ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

FAKULTA BIOMEDICÍNSKÉHO INŽENÝRSTVÍ

Katedra informačních a komunikačních technologií v lékařství

Zpracování, uložení a agregace genetických variant

Processing, storage and aggregation of genetic variants

Bakalářská práce

Studijní program:	Informatika a kybernetika ve zdravotnictví
Studijní obor:	Biomedicínská informatika

Autor bakalářské práce:	Veronika Bůžková
Vedoucí bakalářské práce:	doc. Mgr. Radim Krupička, Ph.D

Kladno 2023

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení:	**Bůžková**	Jméno:	**Veronika**	Osobní číslo:	**500001**
Fakulta:	**Fakulta biomedicínského inženýrství**				
Garantující katedra:	**Katedra informačních a komunikačních technologií v lékařství**				
Studijní program:	**Informatika a kybernetika ve zdravotnictví**				

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Zpracování, uložení a agregace genetických variant

Název bakalářské práce anglicky:

Processing, storage and aggregation of genetic variants

Pokyny pro vypracování:

Cílem bakalářské práce je navrhnout metody, vybrat technologie a vytvořit skripty pro vytvoření databáze agregovaných anotovaných genetických variant pro daný panel vzorků. V práci bakalářské práce se seznamte s nástroji pro práci s genetickými daty. Na základě rešerše vyberte technologie pro zpracování, anotaci a databázi pro ukládání velkých genetických dat. Navrhněte a popište řetězec zpracování souborů gvcf. Implementujte předzpracování gvcf, uložení všech vzorků do databáze, nasazení anotačního nástroje, a výpočet agregovaných parametrů, jako je frekvence jednotlivých variant ve výzkumném panelu a pokrytí jednotlivých bází.

Seznam doporučené literatury:

[1] EWENS, W. J. a Gregory R. GRANT, Statisical [sic] methods in bioinformatics: an introduction, ed. 2., New York, N.Y.: Springer, 2005, ISBN 978-0-387-40082-2
[2] Hail Team, Hail documentation , 2015, https://hail.is/docs/0.2/index.html
[3] Karczewski, K.J., Francioli, L.C., Tiao, G. et al., The mutational constraint spectrum quantified from variation in 141,456 humans., Nature 581, Přístupné z: https://doi.org/10.1038/s41586-020-2308-7

Jméno a příjmení vedoucí(ho) bakalářské práce:

doc. Mgr. Radim Krupička, Ph.D.

Jméno a příjmení konzultanta(ky) bakalářské práce:

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

doc. Ing. Karel Hána Ph.D.
vedoucí katedry

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

PROHLÁŠENÍ

Prohlašuji, že jsem bakalářskou práci s názvem „Zpracování, uložení a agregace genetických variant“ vypracovala samostatně a použila 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 18.5.2023 ……..……...

Veronika Bůžková

PODĚKOVÁNÍ

Ráda bych poděkovala vedoucímu práce doc. Mgr. Radimu Krupičkovi, Ph.D. Děkuji především za možnost pravidelných konzultací a za jeho odborný dohled při tvorbě práce.

ABSTRAKT

Zpracování, uložení a agregace genetických variant

Populační databáze pomáhají při interpretaci genetických variant. V těchto databázích jsou informace o frekvenci variant v různých populacích, které pomáhají určit, zda je varianta nalezená u pacienta běžná pro jeho populaci.

Cílem práce bylo vytvořit řešení umožňující v budoucnu tvorbu české populační databáze. Prvním úkolem bylo navrhnout řetězec zpracování a pro něj vybrat metody a nástroje pro následnou implementaci. Navržený řetězec zpracování implementovat a spustit na vzorku dat.

Pro implementaci bylo využito krom jiných nástrojů Docker, Hail a VEP (Variant Effect Predictor).

Výsledkem práce jsou navržené skripty a vytvořené prostředí v Dockeru pro jejich správné spuštění. Pomocí těchto skriptů byla úspěšně vytvořená ze vzorku dat populační databáze.

Klíčová slova

Populační databáze, analýza genetických variant, variabilita lidského genomu, Hail

ABSTRACT

Processing, storage and aggregation of genetic variants

Population databases help in the interpretation of genetic variants. These databases contain information about the frequency of variants in different populations to help determine whether a variant found in a patient is common in their population.

The goal of the work was to create a solution enabling the creation of a Czech population database in the future. The first task was to design the workflow and to select methods and tools for subsequent implementation. Next use the designed workflow and implement it and run it on the sample data.

Among other tools, Docker, Hail and VEP (Variant Effect Predictor) were used for the implementation.

The result of the work is the created scripts and the created environment in Docker for their proper execution. From using these scripts on the sample data was successfully created a population database.

Keywords

Population database, genetic variants analysis, human genome variability, Hail

Obsah

Seznam symbolů a zkratek

Seznam zkratek

Zkratka	Význam
DNA	Deoxyribonukleová kyselina
RNA	Ribonukleová kyselina
A	Adenin
T	Thymin
C	Cytosin
G	Guanin
mDNA/mtDNA	Mitochondriální DNA
tRNA	Transferová RNA
rRNA	Ribosomální RNA
mRNA	Messenger RNA
ncRNA	Nekódující RNA
dATP	Deoxyadenosintrifosfát
dTTp	Deoxythymidin trifosfát
dCTP	Deoxycytidin trifosfát
dGTP	Deoxyguanosin trifosfát
ATP	Adenosintrifosfát
UTP	Uridintrifosfát
CTP	Cytidintrifosfát
GTP	Guanosintrifosfát
NGS	Nové generace sekvenování
DP	počet filtrovaných čtení (readů) pro daný lokus (bázi)
AN	Celkový počet alel ve volaném genotypu
AC	Počet alel pro každou alternativní alelu ve volaném genotypu
AF	Alelická frakce zkratka AF ale také popisuje hodnotu alelické frekvence počítané z AC a AN
GT	Genotyp
PL	Pravděpodobnost možných genotypů (diploidní organismy mají 3 možné genotypy.
GQ	Kvalita genotypu
AD	Počet nefiltrovaných readů podporující nalezené alely

1 Úvod

DNA (deoxyribonukleová kyselina) je molekula složená z nukleotidů. Jejich sekvence kóduje genetické informace jedince. Kompletní soubor genetické informace uložené v DNA se nazývá genom. Celý genom o určuje znaky a vlastnosti jedince.

