizní hledání variant důležité pro správné zhodnocení prahové hodnoty pro každou sekvenovanou pozici.

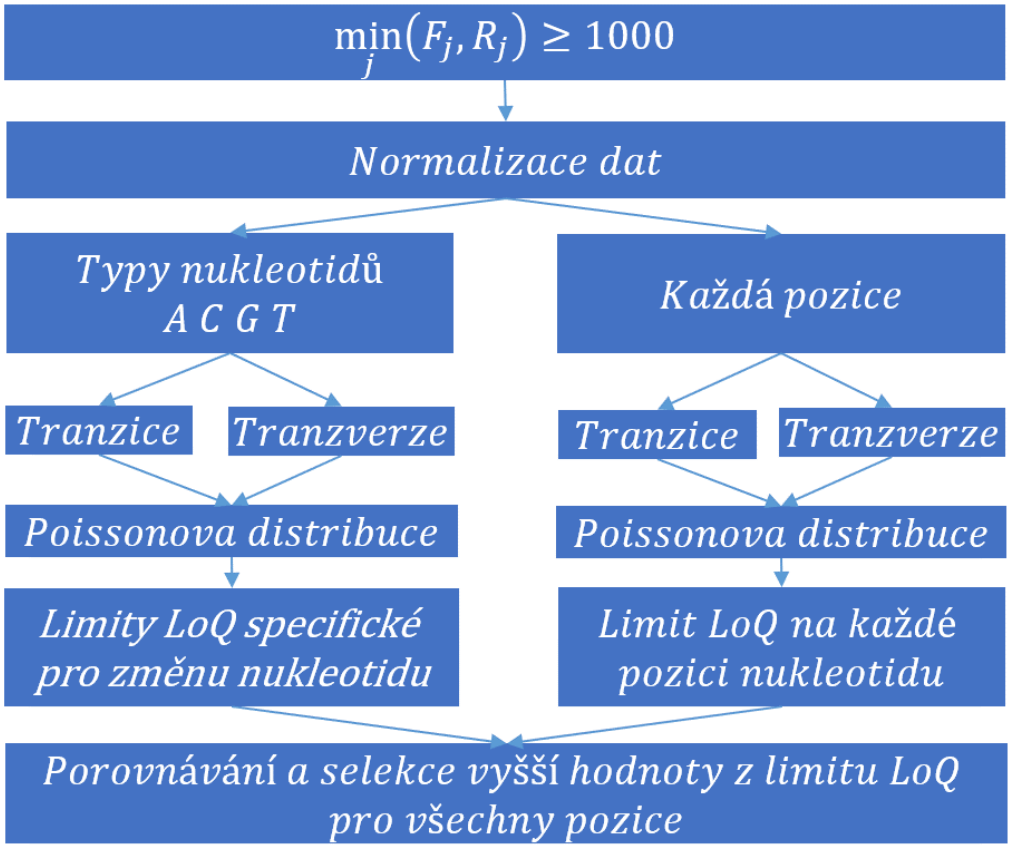
Výpočet prahových hodnot využívající data NGS probíhá na základě sekvenačních dat zdravých dárců. Zdravým dárcem je všeobecně osoba, která nemá známé genetické poruchy, netrpí žádnými příznaky zkoumaného onemocnění a ani v jeho rodinné anamnéze nebyly detekovány žádné závažné genetické vady. Velké odchylky v souborech zdravých dárců by měly projít důkladnou kontrolou, jelikož by mohly zvýšit riziko falešně pozitivních výsledků. Výpočet prahových hodnot probíhá za předpokladu, že data zdravých dárců nemají žádné somatické změny a odpovídají kritériím a nastaveným hladinám významnosti. Vstupním genetickým materiálem je RNA izolovaná z leukocytů 30 zdravých dárců. Knihovna kinázové domény byla připravena pomocí Nextera XT DNA Library Preperation Kit od firmy Illumina a sekvenována na sekvenátoru MiSeq také od firmy Ilumina. [2, 24]

Algoritmus výpočtu

Jak již bylo zmíněno výpočet prahových hodnot probíhá na základě sekvenačních dat zdravých dárců, díky kterým se stanoví pozadí na jednotlivých pozicích pro běžnou zdravou populaci a na základě odchylek od těchto dat se stanoví prahové hodnoty. Celý proces je podrobněji popsán na Obr. 5.

Vstupními daty pro analýzu může být excelovský dokument (.xlsx) a nebo textový dokument s definovanou strukturou (.csv).

Algoritmus nejdříve provede dopředné a zpětné čtení genu a následně data normalizuje. Poté, co jsou data normalizována, provede rozdělení na 2 skupiny. První skupina obsahuje typy nukleových bází (A: Adenin, C: Cytosin, G: Guanin, T: Thymin) a druhá jednotlivé pozice. Na obou se poté počítá tranzice a transverze. Poté algoritmus využívá normální a Poissonovo rozdělení, které efektivním a robustním způsobem určuje optimální hranice prahových hodnot. Na základě pravděpodobnostních rozdělení vzniknou LoQ hodnoty u obou skupin a poté se porovnájí a vybere se vyšší hodnota pro danou pozici. Přehled substitucí na našich datech mezi zdravými dárci a nemocnými je přehledně vidět v Tabulka 1 Přehled substitucí.



Obr. 5 Diagram výpočtu prahových hodnot

Tabulka 1 Přehled substitucí

LoQ	Tranzice	Transverze
A	55,6	15,1
C	18,2	15,1
G	18,3	15,1
T	55,2	15,1

Normalizovaná hodnota na pozici v nukleotidu se vypočítá dle rovnice níže, kde NPN je výsledná normalizovaná hodnota, 3 000 jsou normalizované hodnoty čtení, X je počet čtení daného typu nukleotidu, Y je celkový počet nalezených nukleotidů tohoto typu a písmenem j je označen řádek pozice.

$$NPN = 3000 * X_j/Y_j$$

Výsledek se ukládá do excelovské tabulky (.xlsx) či do více textových souborů ve formátu .csv a obsahuje individuální výsledky pro LoB (Limits of Blank), LoD (Limits of Detection) a LoQ (Limits of Quantification).

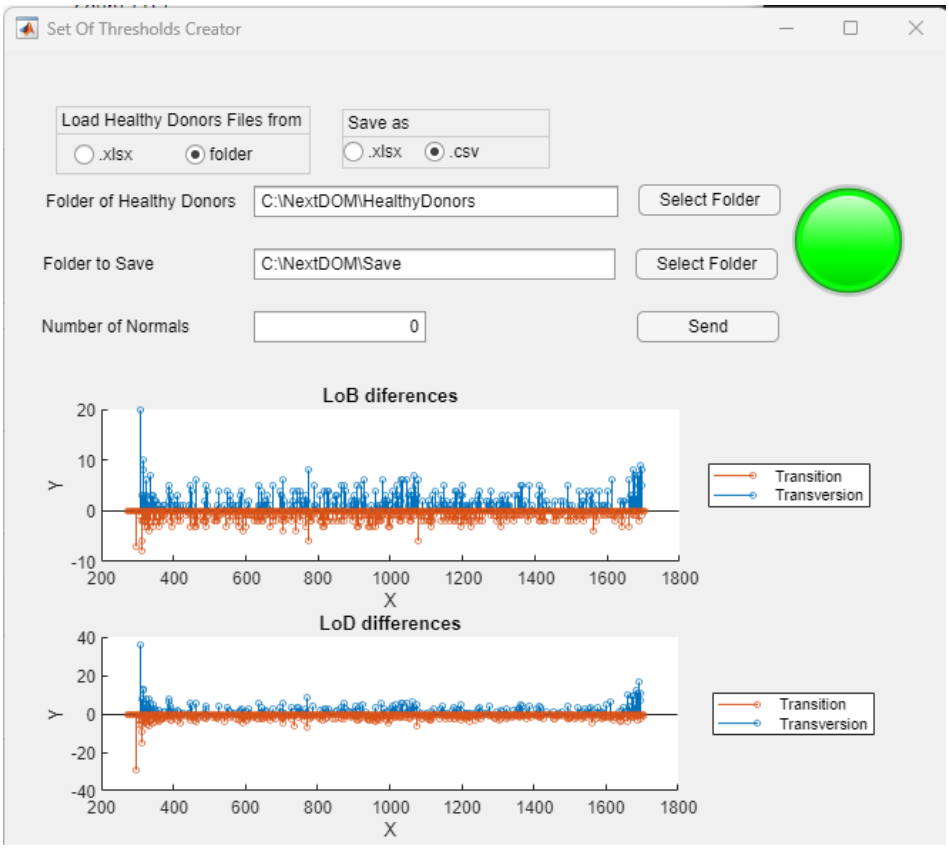
Vzhledem k tomu, že genetické mutace jsou poměrně vzácné, je třeba využít citlivé metody detekce pomocí statistických metod, které jsou schopny odhalit i velmi nízké

úrovně mutací. Tyto metody mohou ale vytvářet i tzv. LoB hodnoty, jenž nejsou důsledkem mutací, ale může se jednat o šum či variabilitu v datech. Zároveň prahová hodnota pro detekci těchto mutací, neboli LoD hodnota, nám říká, že vše pod touto hodnotou nelze se statistickou určitostí určit jako mutaci. Zvednutím statistické prahové hodnoty pro detekci těchto mutací je možné snížit riziko detekce falešně pozitivních výsledků, avšak tím se sníží i citlivost testu. Obvykle se však tato p-hodnota pohybuje mezi 0,3-0,5. Pokud jsme však schopni identifikovat mutaci na některé pozici v genu, ale nejsme schopni ji kvantifikovat můžeme předpokládat, že jsme našli minimální hladinu mutace (LoQ hodnotu), jenž jsme schopni spolehlivě najít ve fúzním genu BCR::ABL1.

Následně aplikace vykreslí graf, který zobrazuje rozdíly mezi limity pro jednotlivé pozice pro každý druh substituce jako je transverze a tranzice.

Tato aplikace v současné chvíli vypadá jako na Obr. 6, kde uživatel nahrává soubor s daty zdravých dárců, určuje složku, kam se výsledek výpočtu prahových hodnot uloží a po odeslání se mu vykreslí grafy a uloží výsledek do předem vybrané složky.

Je potřeba prahové hodnoty vypočítat pokaždé, pokud se něco změní v procesu přípravy knihovny, protože každá změna může ovlivnit pozadí prahových hodnot a tím ovlivnit konečný výsledek analýzy.



Obr. 6 Ukázka aplikace pro výpočet prahových hodnot

2.5.2 Analýza

Druhou částí aplikace je analýza fúzního genu BCR::ABL1. Ta analyzuje data, která jsou rozdělena do dvou částí (Part 1 a Part 2). Tyto části jsou výsledky generované jinou aplikací s názvem NextGENe, která již byla v práci dříve zmíněna. Tyto části jsou výsledkem dvou analýz. Part 1 je vygenerován analýzou, která předpokládá, že každá sekvenovaná pozice je zmutovaná a obsahuje čtecí bázi na jednotlivých pozicích v dopředném i zpětném čtení. Tato analýza nám získala zdrojová čtecí data pro každou pozici. Part 2 již extrahuje pozice s možným seznamem bodových mutací na jednotlivých pozicích včetně záměn aminokyselin, jejichž pravděpodobnost je větší než 0,5 %.

Následně jsou data normalizována a aplikují se na ně limity LoD a LoQ k jednotlivým pozicím a k získání potenciálních variant. Varianty, které překročí prahové hodnoty jsou dále porovnávány se seznamem klinicky relevantních mutací, keré jsou zjištěny a použity na základě článku [21]. V rozšířené verzi je možné nahradit tento seznam svým vlastním. Ukazatele správně zpracovaného vzorku jsou také průměrným pokrytím a také určují oblast, která splňuje kvalitativní kritéria (1 000 čtení v dopředném i zpětném čtení).

Celá analýza je rozdělena na tři části, jejíž vstupními parametry jsou: cesta ke složkám s daty Part 1 a Part 2, jedinečný název analýzy, cesta ke složce, kam chceme uložit výsledky analýzy a zda chceme provést analýzu pomocí limitů LoD a nebo pomocí LoQ. Na základě zvoleného limitu se průběh analýzy liší. Pokud zvolíme LoD, může analýza objevit chyby v přípravě knihoven aplikace kontrolou pozic zdravých dárců či analýzy. Pacientova data jsou analyzována pomocí LoQ, které obsahují výsledné relevantní varianty.

Výsledkem této analýzy je více textových souborů ve formátu .csv a nebo excelovský soubor, který aplikuje podmíněné formátování, aby usnadnil další zpracování výsledných dat a zlepšil v nich orientaci.

Algoritmus analýzy

Vstupními soubory pro analýzu jsou data z NGS, která jsou rozdělena na dvě části. Part 1, který obsahuje dopředné a zpětné čtení pro jednotlivé aminokyselinové báze pro jednotlivé pozice v sekvenování části. Ukázku vstupních dat do této části můžeme vidět na Obr. 7.

Software	NextGENe	V2.4.2.2			
Project		Name:B8M1A_S1.pjt			
Date/Time:	02.02.2023	7:41:15			
Index	Pos	A#(F;R)	C#(F;R)	G#(F;R)	T#(F;R)
1	272	3700;1886	0;0	5;1	0;0
2	273	3718;1892	0;0	11;1	1;0
3	274	2;0	0;0	4028;1944	0;0
4	275	0;0	4146;2295	1;0	3;0
5	276	1;0	4153;2443	0;0	1;1
6	277	1;0	4410;2509	1;0	2;0
7	278	0;1	37;18	1;1	4378;2684
8	279	3;1	12;8	0;0	4487;2728
9	280	1;0	4509;2742	0;0	2;2
10	281	4502;2738	1;1	8;8	2;0
11	282	4;4	0;1	4543;2751	0;0
12	283	0;1	4547;2844	1;0	3;1
13	284	3;2	0;0	5120;2986	0;0

Obr. 7 Ukázka vstupních dat Part 1

První část analýzy používá složku Part 1, která obsahuje NGS analýzu jednoho či více probandů. Na základě hodnot u dopředného a zpětného čtení se vytváří nová data, která obsahují součet těchto čtení pro jednotlivé báze. Dále vytváří hlavičku souboru, která obsahuje datum, název projektu, použitý software a dále vytváří nadpisy pro jednotlivé báze a vytváří jejich součet. Výsledek se ukládá do souboru, který se dále používá pro další část analýzy a jehož ukázku si můžeme prohlédnout na Obr. 8

Software	NextGENe V2.4.2.2					Average			Dolní mez:		272	A28															
Project Name	B8M1A_S1.pjt					31714			Horní mez:		1686	F497															
Date/Time	02.02.2023 7:41:15																										
doména	pořadí AA v proteinu	ref. NK	ref. AMK	Index	Pos	A#(F)	A#(R)	C#(F)	C#(R)	G#(F)	G#(R)	T#(F)	T#(R)	-- >	A	C	G	T		suma	//	F	R				
		A		1	272	3700	1886	0	0	5	1	0	0		5586	0	6	0		5592		3705	1887				
		A		2	273	3718	1892	0	0	11	1	1	0		5610	0	12	1		5623		3730	1893				
	28	G	A	3	274	2	0	0	0	4028	1944	0	0		2	0	5972	0		5974		4030	1944				
		C		4	275	0	0	4146	2295	1	0	3	0		0	6441	1	3		6445		4150	2295				
		C		5	276	1	0	4153	2443	0	0	1	1		1	6596	0	2		6599		4155	2444				
	29	C	L	6	277	1	0	4410	2509	1	0	2	0		1	6919	1	2		6923		4414	2509				
		T		7	278	0	1	37	18	1	1	4378	2684		1	55	2	7062		7120		4416	2704				
		T		8	279	3	1	12	8	0	0	4487	2728		4	20	0	7215		7239		4502	2737				
	30	C	Q	9	280	1	0	4509	2742	0	0	2	2		1	7251	0	4		7256		4512	2744				
		A		10	281	4502	2738	1	1	8	8	2	0		7240	2	16	2		7260		4513	2747				
		G		11	282	4	4	0	1	4543	2751	0	0		8	1	7294	0		7303		4547	2756				
	31	C	R	12	283	0	1	4547	2844	1	0	3	1		1	7391	1	4		7397		4551	2846				
		G		13	284	3	2	0	0	5120	2986	0	0		5	0	8106	0		8111		5123	2988				
		G		14	285	2	4	0	0	5232	3074	4	0		6	0	8306	4		8316		5238	3078				
	32	C	P	15	286	0	0	5248	3540	0	0	2	0		0	8788	0	2		8790		5250	3540				
		C		16	287	1	0	5464	3616	0	0	2	2		1	9080	0	4		9085		5467	3618				
		A		17	288	5490	3627	1	1	6	5	1	1		9117	2	11	2		9132		5498	3634				
odsud amplikon zdravý dárce	33	G	V	18	289	5	2	0	0	5689	3688	1	1		7	0	9377	2		9386		5695	3691				
		T		19	290	4	0	9	5	1	0	5700	3822		4	14	1	9522		9541		5714	3827				
		A		20	291	5713	3829	3	2	7	3	1	0		9542	5	10	1		9558		5724	3834				

Obr. 8 Ukázka výstupních dat Part 1

Vstupními soubory pro druhou část analýzy jsou další soubory z NGS analýzy. Označené jako Part 2, což si můžeme prohlédnout na Obr. 9

Software	NextGENe V2.4.2.2																
Project Name:	B8M1A_S1.pjt																
Date/Time:	02.02.2023 7:51:29																
Index	Chrom	Pos	Coverage	Alt	Ref#(F;R)	Alt#(F;R)	Alt%	A#(F;R)	C#(F;R)	G#(F;R)	T#(F;R)	Overall Score	Mutation Call: Genomic	Mutation Call: Relative To mRNA	Function	Gene	Amino Acid Change
1	9	278	7121	C	4378;2684	37;18	0,77	0;1	37;18	1;1	4378;2684	30,8	T>CT	c.278T>CT	Missense	ABL1	p.L29PL
2	9	321	15567	C	9533;5922	28;63	0,58	6;0	28;63	10;5	9533;5922	33,5	T>CT	c.321T>CT	Synonymous	ABL1	p.S43SS
3	9	331	16758	T	9944;6697	75;41	0,69	1;0	9944;6697	0;0	75;41	33,7	C>CT	c.331C>CT	Missense	ABL1	p.R47RC
4	9	354	19637	C	11083;8433	55;58	0,58	3;4	55;58	0;0	11083;8433	34,3	T>CT	c.354T>CT	Synonymous	ABL1	p.L54LL
5	9	372	21993	G	12841;9009	78;60	0,63	12841;9009	2;0	78;60	3;0	34,7	A>AG	c.372A>AG	Synonymous	ABL1	p.E60EE
6	9	376	22123	A	12822;9165	85;47	0,6	85;47	1;1	12822;9165	2;0	34,7	G>AG	c.376G>AG	Missense	ABL1	p.D62ND
7	9	386	23266	C	13436;9687	75;50	0,54	2;2	75;50	0;1	13436;9687	34,9	T>CT	c.386T>CT	Missense	ABL1	p.L65PL
8	9	388	23455	C	13526;9799	68;55	0,52	3;2	68;55	1;1	13526;9799	34,9	T>CT	c.388T>CT	Missense	ABL1	p.F66LF
9	9	444	31443	G	19001;1225	111;64	0,56	19001;12258	3;0	111;64	4;2	35,9	A>AG	c.444A>AG	Synonymous	ABL1	p.K84KK
10	9	465	33381	G	19303;1389	86;81	0,5	19303;13898	3;0	86;81	5;4	36,1	A>AG	c.465A>AG	Synonymous	ABL1	p.L91LL
11	9	485	35826	G	21223;1429	216;64	0,78	21223;14294	8;0	216;64	9;6	36,4	A>AG	c.485A>AG	Missense	ABL1	p.E98EG
12	9	533	38893	G	22074;1658	120;100	0,57	22074;16583	3;6	120;100	3;4	36,7	A>AG	c.533A>AG	Missense	ABL1	p.N114NS
13	9	613	39872	T	19839;1973	143;146	0,72	2;0	19839;19737	2;3	143;146	36,8	C>CT	c.613C>CT	Synonymous	ABL1	p.L141LL
14	9	642	36595	A	16877;1939	165;159	0,89	165;159	0;0	16877;19391	2;1	36,5	G>AG	c.642G>AG	Synonymous	ABL1	p.L150LL
15	9	647	36954	A	17512;1918	128;126	0,69	128;126	0;1	17512;19184	1;0	36,5	G>AG	c.647G>AG	Missense	ABL1	p.R152HR

Obr. 9 Ukázka vstupních dat Part 2

Druhá část analýzy využívá vytvořený soubor z části 1 a kombinuje ho s analýzami vloženými ve složce „Part 2“. Na základě zvoleného limitu se zvolí koeficient, který se používá pro násobení hodnot v tabulce. Soubory ve složce „Part 2“ obsahují také dopředné a zpětné čtení, avšak obsahují navíc informace o možných mutacích a změnách aminokyselin na jednotlivých pozicích zkoumaného řetězce. Výsledkem druhé části je spojení těchto informací s výsledkem první části.

Třetí část vezme výsledný soubor druhé části a na základě seznamu známých mutací aminokyselin aplikace zkoumá, zda na zadané pozici vznikla některá z mutací a pokud ano, tento řádek označí. Také na základě výběru limitu, aplikace vytváří dva různé soubory.

LoD limit nám k souboru přidá další část k souboru z druhé části. Pokud se daná aminokyselinová mutace nachází na nějaké pozici v genu, přípíše se do sloupce CRM. Tabulka napravo od CRM ukazuje jednotlivé tranzice a transverze, které ukazují, na kterých řádcích, dle algoritmu, došlo k mutaci genu, jak lze vidět na Obr. 10.

			tranzice	cetnosti		%			transverze	cetnosti		%	
Gene	Amino Acid Change	CRM		app.matrice	sample	app.matrice	sample			app.matrice	sample	app.matrice	sample
ABL1	L42L		316	4,8191	26,1690647	0,16063667	0,87230216	0	392	5,8647	18,728223	0,19549	0,6242741
ABL1	S43S		483	4,8093	5,4039357	0,16031	0,18013119	0	447	7,7061	9,76899207	0,25687	0,32563307
ABL1	V67G		485	15,44	22,262885	0,51466667	0,74209617	0	772	9,4462	12,7504554	0,31487333	0,42501518
ABL1	E98G		488	4,8093	6,55111232	0,16031	0,21837041	0	823	6,232	6,8372358	0,20773333	0,22790786
ABL1	C100R		498	4,8191	5,27900917	0,16063667	0,17596697	0	826	5,8647	5,99600266	0,19549	0,19986676
ABL1	S143G		503	4,8191	5,05544134	0,16063667	0,16851471	0	1076	9,2626	11,4995827	0,30875333	0,38331942
ABL1	F149L		513	4,8191	9,29011411	0,16063667	0,30967047	0	1108	6,0561	6,6613033	0,20187	0,22204344
ABL1	V199A		518	4,8093	5,33035009	0,16031	0,17767834	0	1193	5,8647	26,3092225	0,19549	0,87697408
ABL1	E281G		527	4,8191	6,80554668	0,16063667	0,22685156	0	1232	5,8647	7,3452256	0,19549	0,24484085
ABL1	F283S		573	4,8093	7,25423939	0,16031	0,24180798	0	1492	6,0685	7,14803688	0,20228333	0,2382679
ABL1	L284S		584	4,8191	8,72065165	0,16063667	0,29068839	0	1509	5,8647	6,03936773	0,19549	0,20131226
ABL1	K285K		593	4,8093	5,89220169	0,16031	0,19640672	0	1511	5,8647	6,72872042	0,19549	0,22429068
ABL1	K291K		619	15,44	31,8044968	0,51466667	1,06014989	0	1622	5,8647	52,8332877	0,19549	1,76110959
ABL1	I293T		622	4,8093	5,10384233	0,16031	0,17012808	0	1661	8,8947	9,62660443	0,29649	0,32088681
ABL1	L298P		637	17,448	35,8465651	0,5816	1,1948855	0	0	0	0	0	0
ABL1	V299A		639	4,8191	5,14071047	0,16063667	0,17135702	0	0	0	0	0	0
ABL1	T306A		653	4,8093	9,33170857	0,16031	0,31105695	0	0	0	0	0	0
ABL1	T315I	CRM	662	4,8093	5,47578486	0,16031	0,18252616	0	0	0	0	0	0

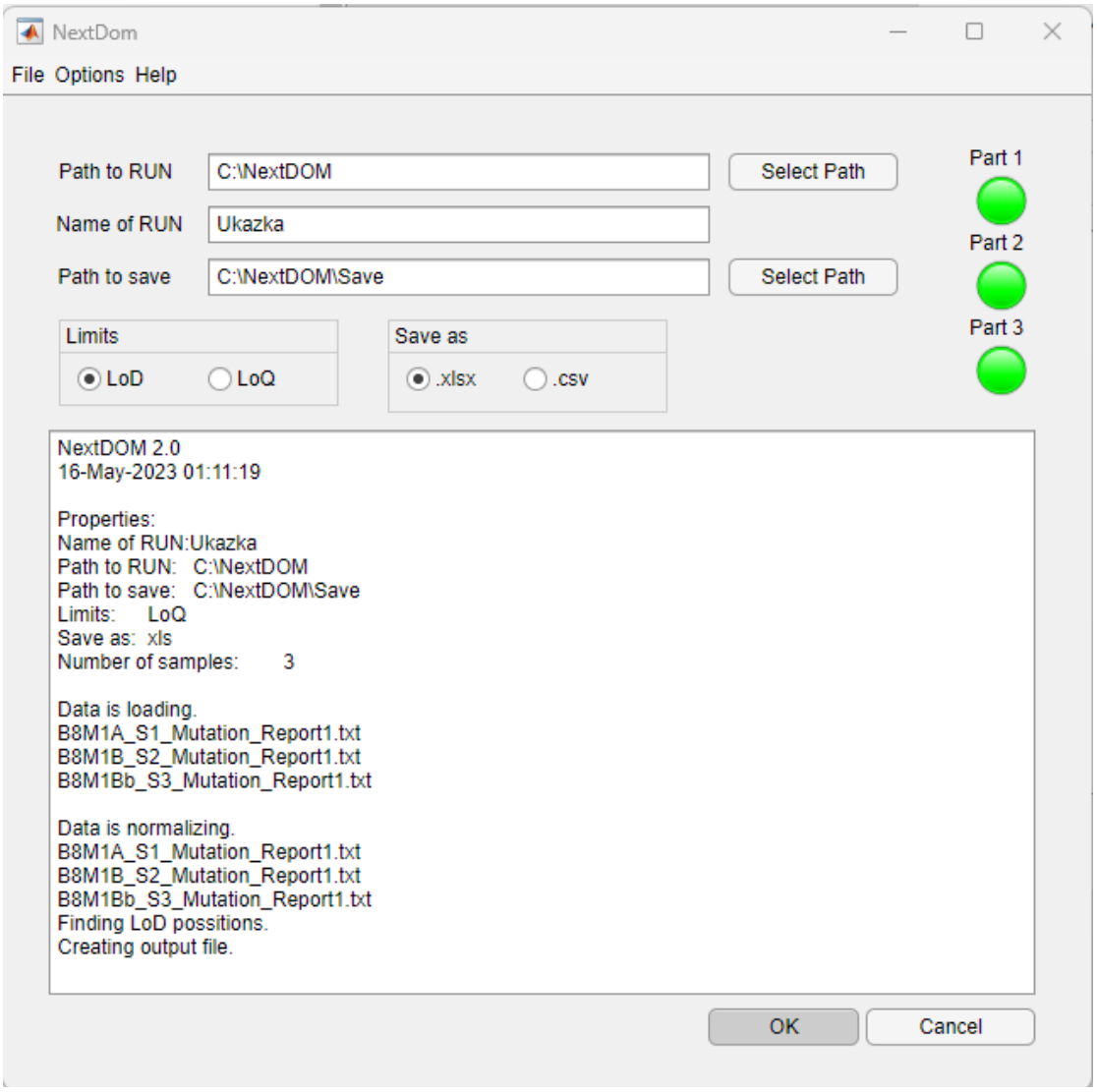
Obr. 10 Ukázka výstupních dat při použití LoD

LoQ limit na druhou stranu k souboru přidá jinou tabulku těchto mutací, jelikož koeficient výpočtu je odlišný. Dále přidává do excelovské tabulky podmíněnné formátování, které barevně zvýrazní všechny nalezené mutace. Barevně je určuje na základě pravděbodobnostních hodnot, jak je patrné z Obr. 11 .

		Synonymous													
		above 2%													
		below 2%													
					tranzice	cetnosti		%			transverze	cetnosti		%	
Mutation	Function	Gene	Amino Aci	CRM		app.matri	sample	app.matri	sample			app.matri	sample	app.matri	sample
c.316C>CT	Synonymc	ABL1	L42L		316	24,0955	26,1691	0,80318	0,8723	0	1622	29,3235	52,8333	0,97745	1,76111
c.321T>CT	Synonymc	ABL1	S43S		1088	80,525	81,0711	2,68417	2,70237	0	0	0	0	0	0
c.392T>GT	Missense	ABL1	V67G		1136	24,0955	202,298	0,80318	6,74327	0	0	0	0	0	0
c.485A>AC	Missense	ABL1	E98G		1196	86,4	110,484	2,88	3,6828	0	0	0	0	0	0
c.490T>CT	Missense	ABL1	C100R												
c.619A>AC	Missense	ABL1	S143G												
c.637T>CT	Missense	ABL1	F149L												
c.788T>CT	Missense	ABL1	V199A												
c.1034A>A	Missense	ABL1	E281G												
c.1040T>C	Missense	ABL1	F283S												
c.1043T>C	Missense	ABL1	L284S												
c.1047A>A	Synonymc	ABL1	K285K												
c.1065A>A	Synonymc	ABL1	K291K												
c.1070T>C	Missense	ABL1	I293T												
c.1085T>C	Missense	ABL1	L298P												
c.1088T>C	Missense	ABL1	V299A												

Obr. 11 Ukázka výstupních dat při použití LoQ

Funkční aplikace, vypadá jako na Obr. 12. Je zde i ukázka logu, který vypisuje, která část algoritmu je momentálně počítána.



Obr. 12 Ukázka aplikace pro analýzu dat

3 Cíle práce

Cílem práce je navrhnout a implementovat statistické metody pro stanovení prahových hodnot při detekci somatických bodových variant a následná analýza dat, jenž je hlavní částí tohoto řešení. Součástí této části je také návrh a implementace webové aplikace, která bude uživatelsky přístupnější, bude ji možné aktualizovat a její nasazení bude jednoduché a multiplatformní. Je zapotřebí vybrat vhodné technologie a přidat dokumentaci, která bude popisovat nasazení a práci s aplikací.

Dalším cílem je seznámení se základními znalostmi bioinformatiky, statistiky a formáty dat, které využívají sekvenování nové generace (NGS), jenž využívá aplikace při analýze dat. V neposlední řadě také prostudovat problematiku genetických podstat leukémie za pomoci metod využívajících NGS.

4 Metody

Tato kapitola popisuje metody použité v současně používané aplikaci jakož i metody použité v této práci. Další část se zabývá srovnáním obou programovacích jazyků.

4.1 Srovnání programovacích jazyků

Programovací jazyky MATLAB a Python použité pro tento algoritmus mají několik odlišností, které si popíšeme níže.

- Použití – MATLAB je často využíván ve vědeckém výzkumu či při matematickém modelování. Je specializován na práci s maticemi a numerickými výpočty, zatímco Python se používá v široké škále oblastí, jako je strojové učení, analýza dat, automatizace nebo pro práci s řetězci.
- Syntaxe – MATLAB má vlastní syntaxi zaměřenou na maticové operace a numerické výpočty, zároveň umožňuje vypisovat výsledky do konzole tím, že na konec řádky nenapíše středník. Python má velmi jednoduchou syntaxi a jednotlivé části a úrovně kódu jsou odděleny odsazením.
- Licence – MATLAB je komerční software, který vyžaduje koupi licence k provozu. Python oproti tomu je open-source a je vyvíjen komunitou vývojářů.
- Integrace – MATLAB sice podporuje integraci s jinými jazyky a technologiemi, ale mnohem silnější propojení má pro řešení napsané ve svém vlastním prostředí. Python má snadné propojení se všemi technologiemi a programovacími jazyky. To je způsobené velkou komunitou, která jí neustále rozšiřuje a obsahuje mnoho knihoven pro rozšíření jeho funkcí.

4.2 Srovnání základních metod

Podkapitola popisuje jednotlivé rozdíly použitých metod, které jsou používané u původní používané verze aplikace v programovacím prostředí MATLAB a verzí napsanou v programovacím jazyku Python.