Změna v sekvenci nukleotidů může zapříčiňovat vážné nemoci, ale nemusí mít žádný viditelný efekt. Každý člověk má genom o velikosti 3,2 miliardy bází v nichž má přibližně pět miliónů variant (změn v sekvenci nukleotidů), kde většina z nich je běžná pro jeho populaci a okolí ve kterém žije.

Díky tomu, že je variabilita v lidském genomu obvyklá, tak je identifikace odhalení medicínsky významných variant náročné. V tom ale mohou pomoci populační databáze variant. Hlavním přínosem populačních databází je to, že obsahují informace o tom, jak často se varianty objevují v rozdílných populacích. To pomáhá rozlišit, zda varianta nalezená u pacienta je běžná pro lidi ze stejné populace či nikoliv.

Cílem této práce je navrhnout řetězec zpracování, vybrat metody a nástroje pro tvorbu populační databáze. Následně provést implementaci navrženého řetězce zpracování a na vzorku dat vytvořit populační databázi. Výsledkem práce je připravené řešení pro spuštění na větším počtu dat.

Motivací této práce je v budoucnu vytvořit populační databázi obsahující varianty české populace.

V následující kapitole Přehled současného stavu je rozepsaný teoretický základ této práce. Obsahuje úvod do molekulární biologie, specificky molekul DNA, RNA a proteinů a genetiky. Obsahuje také přehled sekvenačních nástrojů, kroky bioinformatické analýzy genetických informací a přehled nástrojů souvisejících s tvorbou populační databáze.

Kapitola Návrh zpracování a metody popisuje navrhnuté řešení, výběr metod a nástrojů spolu se softwarovými a hardwarovými požadavky pro následnou implementaci.

V kapitole Implementace je vysvětlen způsob nasazení do Docker prostředí a použití vytvořených skriptů spolu s jejich popisem.

Kapitoly Výsledky, Diskuse a Závěr slouží k popsání výsledků, jejich interpretaci a zhodnocení celé práce.

2 Přehled současného stavu

Každý živý organismus se skládá z buněk. Buňka je základní jednotkou života, rostlin a živočichů. [1]

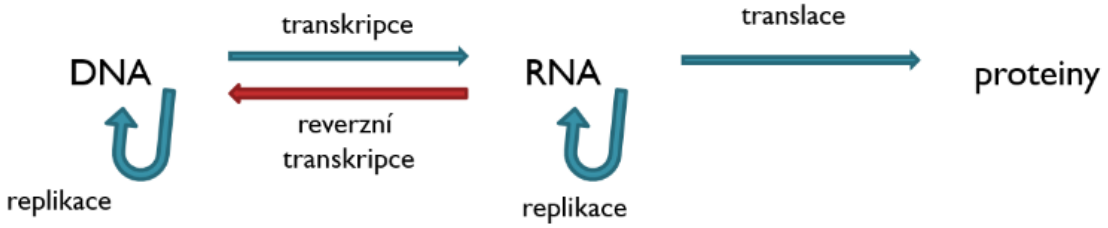
Buňky rozdělujeme na prokaryotické a eukaryotické buňky. Eukaryotické buňky se liší od prokaryotických tím, že mají organely a samostatný nukleus (jádro) oddělující genetický materiál (DNA – deoxyribonukleová kyselina organizovaná v chromozomech) od cytoplazmy a zbytku buňky. Nukleus je uzavřená organela ohraničená dvěma membránami. Obsahuje většinu DNA eukaryotické buňky, zbytek je uložen v semiautonomních organelách (v mitochondriích a v chloroplastech u rostlin). [1]

Eukaryotické buňky můžou existovat ve formě jednobuněčných organismů (měňavka a kvasinky) nebo tvořit mnohobuněčné organismy – rostliny, houby a lidi. [1]

2.1 Centrální dogma molekulární biologie

Centrální dogma molekulární biologie bylo formulované roku 1958 Francisem Crickem. Popisuje procesy tří molekul (DNA, RNA – ribonukleová kyselina a proteinů). Jedná se o proces replikace DNA, transkripce (přepis) DNA na RNA a translace (překlad) RNA na protein. [2]

Tyto procesy jsou odpovědné za expresi genetické informace a jsou zobrazeny na Obrázek 2.1Obrázek 2.1.



Obrázek 2.1 Procesy centrálního dogmatu molekulární biologie mezi molekulami DNA, RNA a proteinů. DNA a RNA jsou schopny replikace, DNA transkripcí vytvoří RNA a ta pomocí translace vytvoří proteiny. Ojediněle může dojít k reverzní transkripci RNA do DNA (využívají toho především retroviry jako HIV). [3]

2.1.1 DNA

Lidský genom se skládá z makromolekul DNA. Struktura a složení DNA byla poprvé správně popsána v roce 1953, kdy byl vytvořen model dvoušroubovice DNA vědci James Watson a Francis Crick. Za tento objev získali Nobelovu cenu.

Pro vytvoření modelu byla důležitou podmínkou rentgenová difrakční fotografie DNA zhotovená Rosalind M. Franklinovou. [2]

DNA je složena z cukru (deoxyribózy), fosfátu (fosforečnan) a na každý cukr se váže dusíkatá báze. Báze v DNA se dělí na purinové báze (adenin – A, guanin – G) a pyrimidinové báze (cytosin – C, thymin – T). [4]

Jeden fosfát, cukr a navázaná dusíkatá báze tvoří nukleotid, základní stavební jednotku nukleových kyselin. Nukleotidy jsou spojeny pomocí fosfodiesterových vazeb do polynukleotidových řetězců a tvoří primární strukturu DNA. [4]