MATLAB je určen pro numerické výpočty a práci s maticemi, což mu umožňuje velmi dobře pracovat s dvoudimenzionálními daty. Oproti tomu Python, který je dobrý pro práci s textovými řetězci, zde lehce zaostává. Tento problém řeší knihovna NumPy, která dokáže velmi efektivně pracovat s numerickými operacemi při velkém množství dat. Dále knihovna Pandas tvořící tzv. DataFramy. Tyto DataFramy ukládají data do struktury, která poskytuje rychlý a přehledný způsob zobrazení dat.

V Tabulka 2 si ukážeme jednotlivé způsoby výpočtu v MATLAB a jeho porovnání oproti Pythonu. Tabulka 2 obsahuje pouze základní rozdíly, které ukazují rozdíly mezi těmito jazyky.

Tabulka 2 Základní rozdíly programového prostředí

Popis	MATLAB	Python
Mocnění čísel	a .^ b	a ** b pow(a, b)
Logické operace	a && b, a || b	a and b, a or b
Chybějící hodnoty	NaN	nan
Vektory	a=[2 3 4 5]	a=array([2,3,4,5])
Spojení vektorů	[a b]	concatenate((a, b))
Násobení dvou vektorů	a.*a	a*a
Matice	a = [2 3;4 5]	a = array([[2,3], [4,5]])
Zploštění	a‘(:)	a.flatten()
Indexace (Python od nuly)	a(2,3)	a[1,2]
Suma v řádku	sum(a‘)	a.sum(axis=1)
Násobení matic	a .* b	a * b or multiply(a, b)
Vyhledávání v poli	[i j] = find(a)	(i, j) = a.nonzero() (i, j) = where(a!=0)
For cykly	for i=1:5	for i in range(1,6):

Jak již bylo řečeno, Tabulka 2 Základní rozdíly programového prostředí obsahuje pouze základní příkazy a jejich ekvivalenty. Tyto příkazy jsou pouze ilustrační a většina z nich se v této práci použila. Nicméně některé části se přepisovaly lehce jinak, a to na základě použitých knihoven, které usnadňují práci s velkým množstvím dat.

4.3 Použité metody

V minulé podkapitole byly vysvětleny základní rozdíly a ukázky ekvivalentních částí kódu. V této podkapitole budeme už rozebírat konkrétní části kódů, které byly v práci použity. Následující tabulky budou obsahovat výběr důležitých částí původního algoritmu a jeho ekvivalentní přepis do Pythonu. Jelikož je MATLAB určen pro pracování s tímto typem souborů, bylo během práce nutné najít jiné, podobné řešení

v programovacím jazyce Python. Z toho vyplývá, že se musely použít knihovny či jiná řešení.

Tabulka 3 Skutečné rozdíly programovacího prostředí

Popis	**Přečtení CSV souborů**
MATLAB	T = readmatrix(fullfile(app.matrixfile,filenames{i}));
Python	T = pd.read_csv(files[i], encoding='latin-1', sep=";")
Popis	**Přidání do pole**
MATLAB	U{i} = T(5:1437,18:21)
Python	U.append(T.iloc[4:, 17:21])
Popis	**Suma lichých řádku**
MATLAB	Forw = sum(normala(:,1:2:end),2);
Python	Forw = normala.iloc[:, 0::2].sum(axis=1)
Popis	**Vyhledání v poli**
MATLAB	zpos = find(~[0 misto' 0]);
Python	zpos = np.where(misto==[0])
Popis	**Normalizování na 3000 řádků**
MATLAB	norm_AG = AG_1vzorek.*(3000*size(U,2))./suma_genu(A)
Python	AG_1vzorek.iloc[:, 0] * np.divide(3000*len(U), suma_genu.iloc[:][A == True]
Popis	**Normální rozdělení**
MATLAB	A = norminv(0.05,kde_mean,sm_odch);
Python	a = norm.ppf(0.05, kde_mean, sm_odch)
Popis	**Ukládání do souboru**
MATLAB	dlmwrite(fullfile(app.folder,'SoTNextDOM.csv'),LoD,'Delimiter‘,'\t'
Python	df.T.to_csv(os.path.join(folder_toSave, 'SoTNextDOM.csv'), sep='\t', index=False,header=False)

MATLAB pro práci s excelovským souborem využívá actxserver, který vytvoří OLE objekt, který na základě vstupního ID parametru určuje, s jakou komponentou bude komunikovat. V programovacím jazyce Python nic podobného neexistuje. Z tohoto důvodu bylo rozhodnuto veškerý tento obsah načítat pomocí knihovny Pandas, která využívá tabulkovou strukturu pro snazší manipulaci s daty.

Tabulka 4 Rozdíly při práci s excelovským souborem v programovacím prostředí

Popis	Práce se soubory typu XLSX

MATLAB	e = actxserver('excel.application');
	eActivesheetRange = get(e.Activesheet, 'Range','A1:B3');
	eActivesheetRange.Value = T[0][0];

Python	T_sablona = pd.read_excel("sablona.xlsx") #načtení dat z excelu
	Wb = openpyxl.Workbook() # vytvoření workbooku
	Wb.cell(row=1, column=1, value=T[0][0]) # zapsání dat do prvního řádku a sloupce workbooku
	wb.save(cestaSouboru) #uložení a vytvoření souboru

Tabulka 5 se výrazně liší od MATLAB metody. Bylo potřeba nejprve zjistit průměrnou hodnotu, kterou jsme vložili do knihovny Scipy v druhém a čtvrtém řádku.

Tabulka 5 Rozdíly statistických funkcí v programovacím prostředí

Popis	Poissonovo rozdělení
MATLAB	AG_L = poissfit(norm_AG);
Python	mu = np.mean(norm_AG) var = poisson.stats(mu, moments='mvsk') AG_L = var[0] AG_k = [AG_L, poisson.ppf(alfa, var[0])]

5 Návrh aplikace

Tato kapitola specifikuje jednotlivé požadavky vycházející ze zadání práce. Ty byly vytvořeny na základě setkání s vedoucím a konzultantkou práce, a to před začátkem a v průběhu práce. Kapitola popisuje řešení jednotlivých požadavků a použitých technologií.

5.1 Funkční požadavky na aplikaci

Funkční požadavky jsou seznam vlastností a potřeb, které chce uživatel, aby aplikace splňovala. Většina těchto požadavků vychází ze zadání této práce a jsou stručně popsány v kapitole 3 Cíle práce. Nyní se na ně zaměříme konkrétněji.

1. Veřejná dostupnost – Webová aplikace bude veřejně dostupná a ke stažení společně s dokumentací či postupy, jak s ní pracovat.
2. Přehledný a optimalizovaný kód – Kód aplikace bude dodržovat zásady pro psaní přehledného kódu (odsazení, komentáře atd.). Vývojář se také pokusí optimalizovat či vylepšit již napsaný algoritmus nebo upravit řešení tak, aby odpovídalo vybrané technologii. Programátor vybere vhodný typ licence.
3. Virtualizované prostředí – Webová aplikace poběží v Dockeru, který usnadní nasazení na všech typech zařízení a dovede tím převést aplikaci do jakéhokoli operačního systému a také pro jakýkoli operační systém.
4. Interaktivní grafy – Webová aplikace bude obsahovat interaktivní grafy výsledků, které bude uživatel moci přizpůsobovat. To znamená přiblížit, posunout či zobrazit jen určitá data.
5. Logovací data – Webová aplikace bude při běhu analýzy vytvářet log, který bude uživateli dávat zpětnou vazbu o průběhu analýzy.
6. Výběr prahových hodnot – Aplikace počítá prahové hodnoty pro budoucí analýzu dat. Je potřeba zajistit, že se vypočtené hodnoty budou automaticky aplikovat pro analýzu dat či jiným způsobem zajistit, že bude možné tyto hodnoty v aplikaci měnit.

5.2 Technické požadavky na aplikaci

Technické požadavky jsou seznam technických specifikací na aplikaci, jenž musí projekt splnit. Tyto požadavky většinou nevychází ze zadání práce a je potřeba zjistit další informace u zadavatele. Tyto informace je vhodné zjistit před začátkem práce, jelikož pozdější požadavky, lze v některých případech velmi složitě implementovat do již implementovaného řešení.

1. Dokumentace – Vývojář vytvoří dokumentaci, jež bude dodávána s aplikací. Dokumentace se bude soustředit hlavně na uživatelskou příručku popisující ovládání samotné aplikace, a také přípravu i nasazení vývojového prostředí.
2. Sběr souborů, dat a analýz – Aplikace si uchovává veškeré soubory, které do ní uživatel nahraje pouze po nezbytně nutnou dobu výpočtu, jelikož se jedná o zdravotnická data pacientů, které by aplikace neměla z bezpečnostních důvodů uchovávat.
3. Výběr technologie – Vývojář si vybere programovací jazyk aplikace, který nebude zatížen licenčními poplatky a bude možné ho provozovat na libovolném operačním systému.
4. Aktualizace – Vývojář zajistí, že bude možné aplikaci aktualizovat či upravovat její obsah a algoritmus.

5.3 Zvolené technologie

Tato podkapitola popisuje možné technologie, pomocí kterých výsledné řešení mohlo být implementováno. Dále popisuje vybraný programovací jazyky, zvolené knihovny, aplikační servery nebo další technické komponenty, které byly při realizaci požadovaného řešení práce použity.

5.3.1 Průzkum „trhu“

Zde si popíšeme jednotlivá technologická řešení, která v průběhu výzkumu a výběru technologií byla nebo mohla být použita.

Například dle práce v článku [7], vytvářeli aplikaci využívající Sangerovu metodu sekvenování a použili k tomu webovou aplikaci napsanou v HTML. Formátování a vzhled stránek vytvořili pomocí Bootstrap frameworku a JavaScriptu. Pro výpočetní část použili flaskový mikro-framework napsaný v jazyce Python a pro dočasné uchovávání výsledků využili databáze Redis.

Dalším příkladem by mohla být aplikace Blast (Basic Local Aligment Search Tool). Jedná se o algoritmus a software, který porovnává dodané sekvence s databází sekvencí, kde jsou hledány shody. Tento algoritmus lze napsat v mnoha jazycích jako je např. Perl, Python, Java nebo C++. Lze ho také použít na operačních systémech Windows, Linux, MacOS a je možné využít i oficiální webovou verzi. [25, 26]

Další ukázkou by mohlo být řešení PacBio sequencing. To využívá jednu velkou molekulu k vytvoření dlouhých sekvencí DNA. Toto řešení většinou využívá aplikace třetích stran, ale má i vlastní řešení od firmy PacBio s názvem SMRT Analysis, které je naprogramované v programovacím jazyce Python. [27]

Vzhledem k tomu, že algoritmus NextDOM je samostatným řešením, tzn. nepotřebuje využívat řešení jako je například Blast, bylo možné vybrat například aplikace jako Elektron, NW.js, ASP.NET Core, které mají vlastní prostředí a jedná se o samostatné aplikace. [28, 29]

Na základě průzkumu technologií a podobných projektů, jako je téma této zadané práce, byly vybrány technologie, které jsou popsané v další části této kapitoly. Hlavními důvody výběru těchto technologií je mimo jiné minulá zkušenost s těmito technologiemi a určitá představa o tom, co vše bude k práci s nimi potřeba.

5.3.2 Výběr technologií uživatelského rozhraní

V této podkapitole si probereme výběr programovacích jazyků, které byly vybrány k této práci, a jejímž úkolem je vytvořit „webovou aplikaci“. Zabývá se výběrem jazyka pro front-end část práce, která má na starost vzhled a uživatelské rozhraní aplikace.

HTML a kaskádové styly

Hlavním úkolem této práce je vytvořit webovou aplikaci. Jako první možnost tvorby byl zvolen Hyper Text Markup Language (dále jen HTML). Jedná se o tagovací jazyk určený k tvorbě webových stránek. Webové stránky jsou tvořeny pomocí takzvaných tagů (neboli značek), které rozdělují dokument na jednotlivé sekce, odstavce či jednotlivé elementy. HTML se používá k tvorbě webových stránek a určuje jejich obsah, avšak pokud chce uživatel vylepšit vzhled svých stránek, často sáhne po kaskádových stylech (dále jen CSS). CSS není povinností, avšak většina stránek napsaná čistě v HTML by byla pro uživatele silně nečitelná a uživatelsky nepřívětivá. Na internetu je mnoho frameworků, které obsahují předem připravené knihovny, kde uživatel již nemusí psát čistě svá nastavení vzhledu pomocí CSS, ale může použít již hotová řešení, která většinou stačí mírně upravit. [30, 31]

V diplomové práci jsou využívány HTML a CSS soubory, aby bylo vytvořeno pro uživatele příjemné webové prostředí, jednoduchá navigace mezi jednotlivými cíli a díky formulářům i předávána data od uživatelů do aplikační vrstvy. Ta má na starost zpracování těchto vstupních dat a navrácení odpovědi zpět uživateli. Tato odpověď se opět zobrazí na HTML stránce. Pro snadnější a modernější vzhled byla zvolena šablona napsaná ve frameworku s názvem Bootstrap a následně byl upraven vzhled šablony k aktuálnímu vzhledu. Některé dynamické části stránky byly napsané pomocí jazyka JavaScript.

5.3.3 Technologie pro zpracování dat

V této části jsou popsané technologie, které mají na starost zpracovávání požadavků, a tvoří logickou výpočetní vrstvu aplikace. Tyto technologie přebírají

informace zadané uživatelem a na pozadí aplikace počítají složité operace. Výsledky těchto operací většinou vrací uživatele do grafického rozhraní.

JavaScript

Jedná se o nejpopulárnější jazyk při tvorbě webových aplikací a obvykle slouží jako logická vrstva nad HTML stránkou. Tento jazyk dokáže implementovat složité funkce a pracovat přímo na webové stránce. Obvykle se JavaScript používá, pokud je potřeba zpracovat některá data a pokud chceme, aby se stránky chovali dynamicky.

Byla snaha práci řešit pouze pomocí CSS a Pythonu a JavaScriptu se vyhnout, avšak ne úplně úspěšně. Práce s tímto skriptovacím jazykem výrazně ulehčuje některé části kódu, které by se pomocí CSS nastavovaly velmi složitě. [32]

Python (Flask)

Python je open-source programovací jazyk, který je objektově orientovaný a jedná se o dynamicky interpretovaný jazyk. To znamená, že případné chyby objeví až při spuštění souboru a je potřeba interpreter (tlumočník) k předávání programového kódu procesoru. Z tohoto důvodu je ale také velmi jednoduchý a rychlost učení se tomuto programovacímu jazyku je velmi rychlá. Tento jazyk se hojně používá při automatizaci a skriptování či třeba při práci v data science oboru, kde pomocí knihoven NumPy, Pandas, Scipy nebo Statistics zvládá jednoduše statistické metody a analýzu dat. [33–35]

Pro tvorbu či zpracování logiky výpočtu prahů či analýzy byl zvolen programovací jazyk Python. Důvodem zvolení tohoto programovacího jazyku bylo to, že se jedná o open-source programovací jazyk. Tento důvod zároveň splnil technický požadavek na nulové zatížení licenčními poplatky a tento jazyk je navíc multiplatformní. Dalším důvodem výběru Pythonu bylo to, že je možné použít framework „Flask“, který lze použít pro tvorbu webových aplikací. Framework Flask dovolí vytvořit RESTové API, které bude zpracovávat dotazy vytvořené webovou aplikací a bude sloužit jako logická vrstva aplikace pro výpočty a zpracování vstupních dat, které uživatel vkládá pomocí formulářů ve webovém prohlížeči.

5.3.4 GIT

Git je open-source distribuovaný verzovací systém, který zefektivňuje tvorbu projektů všech velikostí. Git umožňuje vývojáři přehledně sledovat změny ve zdrojovém kódu, vyvíjet jednotlivé části kódu ve větvích. To zajistí, že se uchová původní program bez zásahu do funkčního kódu. Také je díky němu možné přehledně a efektivně rozdělovat práci pro více lidí. [36]

Git byl v projektu využit jako verzovací systém i jako repositář, který uchová kód ke stažení. Byl vybrán, protože je jedním z nejpoužívanějších řešení, které programátoři používají při práci na projektech.