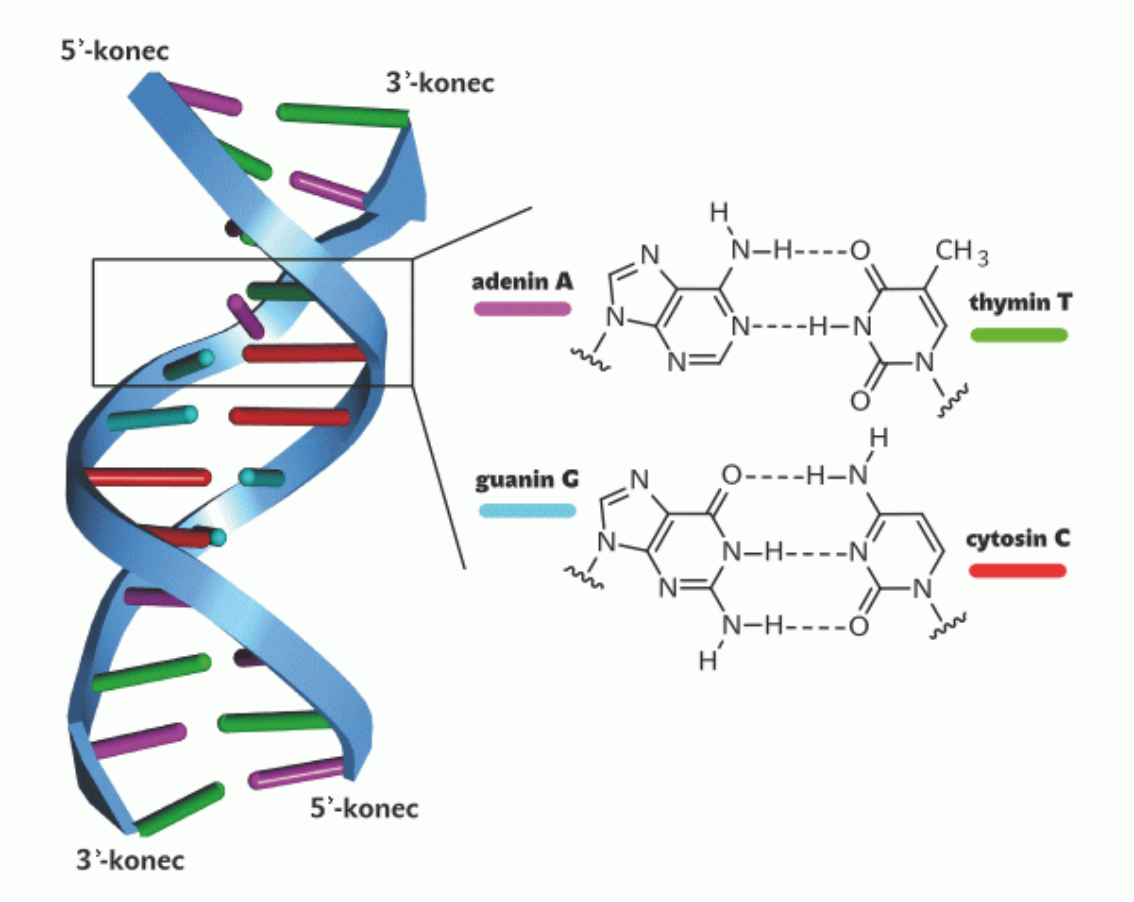
Jedna makromolekula DNA má sekundární strukturu tvořenou dvěma polynukleotidovými řetězci stočenými do pravotočivé dvoušroubovice. Dva polynukleotidové řetězce jsou spojeny na úrovní bází (vodíkovými můstky) a platí zákon komplementarity. [4]

Zákon komplementarity zajišťuje, že se na sebe váží vždy dvě specifické báze (1 pyrimidinová a 1 purinová):

- A–T (spojeny 2 vodíkovými můstky)
- C–G (spojeny 3 vodíkovými můstky)

Řetězce v DNA jsou antiparalelní – jeden řetězec má směr fosfodiesterových vazeb (5' » 3') a druhý má směr opačný k polaritě prvního řetězce (3' » 5'). Podle toho hovoříme o 3' nebo 5' konci. [4]

Na následnicím Obrázek 2.2 je zobrazena struktura a složení DNA. Je na něm také pozorovatelný antiparalelní směr dvou vůči sobě komplementárních řetězců.



Obrázek 2.2 Vlevo na obrázku je vidět dvoušroubovice DNA s viditelnými bázemi. Na dvoušroubovici je také znázorněno označení 5'konec a 3'konec. Vpravo je ukázána chemická struktura bází a jak se na sebe komplementárně váží. [5]

2.1.2 DNA replikace

Komplementarita bází je základní vlastností DNA umožňující její replikaci. Díky komplementaritě DNA mohou obě vlákna dvoušroubovice sloužit jako předloha pro syntetizování komplementárně identických protějšků.

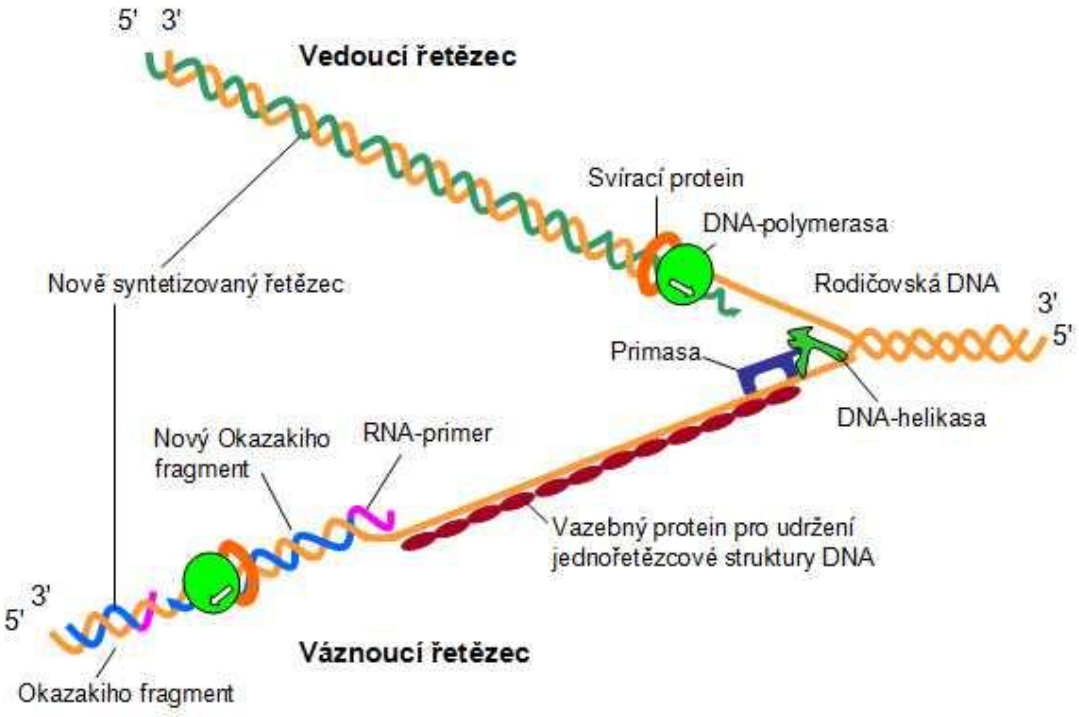
Proces syntézy nové DNA zobrazený na **Chyba! Nenalezen zdroj odkazů.** začíná na místech zvané replikační počátky. Na tato místa se navážou iniciační proteiny a rozbijí vodíkové můstky spojující oba řetězce DNA. Člověk má replikačních počátků v celém genomu asi 10 000 (přibližně 220 na jeden chromozom). Tím, že člověk má tolik replikačních počátků, se značně urychluje čas potřebný pro zkopírovaní celého genomu, jelikož replikace může probíhat na více místech současně. [1] [2]

Na začátcích replikace se nacházejí útvary ve tvaru písmena Y nazývající se replikační vidličky. Replikace je oboustranná, jelikož v jednom replikačním počátku se vytvoří dvě replikační vidličky pohybující směrem od sebe. V replikačních vidličkách jsou navázány proteiny (DNA polymeráza, enzym primasa, DNA-helikasa, SSB-proteiny, svírací protein) replikačního aparátu, které se pohybují ve směru replikace a rozvíjejí dvoušroubovicovou strukturu za současné syntézy nového řetězce.