5.3.5 Docker

Docker je softwarová platforma, která dovoluje vytvářet, nasazovat a testovat aplikace. Jednotlivé programy zabalí do balíčků, které nazývá kontejnery. Každý takovýto kontejner obsahuje vše, co ke svému běhu potřebuje. Využití Dockeru nám pomáhá k rychlejšímu nasazení a zabezpečení toho, že námi požadovaná aplikace či program poběží na přesně zadaném prostředí. Z tohoto důvodu se nemůže stát, že by námi vytvořené prostředí na jiném zařízení nefungovalo tak, jak jsme zamýšleli. [32, 33]

Tato platforma se hojně využívá ve světě pro efektivní a lehce opakovatelné nasazení softwarových aplikací na různých operačních systémech a prostředích. Jedná se již o standart pro kontejnerovou virtualizaci. Využitím Dockeru se zvyšuje produktivita, rychlost nasazení nebo jednodušší správa aplikací. Tyto důvody jsou jedním z hlavních důvodů, proč je Docker v posledních letech důležitým nástrojem při vývoji a provozu softwarových aplikací a většina projektů bere toto řešení za standartní postup. [37, 38]

5.3.6 Licence

Pojmem licence se myslí právní dohoda či smlouva, která stanovuje práva a povinnosti poskytovatele. Licence určuje podmínky, které uživatel musí dodržovat, aby mohl využívat daný produkt. Tyto dohody mají ochránit práva držitelů autorských práv a poskytnou základní ujednání mezi uživateli a autory.

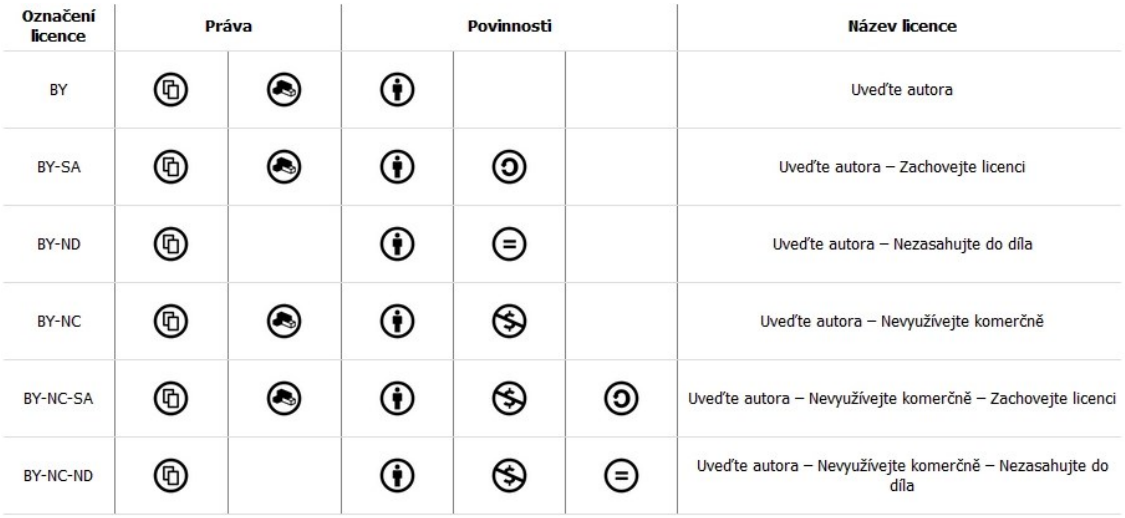
Licence softwaru je právní dohoda, jenž určuje podmínky, za kterých mohou uživatelé tento software používat. Tato pravidla většinou stanovují, jak může uživatel s daným softwarem nakládat. [39–41]

Druhy licencí

Existuje mnoho druhů licencí a nyní si popíšeme pár, které by se dali pro finální produkt této práce použít.

- GNU GPL – Většina licencí pro software a jiné práce se snaží omezit svobodu uživatelů šířit a upravovat vaši práci. Licence GPL je zamýšlena tak, aby vám dovolila svobodně šířit a upravovat všechny verze vašeho softwaru, a to i pokud se uživatel rozhodne upravenou verzi prodávat. Poté však bude nutné i těmto uživatelům dát stejná práva na úpravu, jaká zadal první autor této licence.
- BSD a MIT – Licence Barkley Software Distribution a Massachusetts Institut of Technology nevyžadují šíření odvozených děl pod stejnou či obdobnou licencí a téměř nijak neomezuje uživatele. Lze říct, že dílo odvozené od původního softwaru je prakticky bez rizika porušení licenčních podmínek. Uživatel poté může software využít do svého díla, avšak musí vždy uvést jméno autora a upozornit, že původní autor nenese žádnou odpovědnost.

- Creative Commons licence – Creative Commons (dále jen CC) je souborem veřejných licencí, jenž definují práva autora a za jakých podmínek smí uživatel dílo využít. CC licence je ve stále větší oblibě a hlavním důvodem tohoto nárůstu je mezinárodní srozumitelnost, kterou upevnila přechodem na verzi 4.0. CC licence se skládá celkem ze 6 skupin, kde každá skupina má vlastní ikonu, která určuje jednotlivé podmínky pro nakládání s dílem. Seznam těchto zkratek a ikon viz Obr. 13.



Obr. 13 Creative Commons licence (zkratky)

5.4 Naplnění funkčních požadavků aplikace

Následující kapitola zopakuje, jaké byly kladené funkční požadavky na aplikaci a dále popíše, jak byly tyto požadavky splněny. Většina řešení těchto požadavků je pak dále vysvětlena v dalších kapitolách tohoto dokumentu.

1. Veřejně dostupný – Webová aplikace bude veřejně dostupná a ke stažení společně s dokumentací či postupy, jak s ní pracovat.

Tento požadavek byl splněn tak, že zdrojový kód aplikace je dostupný na gitovém uložišti (dostupné z: https://github.com/koudelkaMatej/NextDOM) a Docker Hubu (dostupné z: https://hub.docker.com/r/minikoudy/NextDOM2/tags).

2. Přehledný a optimalizovaný kód – Kód aplikace bude dodržovat zásady pro psaní přehledného kódu (odsazení, komentáře atd.). Vývojář se také pokusí optimalizovat či vylepšit již napsaný algoritmus nebo upravit řešení tak, aby odpovídalo vybrané technologii. Programátor vybere vhodný typ licence.

Požadavek na přehledný kód byl naplněn již při přepisu do programovacího jazyka Python, jelikož Python potřebuje pro správné fungování dodržovat správné odsazování.

Optimalizace kódu proběhla pouze ve výpočetní části a rozšíření o nové možnosti. Ty jsou dále popsáno v kapitole 9 Diskuse a kapitole 10 Závěr.

Aplikace musela být vzhledově a funkčně upravena, aby odpovídala použité webové technologii, která nemá přístup do adresářové struktury jako původní aplikace. Kód obsahuje komentáře, které blíže popisují jednotlivé části či proměnné. Aplikace byla rozdělena do menších funkcí, které usnadnily úpravy některých částí.

Licence, která byla po konzultaci s vedoucím a zadavatelkou práce vybrána, je MIT licence. Jedná se o jednu z nejméně restriktivních typů licence a splňuje požadavky, které byly na základě konzultací vytyčeny.

3. Virtualizované prostředí – Webová aplikace poběží v Dockeru, který usnadní nasazení na všech typech zařízení a dovede tím převést aplikaci do jakéhokoli operačního systému a také pro jakýkoli operační systém.

Tento požadavek byl splněn, avšak nebyl testován na všech platformách. Vzhledem k obsahu dokumentace a různých článku, by však tento požadavek měl být splněn v plné míře.

4. Interaktivní grafy – Webová aplikace bude obsahovat interaktivní grafy výsledků, které bude uživatel moci přizpůsobovat. To znamená přiblížit, posunout či zobrazit jen určitá data.

Tato část byla vyřešena pomocí knihovny Plotly, která je napsána v Pythonu a na webové stránce je vykreslována pomocí JavaScriptu. Graf je plně interaktivní a je možné si i výsledek stáhnout jako PNG obrázek.

5. Logovací data – Webová aplikace bude při běhu analýzy vytvářet log, který bude uživateli dávat zpětnou vazbu o průběhu analýzy.

Tento požadavek byl splněn, avšak jeho vznik byl až na samém závěru práce, kdy webová stránka nedávala žádnou zpětnou vazbu o tom, jak výpočty analýzy souborů probíhají. Tento soubor dává uživateli informace o průběhu, jako i o chybách, které mohli nastat v průběhu analýzy.

6. Výběr prahových hodnot – Aplikace počítá prahové hodnoty pro budoucí analýzu dat. Je potřeba zajistit, že se vypočtené hodnoty budou automaticky aplikovat pro analýzu dat či jiným způsobem zajistit, že bude možné tyto hodnoty v aplikaci měnit.

Přidání této možnosti nevyplývalo úplně z původní aplikace, jelikož se soubory potřebné k běhu vkládají ručně do složky v adresářovém průzkumníku. Tento požadavek byl vyřešen přidáním tlačítka pro nastavení vypočítaných prahových hodnot jako výchozí nastavení celé aplikace.

5.5 Naplnění technických požadavků aplikace

Kapitola zopakuje, jaké byly kladené technické požadavky na aplikaci a dále popíše, jak byly tyto požadavky splněny. Většina řešení těchto požadavků je pak dále vysvětlena v dalších kapitolách tohoto dokumentu.

1. Dokumentace – Vývojář vytvoří dokumentaci, jenž bude dodávána s aplikací. Dokumentace se bude soustředit hlavně na uživatelskou příručku popisující ovládání samotné aplikace, a také přípravu i nasazení vývojového prostředí.

Dokumentace a uživatelská příručka je blíže popsána v kapitole 7 Uživatelská dokumentace. Jako dokumentace může sloužit i tato diplomová práce či popis implementovaný přímo ve webové aplikaci.

2. Sběr souborů, dat a analýz – Aplikace si uchovává veškeré soubory, které do ní uživatel nahraje pouze po nezbytně nutnou dobu výpočtu, jelikož se jedná o zdravotnická data pacientů, které by aplikace neměla z bezpečnostních důvodů uchovávat.

Jelikož aplikace pracuje s daty pacientů, jež spadají do kategorie citlivých dat, je nutné tato data chránit. Z tohoto důvodu byl požadavek v průběhu práce lehce měněn a veškerá data pacientů, která uživatelé nahrají do webové aplikace může uživatel sám smazat kdykoli uzná za vhodné. Taktéž je nastavená plánovaná akce, která každých 24 hodin smaže veškerý nahraný obsah.

3. Výběr technologie – Vývojář si vybere programovací jazyk aplikace, který nebude zatížen licenčními poplatky a bude možné ho provozovat na libovolném operačním systému.

Výběru technologie se blíže věnuje podkapitola 5.3 Zvolené technologie. Požadavek na provoz na libovolném operačním systému byl splněn pomocí dockerizace a výběrem programovacího jazyka, který funguje na všech platformách.

4. Aktualizace – Vývojář zajistí, že bude možné aplikaci aktualizovat či upravovat její obsah a algoritmus.

Vzhledem k funkčnímu požadavku na tuto aplikaci, který požaduje veřejný kód, je možné pro uživatele si stáhnout funkční aplikaci a upravit jí na základě vlastních požadavků. To znamená, že uživatelé mají možnost měnit a aktualizovat veškerý obsah aplikace. Tyto úpravy však spadají do licenčního ujednání licence MIT a uživatel musí uvést jméno autora viz kapitola 5.3.6 Licence.

6 Implementace

Tato kapitola se zabývá vývojem aplikace a popisem, jak byly jednotlivé technologie využity a implementovány do nového řešení. Tato kapitola bude popisovat krok za krokem vývoj aplikace, navíc bude doplněna o ukázky aplikace, jež blíže popíšou výsledné řešení.

6.1 Webové rozhraní a design

V této části se předložená diplomová práce zabývá vývojem vzhledové části webové aplikace a popisu postupu při vývoji tohoto rozhraní. Webová aplikace se začala vyvíjet, jak již bylo zmíněno v kapitole 5 Návrh aplikace pomocí HTML a CSS.

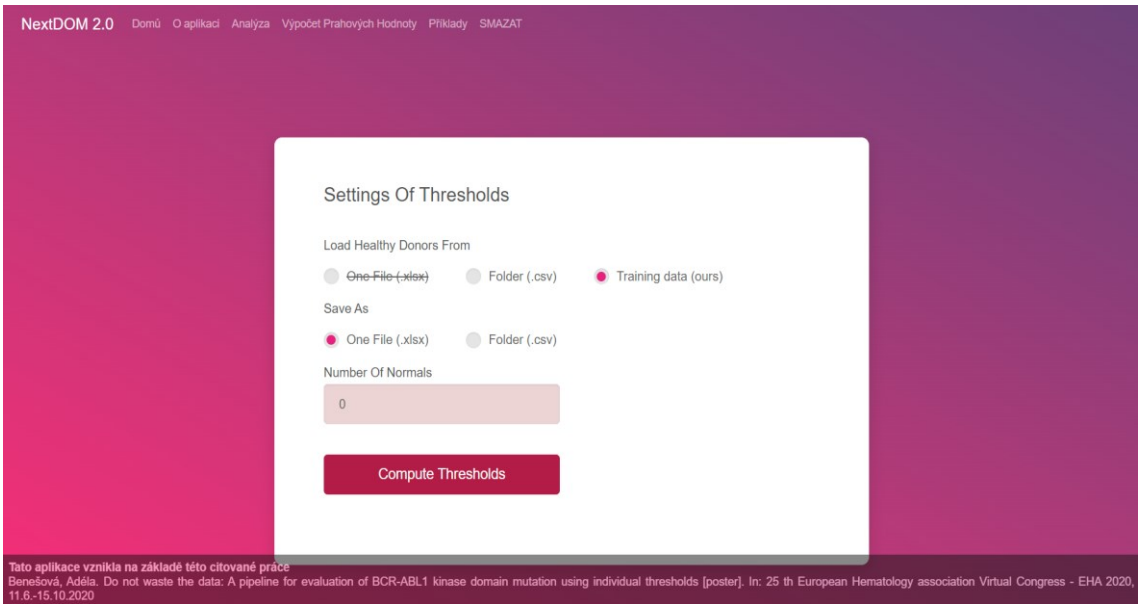
6.1.1 Výpočet prahových hodnot

První navrhnutá stránka, která se inspirovala vzhledem z Obr. 6 (str. 31) vypadala jako na Obr. 14. Tento návrh se však zadavateli práce nelíbil, a bylo tedy nutné vymyslet nový vzhled stránek, který bude splňovat modernější standardy pro tvorbu webových stránek. Nicméně na těchto stránkách bylo vyzkoušeno a otestováno základní předávání parametrů do výpočetní části aplikace.

Obr. 14 Ukázka prvního designu

Při vytváření druhého designu stránek byl využit Bootstrapový vzor, který využívá framework Bootstrap ve verzi 5, jenž vytvořil hlavní vzhled stránky. Do této šablony byl vytvářen dodatečný obsah, který se opět inspiroval prvky z Obr. 6 (str. 31). Vzhledem k tomu, že se jedná o webovou aplikaci, byly pro výpočet prahových hodnot vytvořeny dvě stránky. Důvodem tohoto rozdělení bylo oddělení grafu, který se měl vykreslit po výpočtu prahových hodnot a vytvoření stažitelného obsahu či nastavení nových prahových hodnot jako výchozí pro následnou analýzu.

Stránka pro výpočet prahových hodnot v současné době vypadá jako na Obr. 15 a oproti původní aplikaci obsahuje možnost použít tréninková data, která byla použita při vytváření a testování správnosti výsledků tohoto návrhu.

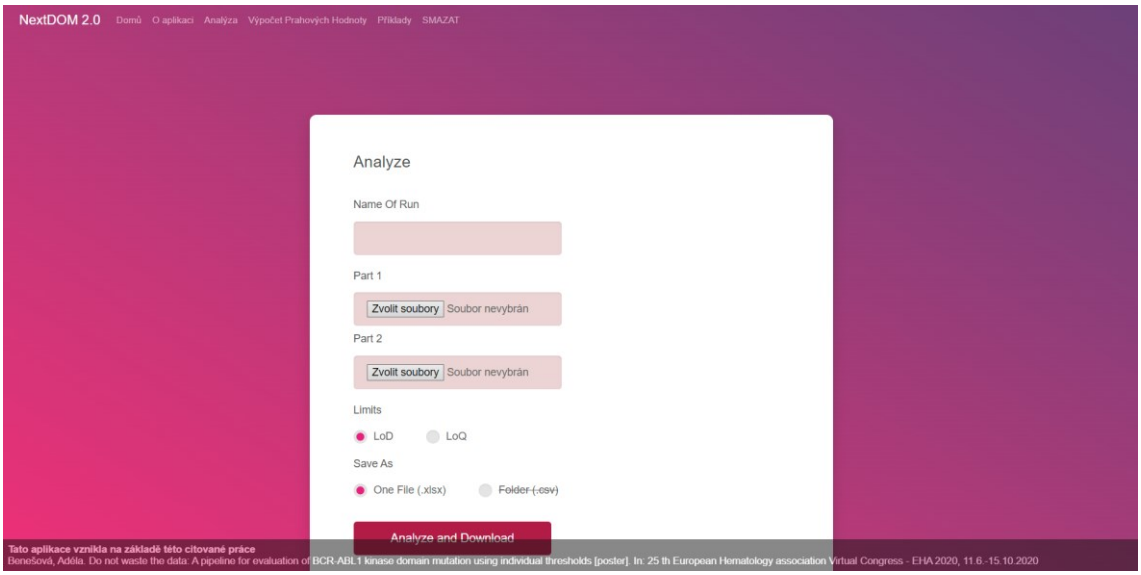


Obr. 15 Ukázka finálního návrhu pro výpočet prahových hodnot

Druhou stránku tvoří interaktivní graf, který je do stránky vložen pomocí JavaScriptu a který zobrazuje výsledky vypočtených prahových hodnot. Dalšími prvky jsou dvě tlačítka. První tlačítko slouží ke stažení výsledných hodnot a druhé tlačítko lze použít pro nastavení vypočtených prahových hodnot jako výchozí nastavení, které se bude používat pro budoucí analýzy.

6.1.2 Analýza dat

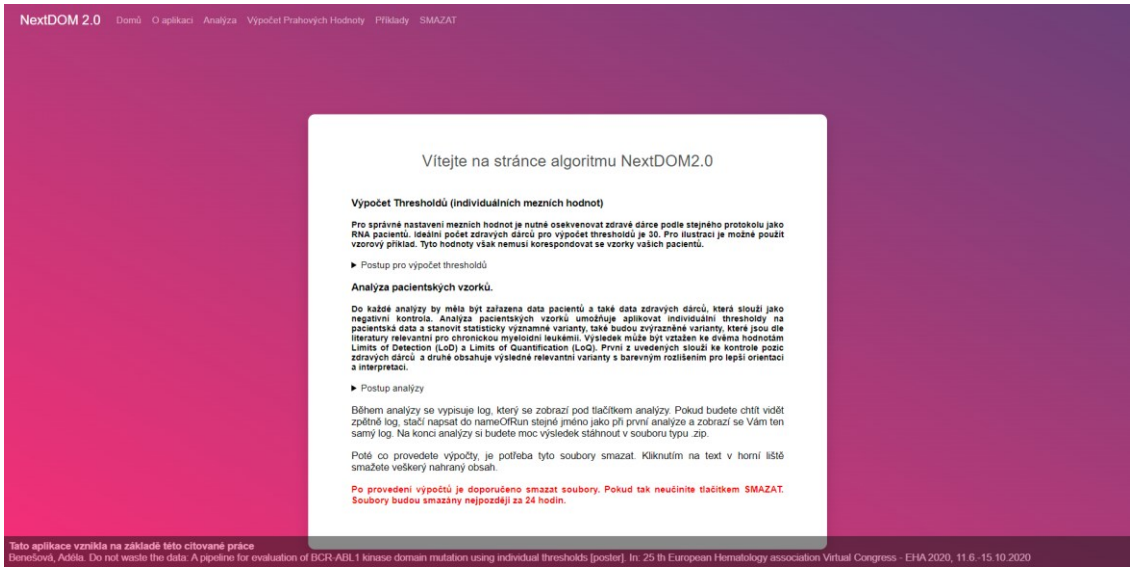
Design stránek pro analýzu dat odpovídá designu zbytku webové aplikace a prvky vložené na této stránce jsou stejné, jako prvky použité na Obr. 12 (str. 36). Vzhledem k tomu, že je složitější nahrávat do webového prostředí více složek, byla vstupní data rozdělena na jednotlivé části. Ty uživatel musí zadat samostatně, nikoliv tak, jak je tomu v původní aplikaci odkazem na složku, jež obsahuje obě části dat. Další odlišností oproti původní verzi je logovací proces. Ten si může uživatel zobrazit, pokud do názvu běhu zadá název již proběhlé analýzy či při prvním kliknutí na tlačítko „Analyze and Download“. Výpis logu probíhá pod tlačítkem a slouží jako ukazatel postupu analýzy. Pokud si uživatel bude přát použít i druhý limit, stačí zadat druhou možnost a analýzu pustit znovu. Ukázka designu je na Obr. 16.



Obr. 16 Ukázka finálního designu analýzy dat

6.1.3 Postup výpočtu

Aby uživatel věděl, jak postupovat pro správné fungování webové aplikace, byl vytvořen popis řešení, který obsahuje základní informace a definuje postup a správné formáty dat. Tento přehled je na titulní stránce aplikace či pod odkazy v horní části navigační lišty. Pokud si uživatel chce přečíst návod, stačí mu kliknout na odkaz „Domů“ a otevře se mu stránka, kterou vidíme na Obr. 17.



Obr. 17 Ukázka hlavní stránky

Titulní strana obsahuje návod pro výpočet prahových hodnot a analýzu, který jde zobrazit po rozkliknutí jednotlivých postupů. Dále obsahuje upozornění, které uživatele upozorňuje na to, že aplikace používá zdravotnická data, a je tudíž nutné tyto záznamy ze serveru vymazat. Pokud ovšem uživatel zapomene tato data odstranit, je upozorněn, že všechna data se automaticky smažou každých 24 hodin.

Pokud si uživatel chce prohlédnout vzhled a formát vstupních dat, je možné rozkliknout v navigační liště odkaz s názvem „Příklady“, který obsahuje obrázky vstupních souborů, jenž uživatelům ukazují vzhled dat.

6.1.4 Základní informace

Základní informace o stránce si uživatel může zobrazit na stránce „Domů“ a „O aplikaci“. Zde je popsán postup řešení výpočtu prahových hodnot i analýzy a základní informace o zkoumané nemoci a použití této aplikace.

6.1.5 Detekce chyb

Poslední částí je webová stránka, která vypisuje chybové hlášky, které většinou vychází ze špatných vstupních souborů. Pokud se chyba stane při výpočtu analýzy dat, je možné si podrobněji přečíst chybu v logovacím souboru, který si uživatel může zobrazit na stránce s analýzou. Tato chybová hláška bude v logu označená slovem „ERROR“.

6.2 Datová a aplikační vrstva

V následující podkapitole se podrobněji popisují logické funkce aplikace. Rozeberou se principy předávání parametrů a výpočtů jednotlivých funkcí. Zároveň je popsán základní princip propojení webové stránky s aplikační vrstvou, která využívá flaskové RESTové API, napsané v programovacím jazyce Pythonu.

6.2.1 Flaskové RESTové API

Aplikace je napsaná pomocí webového mikro-frameworku Flask. Typický způsob vykreslení webových služeb je pomocí jednotlivých komponent, ze kterých se skládá.

První z těchto komponent je cesta neboli sada URL adres, které vykonají akci, pokud na danou adresu přijde HTTP požadavek. Mezi HTTP požadavky se řadí požadavky:

- GET – pro získání dat ze serveru,
- POST – pro přidání nových dat na server,
- PUT – pro aktualizaci či vytvoření nových dat na serveru,
- DELETE – pro smazání dat ze serveru.

Každá taková adresa má tzv. „View funkci“, která tyto požadavky zpracovává. Tyto funkce mohou být složité, ale také mohou pouze odkazovat na stránku, která se má

vykreslit. Mohou obsahovat i vstupní parametry nebo do odkazované stránky posílat odchozí informace.

Zároveň obsahuje chybové odpovědi či stavy, které se uživateli mohou zobrazovat, a které může taktéž sám nastavit. Při ladění aplikace jsou tyto stavy vidět například při přechodech na jiné stránky či při jednotlivých výpočtech funkcí.

Pro lepší komunikaci mezi jednotlivými soubory byla také vytvořena adresářová struktura, která obsahuje veškeré soubory, které aplikace využívá pro výpočty a vykreslování HTML souborů.

- Složka Templates – Obsahuje HTML soubory, které se vykreslují ve webovém prohlížeči.
- Složka Static – Obsahuje soubory, které uživatel nahrává či jsou důležité pro správnou funkci HTML stránek nebo analýzy. Tato složka je dále rozdělena na:
  - DefaultCsv – obsahuje základní soubory, tréninková data atd.,
  - Img – obsahuje obrázky použité na stránce Examples,
  - Styles – obsahuje CSS soubory, které určují vzhled HTML stránek,
  - UploadedFiles – obsahuje veškerý nahraný obsah uživatelem.

6.2.2 Zobrazování webového rozhraní

Zobrazení webového rozhraní probíhá přechodem na adresu, která je definovaná v seznamu flaskových URL adres. Ty v tomto programu obsahují i jiné výpočty, které si popíšeme v další části této kapitoly, avšak pro tuto podkapitolu nejsou důležité. Ve chvíli, kdy View funkce odkáže na HTML soubor, který se má vykreslit ve webovém prohlížeči. Poté se zobrazí webová stránka, která je uložená ve složce na serveru. Aby se HTML soubory vykreslovaly i pomocí CSS stylů bylo potřeba CSS soubor vložit do složky na serveru, na který HTML soubor odkazuje.

6.2.3 Předávání parametrů

Aby mohla aplikační vrstva počítat s daty pacientů, které do aplikace zadává uživatel, je potřeba je z webové stránky předat do aplikační logické vrstvy. Tento proces probíhá pomocí formulářových oken, které se poté odesílají tlačítkem na webové HTML stránce.

Tento formulář obsahuje jednotlivé prvky, které mají svůj unikátní název. Pomocí formuláře lze přistupovat k informacím v nich vložených a předat je aplikační vrstvě. Předávání těchto informací probíhá po kliknutí na tlačítko, které spustí akci, jenž předá všechny tyto informace jiné webové stránce neboli jiné URL adrese. View funkce poté většinou nejprve načtou údaje uložené v jednotlivých prvcích a poté s nimi provedou výpočet. Poté načte webovou stránku, která také může přijímat vstupní parametry z flaskové aplikace. Pokud webová stránka se vstupními parametry pracuje, je možné

například vykreslit graf vypočtených hodnot či vypsání chybové hlášky, které této stránce funkce předala.

6.2.4 Výpočet prahových hodnot

V následující podkapitole je popsáno, jak se předávají data a jak probíhá výpočet prahových hodnot v rámci zpracované aplikace.

Jak již bylo možné vidět na Obr. 15 (str. 51), View funkci se budou předávat tři parametry a to:

- Healthy Donors – data z analýzy zdravých dárců,
- Save as – formát dat, které chceme jako výstup,
- Number of Normals – údaj o počtu souborů, které chceme použít.

Poté, co kliknete na tlačítko „Compute Thresholds“, se spustí funkce pro výpočet prahových funkcí. Tato funkce si uloží do proměnných jednotlivé údaje, které uživatel zadal do proměnných a na základě toho, jaký formát dat jsme vložili, se zvolí další krok. Pokud uživatel zvolil tréninková data, tak se uživateli nastaví vstupní soubory na hodnoty z analýzy NextGENe software od firmy Softgenetics z USA, jenž jsou uloženy ve složce DefaultCsv. Všechny soubory se budou ukládat do složky uploadedFiles, která byla za tímto účelem vytvořena. Poté nás tato funkce přesměruje na jinou URL adresu, která obsahuje všechny vstupní parametry z předchozí stránky.

Tato nová stránka obsahuje již algoritmus výpočtu. Stránka na základě toho, jak chceme data uložit, vybere postup výpočtu prahových hodnot. Po výpočtu prahových hodnot se vytvoří soubor či soubory, které se uloží do složky pro výsledky prahových hodnot. Poté se z výsledků vezmou soubory, které nám pomohou vykreslit graf a příslušné informace předáme jako vstupní parametr na vykreslení nové HTML stránky. Ta nám vykreslí graf tranzicí a tranzverzí pro náš výsledný soubor prahových hodnot.

Tento graf, který můžeme vidět na Obr. 18, je interaktivní a uživatel si může jednotlivé body přiblížit, oddálit či vypnout jeden druh výsledků. Výsledky výpočtu si může uživatel stáhnout kliknutím na tlačítko, které odkazuje na View funkci. Ta pouze vrací zipový soubor obsahující všechny výsledky výpočtu prahových hodnot. Také může výsledné hodnoty aplikovat jako výchozí hodnotu pro všechny další analýzy, které bude v budoucnu zpracovávat.

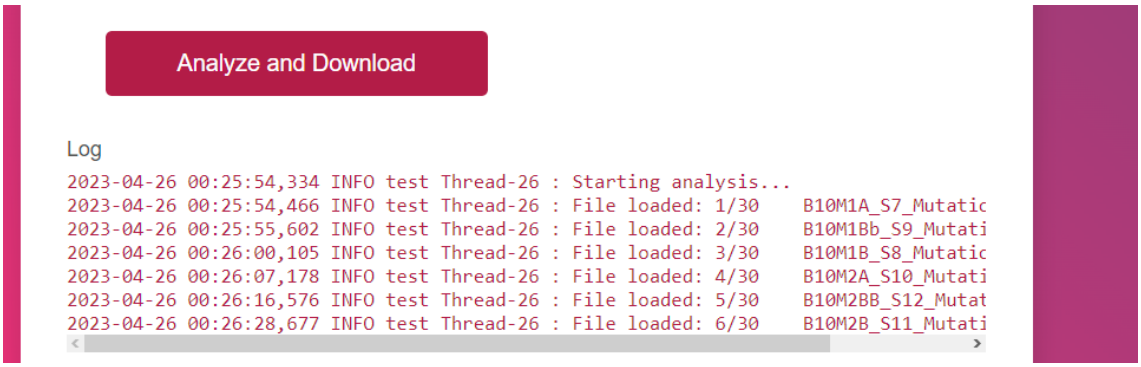
Obr. 18 Výsledný graf prahových hodnot

6.2.5 Analýza

V této podkapitole, je představeno předávání jednotlivých parametrů a popis analýzy dat na základě vypočtené prahové hodnoty.

Jak je již popsáno na Obr. 16 (str. 52), vstupními parametry bude název běhu, soubory obsažené v Part 1 a v Part 2, výběr limitů a způsob uložení dat. Poté, co uživatel zadá tyto údaje, spustí analýzu tlačítkem „Analyze and Download“.

Stejně jako při výpočtu prahových hodnot si View funkce načte vstupní parametry do proměnných a na základě vstupních hodnot uloží vstupní soubory do složky „Analyze“. Poté se založí logger, který nám bude ukládat do logovacího souboru postup analýzy. Na základě způsobu uložení dat si algoritmus vybere možnost výpočtu algoritmu a logger začne vypisovat průběh analýzy. Jakmile analýza skončí, vytvoří se soubor zip, jenž obsahuje všechny soubory, které analýza vytvořila. Následně se uživateli zobrazí dialogové okno s tím, kam si přeje tento soubor uložit. Průběh této akce si můžeme prohlédnout na Obr. 19.



Obr. 19 Ukázka logovacího souboru

Pokud v průběhu analýzy dojde k chybě, uživateli se otevře nová URL adresa, která bude obsahovat chybovou hlášku. Pokud by si uživatel přál vidět další podrobnosti, může se vrátit na stránku s analýzou a zadat do názvu běhu stejný název, jako při nezdařeném pokusu. Jestliže tak učiní, zobrazí se mu pod tlačítkem Log, který bude obsahovat logovací hlášky, které jsou obvykle v kategorii INFO a chybové hlášky v kategorii ERROR.

6.2.6 Reset aplikace

Jelikož aplikace obsahuje nahraná zdravotnická data a tato data spadají do kategorie citlivých údajů, je potřeba je chránit. Abychom nemuseli řešit ochranu těchto dat v rámci naší aplikace, je potřeba mazat nahrané soubory na serveru. Tato povinnost je přenesena na uživatele, který je o této skutečnosti informován v postupu analýzy. Pokud by se však stalo, že by uživatel na tuto akci zapomněl, je na serveru aplikována plánovaná aktivita, která každých 24 hodin smaže všechna nahraná data uživatelů.

6.3 Dockerizace a sdílení

Výslednou aplikaci je potřeba připravit pro nasazení na různá operační prostředí a je třeba otestovat funkčnost aplikace v kontejneru. V podkapitole je popsán celý proces dockerizace a implementace nového řešení pro nová zařízení.

Dockerizace je proces, při kterém se aplikace a všechny knihovny zabalí do kontejneru. Kontejner je izolované prostředí, které obsahuje všechno potřebné pro správný chod aplikace. Pro správnou dockerizaci je potřeba vytvořit Dockerfile, což je textový soubor, který obsahuje instrukce jako je např. platforma či operační systém. To určí prostředí, na kterém kontejner poběží, povolené porty, soubory nebo třeba příkazy v příkazové řádce, které se mají při spouštění kontejneru spustit. Tyto instrukce vlastně určují obsah a závislosti kontejneru.