Replikační vidličky jsou nesymetrické. Důvodem je, že řetězec syntetizující se na templátu 3' → 5' se syntetizuje kontinuálně (vedoucí řetězec), ale syntéza probíhající na templátu 5' → 3' probíhá diskontinuálně (váznoucí řetězec). Váznoucí řetězec vytváří krátké úseky DNA (tzv. Okazakiho fragmenty), které se pomocí enzymů DNA-polymerasy I, DNA-ligasy přeměňují na souvislé vlákno. [6]

DNA se syntetizuje pomocí enzymu DNA polymerázy III, která doplňuje vhodnou bázi (A, T, C, G) komplementárně k původnímu řetězci. Báze vstupující do reakce jsou energeticky bohaté deoxynukleosidtrifosfáty (dATP, dTTP, dGTP a dCTP) a dodávají energii polymerizační reakci. Vznikají dvě nové dvoušroubovice. Každá má jeden originální řetězec a druhý identický nasyntetizovaný řetězec (pokud nepočítáme možné chyby vzniklé během replikace), proto se replikace DNA nazývá jako semikonzervativní. [1] [2]

DNA polymeráza neumí sama začít syntézu nového vlákna. Proto enzym primasa první syntetizuje úseky RNA o velikosti kolem 10 nukleotidů a ty poskytují 3' konec pro DNA-polymerázu a tím slouží jako primery pro syntézu DNA. Je potřeba pouze jeden RNA-primer při syntéze vedoucího řetězce. Naopak při syntéze opožďujícího se řetězce je potřeba RNA-primer pro každý syntetizující se fragment. [6]



Obrázek 2.3 Na obrázku je rodičovská (původní) DNA, která se ve své části helikasou rozdělí na vedoucí a váznoucí řetězec. Na nich RNA-primer inicializuje syntézu pomocí DNA polymerázy. Na obrázků jsou také vidět Okazakiho fragmenty na váznoucím řetězci. [6]

Miliony párů bází se kopírují během každého dělení buňky. Přesto DNA polymeráza zapříčiní pouze jednu chybu na každých 10^7 bází. K většině změnám

v DNA nedochází během replikace DNA, ale účinky chemických reakcí uvnitř buňky. Buňky mají několik mechanismů pro opravu DNA. Většina opravných mechanismů využívá, že DNA je tvořená dvoušroubovicí a tím v sobě ukládá dvě kopie genetické informace. [1]

2.1.3 Chromozomy a uspořádání genomu

Většina lidského genomu (genetické výbavy jedince) je uložena v jádře každé buňky ve formě 46 chromozomů. Počet chromozomů byl určen v roce 1955 Joe Hin Tjio. Chromozomy jsou uspořádány do 23 párů, jeden chromozom z páru se dědí od matky a druhý od otce. Dvacetdva párů se nazývá autozomy – nepohlavní chromozomy a jeden pár chromozomů se nazývá gonozom - tzv. pohlavní chromozom, který určuje pohlaví a jeho pár je tvořený z chromozomů XX (u žen) a XY (u můžu). [2] [4]

Každý lidský chromozom se skládá z jedné spojité dvoušroubovice DNA zabalené do chromatinu, ve kterém je DNA zakomplexované spolu s několika třidami proteinů (histony). Celkem je v jádře každé buňky 46 lineárních DNA molekul obsahujících celkem 3,2 miliard nukleotidových párů. [4]

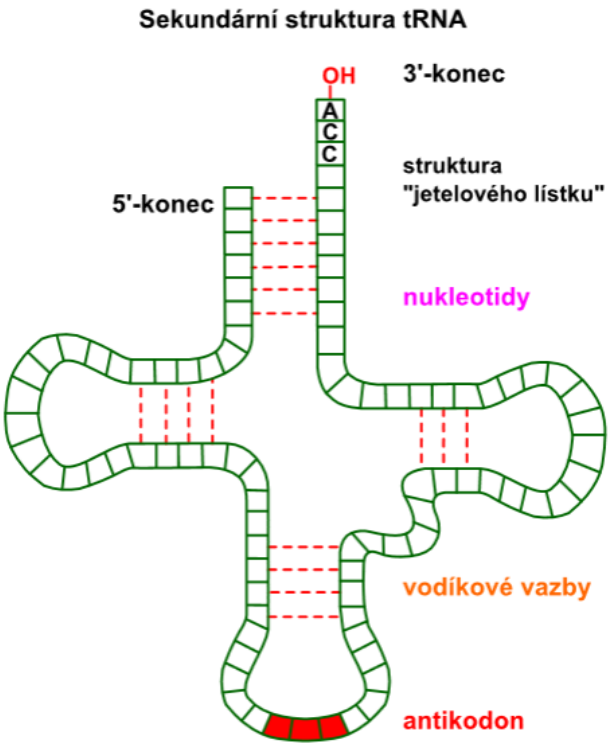
V lidských buňkách je DNA ukryto i v mitochondriích. Mitochondrie jsou semiautonomní organely produkující adenosintrifosfát (ATP) molekulu. ATP je molekula dodávající energii buňce pro většinu buněčných činností. [1]

Mitochondriální DNA (mtDNA nebo mDNA) člověka a ostatních savců je předávána z matčiny mitochondriální DNA (mtDNA) obsažené ve vajíčku. Spermie sice také obsahují mtDNA, ale podle podrobných morfologické studií hlodavců a krav se mtDNA spermie ztrácí v počátcích embryogeneze. Mitochondriální DNA u člověka je dvouvláknová, kruhová molekula o velikosti 16 569 bp. Obsahuje 37 genů kódujících dvě rRNA, 22 tRNA a 13 polypeptidů. Až na jednu regulatorní část mtDNA neobsahuje introny (část která se při transkripci DNA vystříhává viz kapitola 2.1.5). [7]

2.1.4 RNA

DNA je kyselina deoxyribonukleová, která slouží jako úložiště genetické informace člověka. Stejně ne-li více je ale důležitá ribonukleová kyselina (RNA). Její primární struktura je jako u DNA tvořena z nukleotidů (dusíkaté báze, cukru pentózy a ze zbytku kyseliny fosforečné). Rozdíl je však v cukru tvořící nukleotid. V RNA je ribóza místo deoxyribózy. Také u dusíkatých bází najdeme v RNA změnu oproti DNA. V RNA se sice vyskytují dusíkaté báze – A, C, G, ale místo tyminu je vázán uracil (U). [8]

Sekundární struktura viz Obrázek 2.4 je jedno stočené vlákno. Vzniká díky vodíkatým můstkům, které vznikají v místech, kde se ocitnou v blízkosti dva vůči sobě komplementární úseky (A-U; C-G). [8]



Obrázek 2.4 Sekundární struktura RNA specificky transferová RNA (tRNA). Jedná o jedno stočené vlákno nukleotidů s 3‘ a 5‘ koncem. Na několika místech vznikají vodíkové vazby a tím vzniká struktura „jetelového lístku“. Vyobrazený antikodon je trojice nukleotidů objevující se v tRNA a která sloužící pro přiřazení správné aminokyseliny při proteosyntéze podle kodonu na rRNA. [9]

2.1.5 Transkripce DNA do RNA

Všechna RNA buňky vzniká transkripcí DNA. Transkripce začíná rozvolněním DNA a zpřístupněním bází DNA tak, aby jedno vlákno DNA molekuly mohlo sloužit jako templát pro syntézu RNA molekuly. Transkripce využívá komplementarity bází a je založená stejném principu jako replikace DNA s několika rozdíly. [8]

1. Na místo, kde se v templátu nachází A (adenin), se naváže ribonukleotid U (uracil).
2. Nově syntetizované RNA si nezachovává vodíkovou vazbu mezi templátovou DNA a v místě, kde se naváže nový ribonukleotid se odpojí od templátu a DNA se znovu uzavírá.
3. Syntéza využívá RNA polymerázu namísto DNA polymerázy a ribonukleosidtrifosfáty (ATP, UTP, GTP a CTP) namísto deoxynukleosidtrifosfátů.
4. Vznikají malé molekuly RNA, které nemají více než několik tisíc bází. Oproti tomu při syntéze DNA vznikají velké molekuly dlouhé až 250 milionů nukleotidových párů.
5. Syntéza může začít bez využití primeru.

6. Chybovost RNA polymerázy je 1 chyba na 10^4 kopií RNA nukleotidů a je vetší než u DNA polymerázy. [8]

RNA rozdělujeme na dva základní typy:

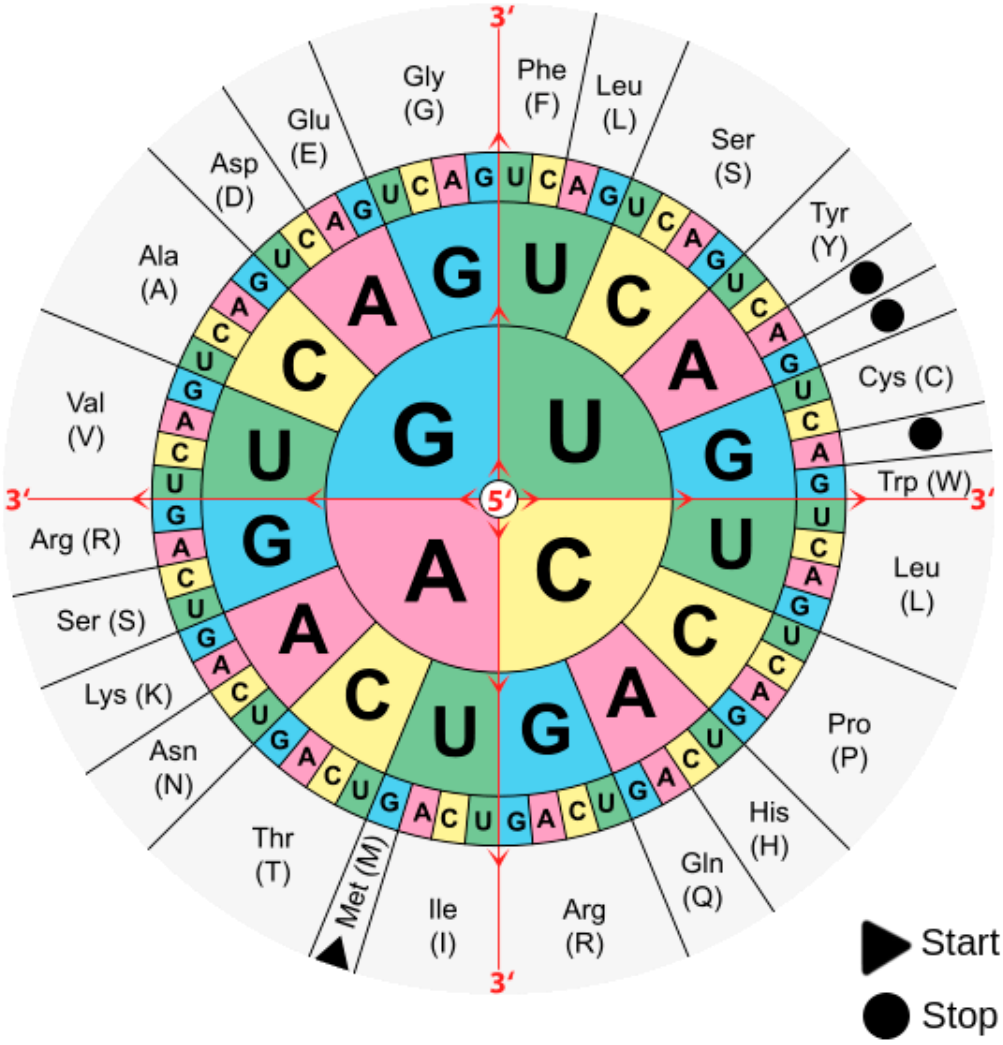
1. Kódující RNA – sloužící jako přímá reference pro translaci proteinu. Takovou RNA označujeme jako kódující RNA a jedná se o tzv. messenger RNA (mRNA). mRNA přenáší informaci z jaderné DNA do cytoplazmy a určuje pořadí aminokyselin které syntetizují protein.
2. Nekódující RNA (ncRNA). Existuje mnoho druhů ncRNA, ale obecně se jedná o RNA regulující genovou expresy (jak moc se konkrétní gen projeví) a jako pomocná jednotka při proteosyntéze (tRNA, rRNA). ncRNA může však fungovat i jako enzym, například se může vázat a štěpit konkrétní úseky DNA. Stále je však hodně druhů ncRNA u kterých ještě neznáme jejich přesnou funkci. [4]