Celý tento proces vytvoření izolovaného prostředí, které zajistí jednoduchý přenos aplikace na jiné operační systémy či cloudové služby, eliminuje potíže, jež by mohly nastat při přesunu mezi systémy. Jedním z problémů, který přišel při dockerizaci prostředí práce bylo použití Windows knihovny, která v Dockeru nefungovala z důvodu, že Docker využívá linuxového základu, a tudíž tato knihovna neexistuje v prostředí Dockeru.

6.3.1 Příprava a spuštění kontejneru

Pro práci s Dockerem je potřeba stáhnout si aplikaci Docker, která je dostupná z oficiálních stránek společnosti a je dostupná na Windows, MacOS a různé Linuxové distribuce. Pro správné fungování aplikace je potřeba mít na základní desce povolenou virtualizaci. Toto nastavení lze povolit v BIOSu prakticky každé základní desky.

Po nainstalování Dockeru je potřeba tuto aplikaci spustit. Poté co je nainstalováno a spuštěno, je možné vytvořit svůj kontejner, který bude obsahovat vytvořené řešení. Jak již bylo zmíněno v předchozí podkapitole, je potřeba vytvořit Dockerfile.

Text napsaný v Dockerfilu je na Obr. 20 a jednotlivé části v souboru jsou:

- určení prostředí příkazem FROM,
- nastavení výchozí složky WORKDIR,
- zkopírování souboru requirements.txt do WORKDIR, který obsahuje všechny knihovny potřebné k běhu flaskové aplikace,
- instalace všeho v souboru requirements.txt pomocí RUN,
- příkazem COPY zkopírování všeho ve složce s projektem do složky WORKDIR v kontejneru,
- otevření portu 5000, na kterém běží flasková aplikace pomocí EXPOSE,
- příkaz CMD, který při spuštění kontejneru provede zapnutí flaskové aplikace.

```
# Use the official Python base image
FROM python:3.9.5
# Set the working directory in the container
WORKDIR /
# Copy the requirements.txt file into the container
COPY requirements.txt .
# Install the dependencies
RUN pip install --no-cache-dir -r requirements.txt
# Copy the rest of the application code into the container
COPY . .
# Expose the port that the app will run on
EXPOSE 5000
# Start the Flask app
CMD ["python", "app.py"]
```

Obr. 20 Dockerfile

Pro vytvoření kontejneru je nyní potřeba otevřít příkazovou řádku ideálně do kořenové složky našeho projektu, která obsahuje náš Dockerfile. Ve Windows stačí kliknout ve složce pravým tlačítkem a dát „Otevřít v aplikaci Terminal“. Poté co otevřeme příkazovou řádku zadáme příkaz pro sestavení kontejneru.

$$docker\ build\ -t\ nextdom\ .$$

Tento proces bude vypisovat postup, který Docker při buildu kontejneru dělá a tento postup je i napsaný v Dockerfilu. Následně je nutné tento kontejner spustit. Pro spuštění se musí zadat příkaz, který obvykle vypadá následovně:

$$docker\ run\ nextdom$$

Problémem tohoto příkazu při využití Flasku je ten, že takto spuštěný kontejner nemá přístup k portu, jenž Flask využívá. Proto je potřeba tento příkaz upravit tak, aby povoloval přístup k portu, na kterém Flasková služba běží.

$$docker\ run\ -p\ 5000{:}\ 5000\ nextdom$$

V tuto chvíli se nám v příkazové řádce vypíše, že Flask server běží a vypíše se adresa, na které je úvodní stránka webové aplikace. Ta běží na localhostové adrese, která je adresou stroje, na kterém je tento kontejner spuštěn. To znamená, že pro ostatní zařízení, je tento server dostupný na IP adrese tohoto stroje na portu 5 000. Výchozí adresa pro tento kontejner je tedy http://127.0.0.1:5000.

6.3.2 Sdílení kontejneru

Pro snadnější předávání hotového řešení je vhodné přidat hotové řešení ve formě Docker obrazu. Docker obraz je samostatný spustitelný balíček. V tomto řešení byly zvoleny dva způsoby, jak tento projekt předat.

Docker Hub

Prvním z těchto řešení je využití Docker Hubu, což je repositář, jenž povoluje ukládání, správu a sdílení Docker obrazů. Pro to, aby si mohli uživatelé vytvořený kontejner stáhnout, stačí uživatelům poskytnout příkaz ke stažení či odkaz do repositáře.

https://hub.docker.com/r/minikoudy/NextDOM2/tags

Pro vložení řešení této práce na tento server je nejprve nutné se po vytvoření kontejneru přihlásit pomocí příkazu.

$$docker\ login$$

Poté je nutné přidat tomuto kontejneru tzv. tag, který nám označí nějakým slovem náš obraz. Příkladem, takového tagu může být třeba číslo verze programu. V příkazu níže je vidět, že uživatelské jméno řešitelova účtu je minikoudy.

$$docker\ tag\ nextdom\ minikoudy/nextdom{:}\ tagname$$

Následně je nutné toto řešení poslat do repositáře a zkontrolovat, zda má výsledný obraz příslušná práva pro stažení z repositáře.

$$docker\ push\ minikoudy/nextdom{:}\ tagname$$

Ve chvíli, kdy máme zkontrolováno, že se do repositáře nahrála naše nejaktuálnější verze kontejneru, je možné vystavit příkaz pro stažení a pro užití jinou osobou.

$$docker\ pull\ minikoudy/nextdom:latest$$

Pokud si uživatel stáhne pomocí příkazu výše tento kontejner, může si ho pustit podobným způsobem, jakým je spuštěn kontejner v podkapitole 6.3.1 Příprava a spuštění kontejneru.

Git

Druhým z těchto řešení je poskytnutí Dockerfilu spolu s kódem aplikace. Tím dokážeme umožnit ostatním uživatelům sestavit si vlastní kontejner, na základě Dockerfilu, který bude přístupný spolu se soubory projektu. Toto řešení nám dovolí jednoduše vyřešit otázku aktualizace či úpravy aplikace.

Pro stažení výsledku z gitového repositáře existuje více možností. Například příkazová řádka nebo může využít odkaz na gitové uložiště a projekt stáhnout přímo z webového prohlížeče. Pro stažení řešení této práce stačí, aby uživatel zadal stránku řešitelova gitového repositáře do webového prohlížeče a v pravé části gitového rozhraní si může celý repositář stáhnout.

https://github.com/koudlkaMatej/NextDOM

7 Uživatelská dokumentace

Tato kapitola popisuje postup a požadavky při nasazování v různých prostředí. V této kapitole je i popsána práce se samotnou aplikací jakož i její nasazení a aktualizace.

7.1 Nasazení, spouštění a aktualizace

Nasazení bylo testováno pouze na platformě Windows, avšak pro ostatní operační systémy by implementace řešení této práce nemělo představovat výrazné problémy. Jedná se o projekt, který využívá Docker, jenž by měl působit jako izolované a bezpečné prostředí.

7.1.1 Prerekvizity nutné k správné funkci

Pro testování, na operačním systému Windows 11, bylo potřeba nainstalovat prerekvizity a příslušné komponenty pro správné fungování celého procesu. Prerekvizity a příslušné komponenty jsou popsány níže.

- BIOS – Je potřeba povolit virtualizaci na základní desce počítače. Každá základní deska má tuto možnost jinde, a proto je nutné si nastavení dohledat pro každý stroj individuálně.
- WSL 2 (Windows Subsystem for Linux) – Tato technologie je emulace jádra Linuxu, která běží pod OS Windows. To nám dovoluje spouštět a provozovat linuxové distribuce přímo na systému Windows.
  (dostupné z: https://docs.microsoft.com/en-us/windows/wsl/installwin10)
- Docker Desktop – Tento nástroj, umožňuje snadné vytváření, správu a spuštění Docker kontejnerů a je potřebný k vytvoření izolovaného prostředí. (dostupné z: https://docs.docker.com/docker-forwindows/install/)
- Verzovací systém GIT – Tento program nám dovolí využívat příkazovou řádku pro komunikaci s verzovacím systémem git.
  (dostupné z: https://git-scm.com/downloads)
- Samotné řešení – Toto řešení je předmětem této práce.
  (dostupné z: https://github.com/koudelkaMatej/NextDOM)
  - Python 3 – Jedná se o programovací jazyk a interpreter, kterým je napsána tato práce. Pro nejsnadnější komptabilitu, je vhodné využít Python ve verzi 3.9.5, který jsem při vývoji využil, ale mělo by být možné použití i vyšší verze. (dostupné z: https://www.python.org/downloads/)
- Visual Studio Code – Tato aplikace je textový editor, který je navržen pro efektivní vývoj softwaru a podporuje mnoho programovacích jazyků. Vyniká hlavně svojí jednoduchou rozšiřitelností a lehkostí. Po nainstalování je potřeba doinstalovat do této aplikace Python, pro komunikaci s interpretem.
  (dostupné z: https://code.visualstudio.com)

7.1.2 Úprava řešení

Následující podkapitola je zaměřena na postup, který nám umožní upravovat existující řešení na zařízeních, jenž doposud nemělo žádnou předchozí zkušenost s daným řešením. Taktéž zde najdete informace o správném nastavení a postupu instalace vývojového prostředí, aby bylo dosaženo stejných podmínek, které byly při práci použity.

Postup instalace

Po nastavení a instalaci výše zmíněných důležitých částí implementace je možné překročit k úpravě staženého řešení. Vzhledem k tomu, že máme již nainstalovaný program pro komunikaci s gitovým verzovacím systémem, lze použít příkaz pro naklonování celého repositáře. Ten použijeme ve složce, kterou si určíme, že bude složkou projektu.

$$git\ clone\ \text{https://github.com/koudelkaMatej/NextDOM.git}$$

Při otevření adresářového prohlížeče na právě naklonovaném řešení, je možné tuto složku otevřít v aplikaci Visual Studio Code. Zde pak stačí otevřít příkazovou řádku ve spodní liště editoru, případně pomocí možnosti View v horní liště, vybrat Terminal CMD nebo PowerShell a zadat následující příkaz.

$$python - m\ venv\ venv$$

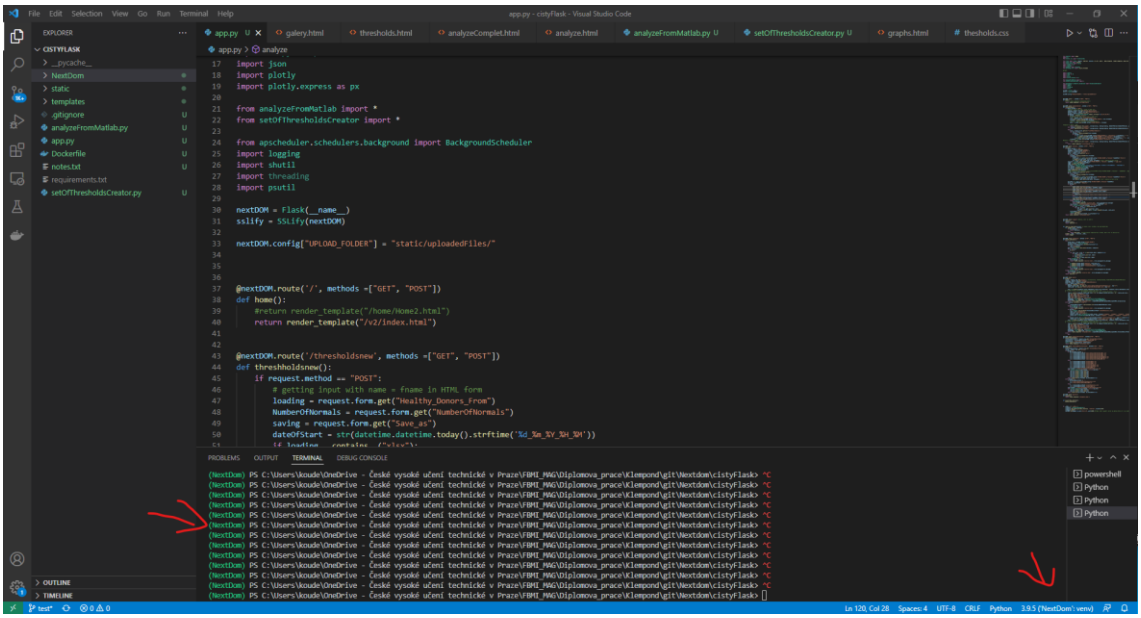
Poté co proběhne vytvoření virtuálního prostředí do složky projektu, stačí toto virtuální prostředí aktivovat. Pro aktivaci virtuálního prostředí musíte zadat následující příkaz.

$$.\backslash venv\backslash Scripts\backslash activate.bat$$

Případně je možné, pokud jste správně nainstalovali Python do Visual Studio Code, pouze kliknout do levé části projektu v části Explorer pravým tlačítkem a dát „Open In Integrated Terminal“.

Poté se Vám ve spodní části otevře příkazová řádka Pythonu a vytvořené virtuální prostředí, do něhož budete instalovat všechny knihovny potřebné k práci se všemi komponentami předložené práce.

O tom, že je naše virtuální prostředí aktivní se můžeme přesvědčit dvěma způsoby viz Obr. 21. Prvním způsobem je, že na začátku řádky uvidíme v zelené závorce název našeho virtuálního prostředí. Druhá možnost je, že v pravé dolní části bude zvolen náš interpreter.



Obr. 21 Kontrola otevření virtuálního prostředí

K instalaci všech knihoven, jež jsou uloženy v souboru requirements.txt, bude sloužit příkaz, který již bude psán do příkazové řádky virtuálního prostředí.

$$pip\ install\ -r\ requirements.txt$$

Pokud vše proběhlo, jak mělo, je nyní možné otevřít soubor app.py a v levé horní části dát spustit. Případně lze program spustit ještě tak, že uživatel dá v levé části obrazovky tlačítko „Run and Debug“ či klávesovou zkratku „CTRL + ALT + D“ a poté vybere možnost „Run and Debug“, kde vybere možnost "Flask“.

Orientace v důležitých částech kódu

Aby se uživatel dobře orientoval v kódu, je potřeba mu vysvětlit základní strukturu a pohyb mezi soubory webového rozhraní a výpočetní částí.

Adresářová struktura

Prvně je vysvětlována adresářová struktura projektu. Flasková neboli výpočetní část je vložena přímo v kořenovém adresáři projektu a jedná se o soubory:

- app.py – samotná flasková aplikace,
- setOfThresholdsCreator.py – obsahuje algoritmus pro výpočet prahových hodnot,
- analyzeFromMATLAB.py – obsahuje algoritmus pro analýzu dat.

HTML část neboli soubory, které obsahují informace o tom, co se nám bude vykreslovat ve webovém prohlížeči je uložena ve složce „templates/v2“.

Složka static slouží pro běh aplikace a obsahuje potřebné soubory k běhu algoritmů, ale také soubory, které upřesňují vzhled webových stránek na základě CSS souborů. Dále

obsahuje soubory, jenž se mají vykreslovat na webové stránce, či soubory nahrané uživatelem.

Orientace ve výpočetní části programu

Dále je vysvětlení, jak je definována výpočetní (flasková) část aplikace. Ta má vždy definovanou cestu neboli URL adresu (route). Pokud uživatel ve webovém rozhraní zadá tuto adresu, vykoná se funkce, která je vložená pod ní. V našem případě je to adresa http://127.0.0.1:5000/ a vykoná se funkce home(), která nám vykreslí HTML stránku uloženou ve složce „static/v2/index.html“. Ukázku jednoduché URL adresy vidíme na Obr. 22.

```
@nextDOM.route('/', methods =["GET", "POST"])
def home():
    return render_template("/v2/index.html")
```

Obr. 22 Ukázka jednoduché URL adresy

Pokud by stránky měli mít některý vstupní parametr, je možné se na ně zeptat pomocí následujících příkazů, viz Obr. 23.

```
saving = request.form.get("saveAs") #načte informace vloženy do fomruláře z HTML stránky
nameOfRun = request.form['NameOfRun']
limits = request.form['Limits']
part1 = request.files.getlist('part1') #načte všechny soubory vloženy do fomruláře z HTML stránky
```

Obr. 23 Příklad načítání vstupních dat

Pokud stránka obsahuje výstupní údaje je možné je předávat jako parametr při vykreslování HTML stránek. Viz Obr. 24, kde se předávají informace o chybové hlášce.

```
except Exception as e:  #Vypsání chybové hlášky na HTML stránce
    error_message = str(e)
    return render_template('/v2/error.html', error_message=error_message)
```

Obr. 24 Ukázka výstupních dat

Orientace v HTML souborech

Základní odlišností oproti klasickému HTML je ta, že odkazy, které obsahují HTML stránky musí odkazovat na URL adresy uvedené ve flaskové aplikaci. To je například námi zmíněná adresa http://127.0.0.1:5000/. Další odlišností je např. odkazování na CSS soubor, který je uložen ve složce „static/styles“. Toto si můžeme prohlédnout na Obr. 25.

```
<link rel="stylesheet" type="text/css" href="{{ url_for('static',filename='styles/styleV2.css') }}" />
```

Obr. 25 Ukázka odkazu pro nastavení stylů HTML stránky

Pokud webová stránka obsahuje vstupní parametry, které jí předal Flask, je možné je zobrazit jako na ukázce níže, kde si třeba vykreslíme chybovou hlášku z Obr. 26.

```
<div class="card-body" >
    <h1 style="text-align:center;color: #b31c47;font-weight: bold;">Vyskytla se chyba: {{ error_message }}</h1>
</div>
```

Obr. 26 Ukázka výpisu chybové hlášky

Oproti nutnosti zadávat do odkazů na jiné stránky celé URL adresy, formuláře vyžadují odlišný přístup. Do formulářů se nepíše jako odkaz URL adresa, ale název funkce, která se pod touto adresou nachází na Obr. 27, je vidět název funkce i vstupní parametr do této funkce.

```
<form action="{{ url_for('graph', thresholdResultsPath=thresholdResultsPath, saving=saving) }}"method="post">
```

Obr. 27 Ukázka odkazu pro webový formulář

Na některých webových stránkách je napsán kód i v JavaScriptu. Důvodem použití JavaScriptu bylo jednodušší použití než přepsání již napsaných funkcí v Pythonu.

7.1.3 Dockerizace řešení

Poté, co uživatel upraví řešení na základě jeho požadavků, může výslednou aplikaci zdokerizovat pro jednodušší a izolovanější běh aplikace. K tomu, aby toto mohl zvládnout je potřeba nainstalovat WSL 2, povolit virtualizaci v BIOSu základní desky a nainstalovat Docker Hub. Poté může uživatel postupovat dle popisu v kapitole 6.3 Dockerizace a sdílení.

7.2 Práce s webovou aplikací

Webová aplikace předpokládá, že budou zadávány pouze soubory, které daná aplikace očekává. To znamená, že musí být vkládány správné formáty souborů, a to mimo jiné i se správnými oddělovači. Jaké formáty dat a jak postupovat při práci s webovou aplikací je popsáno na hlavní webové stránce projektu, kde si uživatel může jednotlivé kroky přečíst. Dále je tam přidaná stránka pojmenovaná „Příklady“, obsahující grafické obrázky, které obsahují vstupní data. Každý obrázek byl vložen do tabulkového procesoru Excel, pro lepší vzhled dat. U každého obrázku je popis toho, jak mají vstupní data vypadat, pokud se bude jednat o soubor typu .csv. Každý obrázek je možné rozkliknout pro zvětšení. Tento zvětšený obrázek se uživateli zobrazí v nové záložce. Zároveň si tyto soubory může stáhnout na hlavní stránce v postupu analýzy.

7.3 Seznam dostupných adres

Níže bude uveden seznam všech URL adres, které jsou na serveru přístupné, a které může uživatel navštívit.

- http://127.0.0.1:5000/ – Toto je domovská stránka aplikace, která obsahuje postup práce s řešením této aplikace.
- http://127.0.0.1:5000/about – Tato stránka obsahuje informace o důvodu vzniku aplikace a co je cílem tohoto projektu.
- http://127.0.0.1:5000/thresholdsnew – Toto je stránka pro výpočet mezních hodnot pro následnou analýzu.
- http://127.0.0.1:5000/analyze – Stránka obsahující rozhraní pro analýzu dat.
- http://127.0.0.1:5000/plot – Stránka, která vykresluje graf výpočtu prahových hodnot. Na tuto stránku se lze dostat po spuštění výpočtu mezních hodnot.
- http://127.0.0.1:5000/treningData – Tato stránka je opět uživateli nedostupná a slouží pouze jako odkaz pro stažení 30 trénovacích dat dostupný z domovské stránky.
- http://127.0.0.1:5000/defaultThresholSet – Stránka opět nedostupná uživateli, jelikož se spouští tlačítkem „Set these thresholds as default“ na stránce plot. Po kliknutí se nastaví jako výchozí možnost a odkážou na stránku s analýzou.
- http://127.0.0.1:5000/endofwork – Stránka, na kterou se uživatel nedostane, jelikož přesměruje uživatele na domovskou stránku poté, co vymaže nahraný obsah.
- http://127.0.0.1:5000/downloadDefaultThreshold – Tato stránka slouží pouze pro stažení vypočtených prahových hodnot, která vrací soubor typu .zip.
- http://127.0.0.1:5000/gallery – Stránka slouží k zobrazení příkladů vstupních dat, která je zobrazuje formou obrázků.
- http://127.0.0.1:5000/error – Tato stránka se uživateli objeví pouze v případě, kdy se v aplikaci stane nějaká chyba. Většina chyb je způsobená chybnými vstupními daty, avšak některé jiné chyby jsou odchyceny a vypsány zde.

V neposlední řadě, je na serveru nastavená plánovaná akce, která každých 24 hodin zavolá funkci my_periodic_function(), která spouští funkci delete_subfolders(). Takto se jmenuje view funkce pro stránku endofwork, a tudíž maže obsah všech nahraných dat.

8 Výsledky

V následující kapitole jsou, jak vyplývá z názvu, popsané výsledky, kterých dosahuje řešení, které bylo předmětem této práce.

Jednotlivé hodnoty, které algoritmus počítal na trénovacích datech vycházely naprosto totožně, a to jak v původní aplikaci, tak i v řešení této práce. Jediným rozdílem mezi těmito hodnotami byl počet, na které programovací jazyk zarovnával desetinné místo. Zatímco MATLAB použil 4 desetinná místa, Python použil 15. Z toho plyne, že Python počítá s větší přesností.

Přestože Python dosáhl přesnějších výsledků, jeho výpočetní rychlost je výrazně pomalejší. Testování výpočtu prahových hodnot probíhalo na 7 souborech, které obsahovaly jak zdravé, tak i nemocné pacienty pro lepší validaci prahů. Při výpočtu prahových hodnot se rychlost výpočtu prakticky nelišila, avšak při analýze dat je rozdíl již znatelný, jak je vidět na Obr. 28.

	MATLAB				Python				Limit	Rozdíl (MATLAB - Python)			
	Part1	Part2	Part3	Celkem	Part1	Part2	Part3	Celkem		Part1	Part2	Part3	Celkem
	0:01:18	0:00:11	0:00:20	0:01:49	0:05:19	0:00:13	0:01:05	0:06:37	LoD	0:04:01	0:00:02	0:00:45	0:04:48
	0:01:10	0:00:12	0:00:20	0:01:42	0:05:23	0:00:14	0:01:23	0:07:00	LoD	0:04:13	0:00:02	0:01:03	0:05:18
	0:01:05	0:00:11	0:00:19	0:01:35	0:05:27	0:00:13	0:01:06	0:06:46	LoD	0:04:22	0:00:02	0:00:47	0:05:11
	0:00:57	0:00:11	0:00:19	0:01:27	0:05:20	0:00:14	0:01:27	0:07:01	LoD	0:04:23	0:00:03	0:01:08	0:05:34
	0:01:01	0:00:11	0:00:18	0:01:30	0:05:23	0:00:13	0:01:25	0:07:01	LoD	0:04:22	0:00:02	0:01:07	0:05:31
	0:01:07	0:00:11	0:00:19	0:01:37	0:05:19	0:00:14	0:01:06	0:06:39	LoD	0:04:12	0:00:03	0:00:47	0:05:02
	0:01:05	0:00:11	0:00:18	0:01:34	0:05:22	0:00:15	0:01:08	0:06:45	LoD	0:04:17	0:00:04	0:00:50	0:05:11
	0:00:57	0:00:11	0:00:20	0:01:28	0:05:24	0:00:15	0:01:08	0:06:47	LoD	0:04:27	0:00:04	0:00:48	0:05:19
	0:01:04	0:00:12	0:00:20	0:01:36	0:05:19	0:00:13	0:01:25	0:06:57	LoD	0:04:15	0:00:01	0:01:05	0:05:21
	0:01:06	0:00:11	0:00:19	0:01:36	0:05:26	0:00:14	0:01:08	0:06:48	LoD	0:04:20	0:00:03	0:00:49	0:05:12
Průměr	0:01:05	0:00:11	0:00:19	0:01:35	0:05:22	0:00:14	0:01:14	0:06:50		0:04:17	0:00:03	0:00:55	0:05:15
Min	0:00:57	0:00:11	0:00:18	0:01:27	0:05:19	0:00:13	0:01:05	0:06:37		0:04:01	0:00:01	0:00:45	0:04:48
Max	0:01:18	0:00:12	0:00:20	0:01:49	0:05:27	0:00:15	0:01:27	0:07:01		0:04:27	0:00:04	0:01:08	0:05:34

	MATLAB				Python				Limit	Rozdíl (MATLAB - Python)			
	Part1	Part2	Part3	Celkem	Part1	Part2	Part3	Celkem		Part1	Part2	Part3	Celkem
	0:00:58	0:00:10	0:00:31	0:01:39	0:06:06	0:00:16	0:01:23	0:07:45	LoD	0:05:08	0:00:06	0:00:52	0:06:06
	0:00:58	0:00:11	0:00:27	0:01:36	0:07:46	0:00:23	0:04:19	0:12:28	LoQ	0:06:48	0:00:12	0:03:52	0:10:52
	0:00:58	0:00:11	0:00:31	0:01:40	0:06:14	0:00:14	0:01:22	0:07:50	LoQ	0:05:16	0:00:03	0:00:51	0:06:10
	0:01:09	0:00:11	0:00:30	0:01:50	0:05:51	0:00:14	0:01:18	0:07:23	LoQ	0:04:42	0:00:03	0:00:48	0:05:33
	0:00:57	0:00:10	0:00:27	0:01:34	0:05:28	0:00:14	0:01:18	0:07:00	LoQ	0:04:31	0:00:04	0:00:51	0:05:26
	0:01:08	0:00:12	0:00:27	0:01:47	0:05:29	0:00:14	0:01:18	0:07:01	LoQ	0:04:21	0:00:02	0:00:51	0:05:14
	0:00:56	0:00:11	0:00:26	0:01:33	0:05:33	0:00:14	0:01:17	0:07:04	LoQ	0:04:37	0:00:03	0:00:51	0:05:31
	0:01:04	0:00:35	0:00:26	0:02:05	0:05:30	0:00:15	0:01:22	0:07:07	LoQ	0:04:26	0:00:20	0:00:56	0:05:02
	0:01:06	0:00:11	0:00:27	0:01:44	0:05:22	0:00:14	0:01:16	0:06:52	LoQ	0:04:16	0:00:03	0:00:49	0:05:08
	0:01:02	0:00:11	0:00:28	0:01:41	0:05:31	0:00:16	0:01:25	0:07:12	LoQ	0:04:29	0:00:05	0:00:57	0:05:31
Průměr	0:01:02	0:00:13	0:00:28	0:01:43	0:05:53	0:00:15	0:01:38	0:07:46		0:04:51	0:00:02	0:01:10	0:06:03
Min	0:00:56	0:00:10	0:00:26	0:01:33	0:05:22	0:00:14	0:01:16	0:06:52		0:04:16	0:00:02	0:00:48	0:05:02
Max	0:01:09	0:00:35	0:00:31	0:02:05	0:07:46	0:00:23	0:04:19	0:12:28		0:06:48	0:00:20	0:03:52	0:10:52

Obr. 28 Tabulka výpočtových časů

Testování rychlosti probíhalo na 30 trénovacích souborech, které byly totožné u obou aplikací. Tyto testovací výpočty probíhaly na jednom stroji, které využívalo procesor

AMD Ryzen5 3600. Part 1 a Part 2 jsou pro obě analýzy stejné. Z toho plyne, že Part 3 je jedinou částí, která by měla obsahovat některé změny na základě výběru limitů. Obr. 28 obsahuje červeně zvýrazněná data, která se výrazně liší od zbytku hodnot a o kterých se dozvíme níže. Nyní si probereme jednotlivé výsledky, které lze vyčíst.

- MATLAB
    - Part 1 – Tato část trvala průměrně 65 sekund při výpočtu LoD, avšak jak již bylo řečeno i pro LoQ je Part 1 a Part 2 stejný. Z tohoto důvodu lze říct, že průměrná doba výpočtu této části je 63 sekund. Rozdíl mezi minimální a maximální hodnotou je 22 sekund. Tento rozdíl značí, že výsledek je lehce závislý na výpočtu dat, které počítač mohl počítat na pozadí.
    - Part 2 – Tato část má průměrnou dobu výpočtu 11 sekund, avšak při analýze LoQ je vidět, že jeden z časů byl 35 sekund. Tato hodnota je však nejspíše způsobená některým procesem, který proběhl na pozadí a jelikož se jedná o ojedinělý výsledek, lze ho zanedbat.
    - Part 3 – Při výpočtu limitů LoD byl průměr 19 sekund, který měl velmi konzistentní výsledky. Pro LoQ byl průměr 28 sekund, který trvá déle o 9 sekund z důvodu podmíněného formátování, které se musí aplikovat při analýze s limitem LoQ.
    - Celkem – Průměrný čas analýzy 30 tréninkových dat byl 99 sekund. Rozdíl mezi největším a nejmenším časem byl 38 sekund, avšak tento rozdíl je způsoben ojedinělým výskytem, který přidává 22 sekund k běžnému času.
- Python
    - Part 1 – Při výpočtu této části byl na pozadí prováděn nějaký výpočet, který zanesl do měření chybu, jelikož čas analýzy se prodloužil o 140 sekund. V měřeních kolem této hodnoty jsou také o něco větší hodnoty, které na tuto skutečnost také poukazují. Pokud tyto skutečnosti zanedbáme byla by průměrná doba této části zhruba 5 minut a 30 sekund.
    - Part 2 – Při analýze této části se průměrná doba pohybuje okolo 15 sekund. Jedna hodnota dosahuje výrazně lepších výsledků než MATLAB, avšak u řádku, který je nejspíše chybný některým procesem na pozadí. Jedna hodnota je také výrazně delší než zbytek testů, avšak stále je to u řádku, který byl silně zatížený výpočtem na pozadí.
    - Part 3 – Tato část také obsahuje analýzu zatíženou procesem na pozadí, ale tentokrát je to již výrazné zhoršení výsledku, které analýzu této části prodloužilo až čtyřnásobně. Pokud bychom tuto hodnotu nevzali v potaz, byla by průměrná doba této části 80 sekund.

    - Celkem – Průměrný čas analýzy v Pythonu trvá průměrně 7 minut s největším rozdílem 77 sekund, pokud zanedbáme již zmiňované měření.
- Rozdíl – Při zanedbání hodnot, které pravděpodobně ovlivnil proces na pozadí operačního systému, z tabulky vyplývá, že rozdíl mezi analýzami je cca 5 minut a 15 sekund u LoD a 5 minut a 30 sekund u LoQ.

Mezi další výsledky by se mohl řadit i výsledný vzhled aplikačního rozhraní. Jelikož se jedná o webovou aplikaci, bylo potřeba výsledný produkt upravit do podoby, která by více odpovídala všem možnostem, které se uživateli normálně dostávají při práci s aktuální adresářovou strukturou. Tato funkcionalita ve webovém rozhraní chybí.

9 Diskuse

Hlavním cílem této práce bylo vytvoření webové aplikace, která bude detekovat somatické bodové varianty u pacientů s leukémií. Jednotlivé kapitoly teoretické části předložené diplomové práce byly vybrány tak, aby dokázaly popsat jednotlivé okruhy potřebné k uchopení základní problematiky a pojmů. Probíranou problematiku a základní informace lze získat z webových stránek produktů, elektronických zdrojů, z přednášek probíraných během celého studia na FBMI ČVUT či z tištěných zdrojů. Vypracování některých částí teoretických okruhů této diplomové práce probíhalo až zpětně po vytvoření samotného konečného řešení. Důvodem bylo to, že k samotné práci nebylo třeba plně znát metody sekvenování, typy leukémií apod., ale stačilo krátké vysvětlení, pro uchopení celé myšlenky výsledného produktu praktické části.