Celý lidský genom obsahuje přibližně 20 000 – 50 000 genů. Přitom ne všechny jsou protein kódující geny. Geny kódující proteiny transkripcí generují pre-mRNA. Z té se dále odebírají regiony nazývané introny a tím vzniká finální mRNA. Tento proces se nazývá „splicing“ a zajišťuje, že finální mRNA je složena pouze z exonů genu (části genu kódující proteiny). Introny ve finální mRNA obsaženy nejsou. [10]

2.1.6 Translace do proteinu

Proteiny vznikají na ribozomech, kde dochází k překladu mRNA molekuly. mRNA molekula se čte z 5‘ do 3‘ konce a každé 3 nukleotidy reprezentují jeden kodon kódující jednu aminokyselinu, která se má nasyntetizovat na ribozomu. Při syntéze proteinu hrají velkou roli i další typy RNA (rRNA a tRNA) a několik pomocných proteinů.

tRNA je transferová RNA sloužící pro přenos správné aminokyseliny na vhodný kodon na mRNA. Struktura jetelového lístku tRNA je vidět na Obrázek 2.4. tRNA má na 3‘ konci sekvenci nukleotidů ACC a obsahuje část nazvanou antikodon, která se váže na příslušný komplementární kodon na mRNA. Kódování aminokyselin pomocí kodonů je vidět na Obrázek 2.5.



Obrázek 2.5 Kruh zobrazující překlad RNA kodonů do proteinů. Střed reprezentuje první nukleotid v kodonu čtený ve směru 5‘ – 3‘. Na kraji kruhu jsou vypsány aminokyseliny, které se kódují kombinací tří nukleotidů. Značky start a stop značí start a stop kodon. [11]

Dalším typem v procesu translace je rRNA obsažená v Ribozomech. Ribozomy jsou organely sloužící pro syntézu proteinů. Ribozomy jsou složeny právě z rRNA a proteinů. Ribozomy dělíme na dvě podjednotky (malá a velká ribosomální podjednotka).

Translaci rozdělujeme na tři fáze iniciace, elongaci a terminaci. mRNA obsahuje kodony začínající (start kodon) a ukončující syntézu (stop kodon). Fáze iniciace využívá start kodonu (u eukaryot se jedna o kodon AUG kódující methionin). Na tRNA se naváže methionin a připojí se k malé ribosomální podjednotce. Malá ribosomální podjednotka se dále váže na mRNA a hledá kodon AUG. Když se nalezne start kodon připojí se velká ribosomální podjednotka.

Elongace je fáze kdy se syntetizuje zbytek proteinu tím, že se ribozom pohybuje po mRNA a podle předlohy tRNA doplňuje vhodnou aminokyselinu podle kodonu na mRNA. Mezi aminokyselina v rámci fáze elongace vznikají peptidové vazby.

Terminace syntézy se signalizuje pomocí třech stop kodonů (UAA, UAG, UGA). V místě těchto kodonů tRNA se váže místo aminokyseliny molekula vody, která ukončuje syntézu, protein se uvolňuje do cytoplazmy. Po skončení proteosyntézy se podjednotky ribozomu od sebe opět uvolňují. [12]

2.2 Variabilita lidského genomu a základní pojmy genetiky

Genomika je studium zabývající se strukturou, funkcí genomu a pochopení genomu jako celku. Genetika je obor zabývající se jednotlivými geny, dědičností a variabilitou v populaci. [11]

Určitý segment DNA nazýváme lokus. Lokus může být segment několika genů, jeden specifický gen ale i jen jedna báze. Alternativní verze na lokusu se nazývají alely. Ve většině genů máme jednu rozšířenou alelu, která se objevuje ve více než polovině populace a říká se jí běžná alela (wild-type, common allele). Alela na lokusu jiná, než běžná alele je varianta (mutace). Pokud se na konkrétním lokusu projeví dvě a více běžných alel (podle konvence frekvence> 1 %) říkáme, že se projevil polymorfizmus. [4]

Při srovnání dvou nepříbuzných jedinců, bude jejich DNA z 99,5 % identická (podle některých zdrojů z 99,9 %). Variabilita lidského genomu mezi dvěma jedinci tedy činí asi 0,01-0,05 %. Toto na první pohled malé procento způsobuje viditelné odlišnosti lidí i závažné poruchy. [4]

Oproti tomu genetická variace v populaci je velmi běžná a každý jednotlivec má minimálně 5 milionů variant. Může se jednat o varianty objevující se ve vysoké frekvenci ve stejné populaci nebo o vzácné varianty. [4]

Některé typy variant jsou uvedeny dále v následujících podkapitolách, nejedná se však o kompletní výčet všech typů variant. V této práci jsou varianty rozdělené podle kritérií: velikost varianty, způsob vzniku, a podle důsledku na fenotyp (fyzický projev genotypu) organismu.

2.2.1 Typy variant podle velikosti lokusu

Velikost ovlivněného úseku mutací může značně ovlivnit, jak vážný účinek bude mutace mít. Mutace podle velikosti rozdělujeme na tři základní typy:

1. Chromozomální mutace – změna v počtu chromozomů.
2. Strukturní mutace chromozomu – Změna velké části chromozomu, jeho přeskupení nebo zkopírování části.