Návrh a implementace praktické části projektu probíhala souvisle s psaním teoretické části. Samotný vývoj aplikace by se dal nazvat jako iterativní proces, který se dá rozdělit na designovou část a výpočtovou část. Jednotlivé části návrhu vzhledu stránek byly konzultovány se zadavatelkou práce, která poskytovala zpětnou vazbu, jež byla zaimplementována do výsledného řešení práce. Vývoj výpočetní části probíhal souběžně se spuštěnou funkční verzí v MATLAB, kde vývoj probíhal zároveň na základě hodnot tréninkových dat. Toto sloužilo k lepšímu pochopení napsaného kódu v programovacím jazyce MATLAB a ke kontrole výsledků, které vycházely v obou programech totožně.

Ze zadání práce vyplývalo, že je potřeba upravit již existující řešení a předělat ho do vhodnější formy. Z tohoto důvodu byla současná aplikace použita jako šablona, která byla brána jako výchozí bod. Tohoto výchozího bodu bylo žádoucí dosáhnout i u navrhované aplikace. Pro vytvoření produktu, který bude moci být spuštěn na libovolném prostředí, byla využita kontejnerizace, která je velmi často používána v reálných řešeních.

Velkou část implementace tvoří popis postupu, kterým se aplikace vyvíjela a snažila se dodržet reálný postup, jaký byl použit v již existujícím řešení. V tomto postupu jsou

popsány a ukázány zajímavé části vytvořené aplikace, jakož i vysvětlení funkčnosti jednotlivých prvků.

Tato práce splnila všechny vytyčené cíle, jako je například, virtuální prostředí, logování dat, tvorba dokumentace nebo veřejně dostupný kód. Tento výsledný program se snaží jednoznačně a výstižně pojmenovat jednotlivé proměnné a funkce, aby byl kód co nejlépe čitelný. Některé části kódu obsahují komentáře pro přehlednější a jasnější uchopení dané části. V kapitole 6 Implementace se popisu kódu příliš nevěnujeme, jsou zde zobrazeny jen důležité části, které jsou významné pro pochopení základní struktury aplikace. Výsledné řešení mělo podporovat běh na jakémkoli prostředí. To nebylo vyzkoušeno na všech platformách, avšak na základě použitých technologií a virtualizace a dle dostupné dokumentace, lze předpokládat plnou funkčnost například i v cloudovém prostředí.

Jelikož mezi cíle práce měla být i snaha vylepšit současné řešení, která se z pohledu výpočetní rychlosti neuskutečnila, funkčnost výsledné aplikace má stále prostor ke zlepšení. Pokud shrneme zmíněné poznatky z kapitoly 8 Výsledky je patrné, že původní aplikace analyzuje data cca 3krát rychleji. Hlavní optimalizační nedostatek je v části Part 1, která vytváří základní strukturu souboru výsledku. V Pythonu je vytváření tohoto souboru řešeno iterativním procesem pomocí knihovny openpyxl, která ho při každé iteraci otevře a zapíše. Otevírání a zavírání těchto souborů je velmi zdlouhavý proces, a tudíž optimalizace této části by byla do budoucna vhodná. Jedním z možných řešení by mohlo být zapisování do souboru až úplně na konci analýzy dat, kde by se soubor otevíral pouze jednou. Dalším možným řešením by možná bylo vhodné zvolit jinou knihovnu, která by mohla práci s excelovským souborem urychlit.

Zásadní otázkou vylepšení by mohlo být také větší využití funkcí, které by mohly práci rozdělit na více menších částí. Ty by následně usnadnily úpravy a práci s některými částmi kódu. Dalším vylepšením, by mohlo být přidání rozšířených možností k analýze dat. Uživatel by mohl měnit například soubor genetických mutací, které jsou staticky definovány na základě článku [21], změny hladiny významnosti nebo například ošetření více chybových možností. Dalším změnou, která by mohla vylepšit webové rozhraní, by mohla být úprava některých prvků k zajištění plně responzivního zobrazení. To se sice do jisté míry chová responzivně, avšak některé prvky lehce zasahují do jiných pokud se obrazovka příliš zmenší. V neposlední řadě je v plánu pustit tuto aplikaci do světa. Z tohoto důvodu se plánuje řešení této práce udělat jak v jazyce českém, tak anglickém. Kvůli budoucímu plánu pustit aplikaci na veřejně dostupné uložiště byla zvolena nejméně restriktivní licence, která zajistí autorské právo na předložené řešení a tou se stala licence MIT.

Další částí projektu byla dockerizace, která využívá izolované prostředí k běhu na všech typech systémů. Tyto dockerovské obrazy, lze lokálně sestavit při použití gitového

repositáře, jak je popsáno v kapitole 7 Uživatelská dokumentace či stáhnout již sestavený kontejner.

Ve výsledném systému je tedy stále prostor pro vylepšení, avšak aplikace splňuje vytyčené cíle a požadavky práce zmíněné v kapitole 5 Návrh aplikace a vyplívající ze zadání této práce. Konečné řešení je tedy dobrým základem pro budoucí vylepšování a přidávání funkčností do tohoto řešení.

10 Závěr

V diplomové práci byly úspěšně splněny a dosaženy všechny stanovené cíle, kterými bylo probrání problematiky návrhu, seznámení se se základními teoretickými znalostmi nutnými k úplnému porozumění práce, a k implementaci a testování daného řešení.

Teoretická část práce čerpala převážně z internetových zdrojů organizací zabývajících se probíranou tématikou, tištěnými a elektronickými zdroji, které se skládaly z odborné literatury či zdravotnických článků a publikací, anebo z oficiálních dokumentací vybraných technologií, jež byly použity a zmíněny v této práci.

Praktická část práce zahrnuje návrh aplikace, implementaci a uživatelskou dokumentaci, která odpovídá současnému stavu funkční verze aplikace. Tato aplikace má stále prostor pro zlepšení, avšak její výsledky odpovídají aktuálně používanému řešení, a proto je možné, aby tato aplikace mohla současnému řešení konkurovat, i když její výpočet trvá skoro 4krát déle. Důvodem tohoto tvrzení je zevšeobecnění, dockerizace a izolování prostředí pro aplikaci na jiné operační systémy. Z tohoto důvodu má toto řešení mnohem větší potenciál růstu.

Seznam použité literatury

[1] NHSWEBSITE. Chronic myeloid leukaemia. *Chronic myeloid leukaemia* [online]. 3. duben 2023 [vid. 2023-04-15]. Dostupné z: https://www.nhs.uk/conditions/chronic-myeloid-leukaemia/

[2] PAVLA SUCHÁNKOVÁ A COL. *NextDOM 2.0: Detector of Somatic Point Mutations in Leukemias Resistant to Therapy* [online]. 2019. ISSN 17568722. Dostupné z: doi:10.1186/s13045-019-0815-5

[3] MUDR. ANTONÍN ŠÍPEK JR. *Genetika - Biologie: Váš zdroj informací o genetice a biologii* [online]. 2010 [vid. 2023-05-14]. Dostupné z: http://www.genetika-biologie.cz/geny-znaky

[4] VERONIKA VYMĚTALOVÁ. *Biologie pro biomedicínské inženýrství*. 1. Praha: ČVUT, 2008. ISBN 978-80-01-04013-3.

[5] PŘISPĚVATELÉ WIKISKRIPT. *Bioinformatika* [online]. [vid. 2023-05-01]. Dostupné z: https://www.wikiskripta.eu/index.php?title=Bioinformatika&oldid=462500

[6] POSPÍŠILOVÁ Š., B. TICHÝ a J. MAYER. ČASOPIS LÉKAŘŮ ČESKÝCH Č. *ČASOPIS LÉKAŘŮ ČESKÝCH*. 2009, **148**(7), 296–302.

[7] BRAZDILOVA, Kveta, David PRIHODA, Quynh TON, Heath KLOCK a Danny A. BITTON. TraceTrack, an Open-Source Software for Batch Processing, Alignment and Visualization of Sanger Sequencing Chromatograms. *bioRxiv* [online]. 2022, 2022.07.28.501824 [vid. 2023-05-02]. Dostupné z: doi:10.1101/2022.07.28.501824

[8] EXPERIMENTÁLNÍ BIOLOGIE ODDĚLENÍ GENETIKY MOLEKULÁRNÍ BIOLOGIE, Ústav A a Marie KÝVALOVÁ BAKALÁŘSKÁ PRÁCE. *MASARYKOVA UNIVERZITA PŘÍRODOVĚDECKÁ FAKULTA SEKVENAČNÍ METODY NOVÉ GENERACE: JEJICH PRINCIPY A POTENCIÁLNÍ VYUŢITÍ V GENETICE ČLOVĚKA, ETICKÉ ASPEKTY*. 2011.

[9] MAXAM, Allan M a Walter GILBERT. *A new method for sequencing DNA (DNA chenistry/dimethyl sulfate cleavage/hydrazine/piperidine)* [online]. 1977. Dostupné z: https://www.pnas.org

[10] SANDER, F a A R GOULSON. *A Rapid Method for Determining Sequences in DNA by Primed Synthesis with DNA Polymerase*. 1976.

[11] PROF. RNDR. ROMAN PANTŮČEK, Ph.D. Prezentace k sedmé lekci sekvenovani. In: . B.m. 26. duben 2018.

[12] BAYAT, Ardeshir. *Clinical review Science, medicine, and the future Bioinformatics* [online]. nedatováno. Dostupné z: www.ebi.ac.uk

[13] MASARYKOVA UNIVERZITA V BRNĚ. *Sekvenování DNA • stanovení pořadí nukleotidů v molekule DNA (primární struktury)*. nedatováno.

[14] MARTIN KOLÍSKO. Moderní metody sekvenování DNA. *Živa* [online]. 2017, **3**. Dostupné z: https://flxlexblog.wordpress.com/2014/

[15] ANSORGE, Wilhelm J. *Next-generation DNA sequencing techniques* [online]. duben 2009. ISSN 18716784. Dostupné z: doi:10.1016/j.nbt.2008.12.009

[16] SOFTGENETICS. *NextGENe, Next Generation Sequencing Software for Biologists*. 2011.

[17] THE 1000 GENOMES PROJECT CONSORTIUM. An integrated map of genetic variation from 1,092 human genomes. *Nature* [online]. 2012, **491**(7422), 56–65 [vid. 2023-05-08]. ISSN 0028-0836. Dostupné z: doi:10.1038/nature11632

[18] ZVÁROVÁ, Jana. *Biomedicínská statistika I.* [online]. 3. B.m.: Karolinum, 2016. ISBN 978-80-246-1931-6. Dostupné z: http://new.euromise.org/czech/tajne/ucebnice/html/html/node1.html

[19] SARKAR, Indra Neil. *Methods in Biomedical Informatics : a Pragmatic Approach*. 2013. ISBN 9780124016842.

[20] GORDON COHEN M.D., M.P.H. *What Is Philadelphia Chromosome Positive ALL?* [online]. 2017 [vid. 2023-04-30]. Dostupné z: https://www.stbaldricks.org/blog/post/what-is-philadelphia-chromosome-positive-all/

[21] SOVERINI, Simona, Elisabetta ABRUZZESE, Monica BOCCHIA, Massimiliano BONIFACIO, Sara GALIMBERTI, Antonella GOZZINI, Alessandra IURLO, Luigiana LUCIANO, Patrizia PREGNO, Gianantonio ROSTI, Giuseppe SAGLIO, Fabio STAGNO, Mario TIRIBELLI, Paolo VIGNERI, Giovanni BAROSI a Massimo BRECCIA. Next-generation sequencing for BCR-ABL1 kinase domain mutation testing in patients with chronic myeloid leukemia: A position paper. *Journal of Hematology and Oncology* [online]. 2019, **12**(1), 1–11 [vid. 2023-04-30]. ISSN 17568722. Dostupné z: doi:10.1186/S13045-019-0815-5/TABLES/2

[22] MACHOVA POLAKOVA, Katerina, Vojtech KULVAIT, Adela BENESOVA, Jana LINHARTOVA, Hana KLAMOVA, Monika JARUSKOVA, Caterina DE BENEDITTIS, Torsten HAFERLACH, Michele BACCARANI, Giovanni MARTINELLI, Tomas STOPKA, Thomas ERNST, Andreas HOCHHAUS, Alexander KOHLMANN a Simona SOVERINI. Next-generation deep sequencing improves detection of BCR-ABL1 kinase domain mutations emerging under tyrosine kinase inhibitor treatment of chronic myeloid leukemia patients in chronic

phase. *Journal of Cancer Research and Clinical Oncology* [online]. 2015, **141**(5), 887–899 [vid. 2023-04-30]. ISSN 14321335. Dostupné z: doi:10.1007/S00432-014-1845-6/METRICS

[23] AMERICAN CANCER SOCIETY, Inc. *American Cancer Society* [online]. [vid. 2023-04-30]. Dostupné z: https://www.cancer.org

[24] ADÉLA BENEŠOVÁ A COL. *DO NOT WASTE THE DATA: A PIPELINE FOR EVALUATION OF BCR-ABL1 KINASE DOMAIN MUTATION USING INDIVIDUAL THRESHOLDS*. 2020.

[25] NATIONAL LIBRARY OF MEDICINE. *Basic Local Alignment Search Tool* [online]. [vid. 2023-05-02]. Dostupné z: https://blast.ncbi.nlm.nih.gov/blast/Blast.cgi

[26] CAMACHO, Christiam, George COULOURIS, Vahram AVAGYAN, Ning MA, Jason PAPADOPOULOS, Kevin BEALER a Thomas L MADDEN. BLAST+: architecture and applications. *BMC bioinformatics* [online]. 2009, **10**, 421 [vid. 2023-05-02]. ISSN 1471-2105. Dostupné z: doi:10.1186/1471-2105-10-421

[27] PACBIO. *Sequencing Methods For Solutions That Matter* [online]. [vid. 2023-05-02]. Dostupné z: https://www.pacb.com

[28] NW.JS COMMUNITY. *NW.js (previously known as node-webkit)* [online]. [vid. 2023-04-17]. Dostupné z: https://nwjs.io

[29] ANANDMEG A COL. *Tutorial: Get started with C# and ASP.NET Core in Visual Studio* [online]. 24. únor 23n. l. [vid. 2023-04-17]. Dostupné z: https://learn.microsoft.com/en-us/visualstudio/get-started/csharp/tutorial-aspnet-core?view=vs-2022

[30] ASTARI, S. *What is html* [online]. 10. duben 2023 [vid. 2023-04-17]. Dostupné z: https://www.hostinger.com/tutorials/what-is-html

[31] ARTŪRAS, B. *What is CSS and How Does It Work?* [online]. 4. leden 2023 [vid. 2023-04-17]. Dostupné z: https://www.hostinger.com/tutorials/what-is-css

[32] MOZILA DEVELOPERS A WIKI CONTRIBUTORS. *What is JavaScript* [online]. [vid. 2023-04-17]. Dostupné z: https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript

[33] HAYASHI, Chikio. What is Data Science ? Fundamental Concepts and a Heuristic Example. In: [online]. 1998 [vid. 2023-04-16], s. 40–51. ISBN 9784431702085. Dostupné z: doi:10.1007/978-4-431-65950-1_3

[34] REVOLUTION SYSTEMS A COL. *What is Python? Executive Summary* [online]. [vid. 2023-04-16]. Dostupné z: https://www.python.org/doc/essays/blurb/

[35] GCX11. *Lekce 1 - Úvod do Pythonu* [online]. [vid. 2023-04-16]. Dostupné z: https://www.itnetwork.cz/python/zaklady/python-tutorial-uvod-do-pythonu-a-zakladni-matematicke-operace/?all-comments

[36] CHACON, Scott a Ben STRAUB. *Pro Git* [online]. 2014 [vid. 2023-04-17]. Dostupné z: https://github.com/progit/progit2/releases/download/2.1.386/progit.pdf

[37] DOCKER INC. *Docker* [online]. [vid. 2023-04-17]. Dostupné z: https://www.docker.com

[38] AMAZON WEB SERVICES, Inc. *What is Docker?* [online]. [vid. 2023-04-17]. Dostupné z: https://aws.amazon.com/docker/

[39] FREE SOFTWARE FOUNDATION, Inc. *Free Software Foundation* [online]. 2007 [vid. 2023-05-14]. Dostupné z: https://www.fsf.org

[40] FREE SOFTWARE FOUNDATION, Inc. *GNU General Public Licence* [online]. 2013 [vid. 2023-05-14]. Dostupné z: https://www.gnu.org/licenses/gpl-3.0.html

[41] MYŠKA, Matěj, Radim POLČÁK, Jaromír ŠAVELKA a Libor KYNCL. *Veřejné licence v České republice* [online]. 2014 [vid. 2023-05-14]. Dostupné z: https://is.muni.cz/repo/1203341/Myska_et_al._-_Verejne_licence_2.0_-_online.pdf