3. Změna krátké sekvence od 1 báze po 100 kb.

Varianty o jedné bázi se označují jako SNVs (jednonukleotidové varianty). V lidském genomu se objevují ve velké míře. Počet SNV v genomu člověka se pohybuje kolem 4 000 000 – 5 000 000. Zároveň ale analýza dat z databáze The Human Gene Mutation Database ukazuje že 34 % variant způsobující nemoc jsou varianty větší než SNVs. [13]

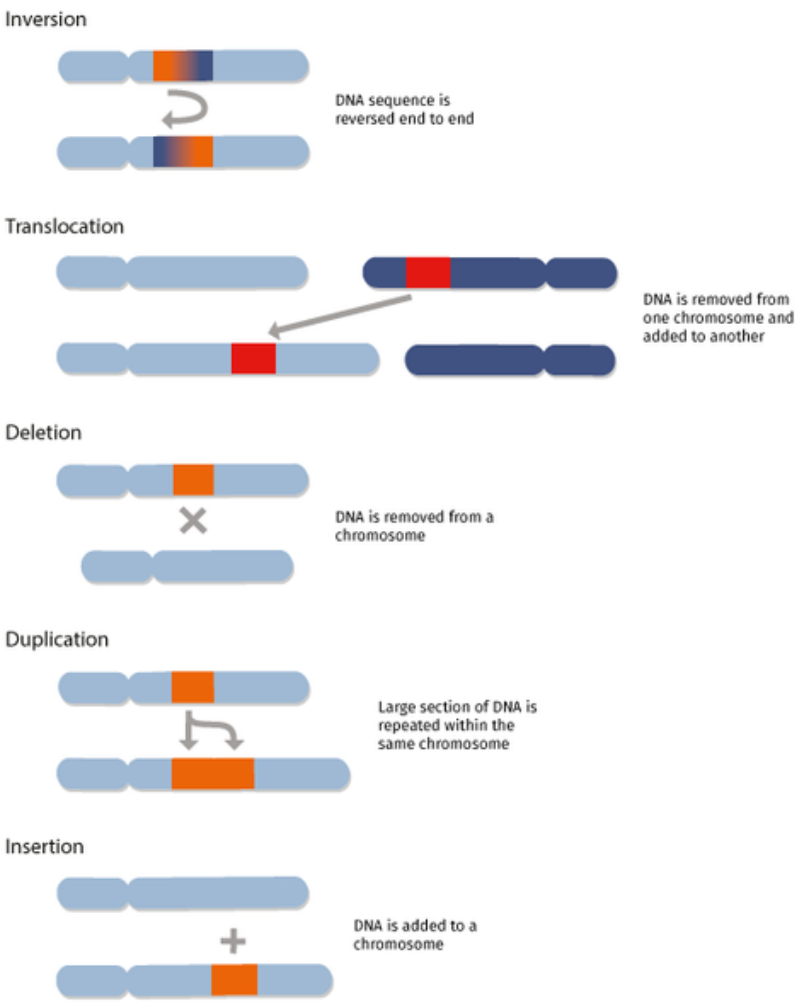
2.2.2 Typy variant podle způsobu vzniku

Varianty můžou vznikat následujícími způsoby:

1. záměna
2. inzerce
3. delece
4. inverze
5. duplikace
6. translokace

Záměna báze je změna sekvence na určitém lokusu. Inverze je otočení sekvence zrcadlově. Změna bází a inverze neovlivní při translaci další část genomu. Oproti tomu inzerce a delece mění počet nukleotidů a může posunout sekvenci tak, že se sekvence přeloží do úplně jiných aminokyselin.

Vizualizace typů variant inverze, translokace, delece, duplikace a inzerce jsou vidět na Obrázek 2.6



Obrázek 2.6 Zobrazení typů variant: inverze, translokace, delece, duplikace a inzerce. [14]

2.2.3 Typy variant podle důsledku

1. Missense varianta – mění kodon a tím mění aminokyselinu ve výsledném proteinu.
2. Nonsense variant – mění kodon na stop kodon (ukončuje translaci proteinu).
3. Synonymní (tichá) varianta – překládá stejný protein jako běžná alela.
4. Posunové (frameshift) mutace – vznikají delecí nebo inzercí. Dochází u nich k posunu kodonů a tím se překládají jiné aminokyseliny. [15]

Reálný důsledek varianty na fenotyp však můžou ovlivnit i další faktory, např. místo výskytu varianty (kódující či nekódující části genomu).

2.2.4 Fenotyp a genotyp v populačních databázích

Hlavním důvodem, proč vznikají populační databáze je cíl zjistit které varianty jsou pro určité populace běžné a které se vymykají, a tedy mohou být důvodem projevu nemoci či jiného jevu.

Populace se rozdělují podle fenotypu, což jsou viditelné vlastnosti a znaky organismu v našem případě člověka. Fenotyp je ovlivněn genotypem organismu, kde genotyp je soubor všech alel organismu. Fenotyp je ovlivněn i vlivem prostředí. [16]

V populačních databázích se fenotypem nemyslí pouze fyzické projevy genotypu, ale označují se tak i údaje typu národnost, přesnější geografické začlenění, informace o rodičích a další.

Příkladem populace je např. populace žen, populace určitého regionu nebo populace lidí s určitou chorobou. [16]

Jestli nebo jak výrazně se projeví genotyp na fenotypu člověka krom jiného určuje interakce alel na pozorovaném genu. Místa v genomu se dvěma variantami alel se nazývají diploidní a místům v genomu s pouze jednou variantou alely se říká haploidní. Autozomy a pohlavní chromozomy ženy jsou diploidní oproti tomu pohlavní chromozomy muže jsou haploidní. Pokud jsou obě alely stejné na sledovaném lokusu, říkáme, že se jedná o homozygota. Pokud jsou alely různé jedná se heterozygota.

Alely téhož genu (lokusu) mají mezi sebou interakce tří typů:

1. Úplné dominance:
   - Projeví se vždy dominantní alely z dvojice a recesivní alely se projeví pouze pokud jsou jako homozygoti na genu.
   - Příkladem je gen PAH (phenylalanine hydroxylase) kódující enzym fenylalaninhydroxylázu.
2. Neúplná dominance:
   - Pokud se objeví na genu heterozygot neprojeví se fenotyp ani jedné varianty, ale vzniká nový fenotypový znak vzniklý z fenotypu dominantní alely a potlačeného fenotypu recesivní alely.
3. Kodominance:
   - U heterozygotů se obě alely projevují rovnocenně a paralelně.
   - Příkladem je gen kódující antigeny krevního systému AB0 (v populaci jsou 3 typy alel – A, B, 0, ale vytvářejí fenotypy A, B, AB, 0).

V místech pohlavních chromozomů, kde muži nemají dvě varianty alel se projeví vždy ta alela, která je obsažená v genu nezávisle na tom, jestli se jedná o recesivní nebo dominantní alelu. [17]

2.3 Sekvenování

Pro studium genomiky a genetiky bylo převratné vytvoření metod umožňující čtení a zpracování DNA. Metody sekvenování se dělí na klasické metody sekvenování a sekvenování nové generace. Mezi klasické metody patří Maxam-Gilbertova a Sangerova metoda. Sekvenování nové generace se dělí na druhou a třetí generaci sekvenování.

2.3.1 Historie a vznik prvních metod sekvenování

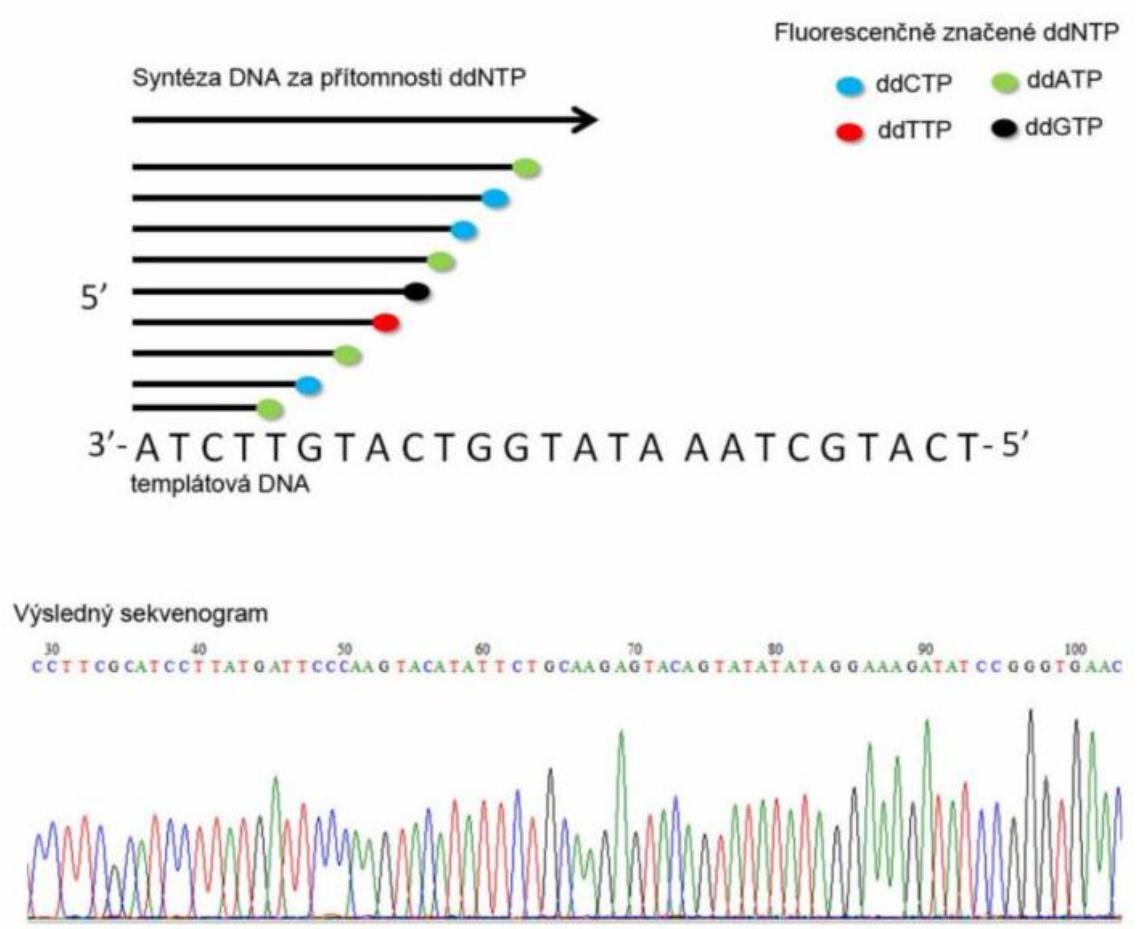
Vývoj sekvenačních metod započal u sekvenování prokaryotické RNA (hlavně tRNA a rRNA). Důvodů, proč se místo DNA začalo s vývojem sekvenačních metod u RNA, je několik. Za prvé se u ní nemusí řešit problematický komplementární řetězec, je často o poznání kratší než DNA a v době vývoje metod byly už známe enzymy pro štěpení RNA. V počátcích sekvenování se kromě RNA sekvenovaly i výsledné proteiny. [18]

V počátcích známé techniky analytické chemie umožňovaly zjistit pouze poměr jednotlivých nukleotidů obsažených v sekvenované oblasti RNA. První sekvence všech nukleotidů byla získána v roce 1965 zásluhou Roberta Holleyho a jeho kolegů. Jednalo se o alanin tRNA Saccharomyces cerevisiae. Prvními osekvenovanými úseky DNA byly přečnívající 5´ konce bakteriofága λ na které se navázaly radioaktivní nukleotidy. [18]

V roce 1977 vznikly nezávisle na sobě dvě metody sekvenování DNA, Maxam-Gilbertova a Sangerova metody, dnes známé jako klasické metody sekvenování. Sangerova metoda (v kapilárovém provedení viz dále) má své využití v klinické praxi dodnes. [19]

Sangerova metoda využívá proces replikace DNA a v klasické verzi na začátek sekvenováné už jednořetězcové DNA naváže komplementární radioaktivně značený primer v délce 15-25 bp. Dále se podle předlohy syntetizuje DNA ve čtyřech zkumavkách. V každé zkumavce je mnoho kopií předlohy, 4 typy dNTP (dATP, dCTP, dGTP a dTTP) a vždy jedna varianta dideoxynukleotidu (ddATP, ddCTP, ddGTP nebo ddTTP). Syntetizují se dNTP a v místě kde se náhodně začlení do syntetizovaného řetězce ddNTP místo dNTP se syntéza zastaví, protože ddNTP nemají OH skupin. Díky tomu vznikají různě dlouhé řetězce a délka řetězce určuje pozici příslušného ddNTP. Délka řetězce se určuje pomocí elektroforézy v polyakrylamidovém gelu. [19]

V modifikované verzi jsou ddNTP fluorescenčně značeny (umožňuje zpracování jen v jedné zkumavce) a analyzovány pomocí kapilární elektroforézy. Výsledkem Sangerovy metody je sekvenogram. Výsledná syntéza DNA a výsledný sekvenogram je vidět na Obrázek 2.7. Klasické metody jsou dodnes vhodné pro sekvenování jednotlivých genů a variant. Sangerova metoda sekvenování má chybovost kolem 1,5 %. [19]



Obrázek 2.7 Výsledek modifikované Sangerovy metody [19]

2.3.2 Metody druhé generace sekvenování

Sekvenování nové generace (NGS) mezi které se řadí jak druhá, tak třetí generace sekvenování, začaly vznikat koncem 90 let 20. století. Důvodem byla snaha o zefektivnění (rychlejší a levnější) sekvenování. [20]

Sekvenátory druhé generace všechny pracují na stejném principu. Proces začíná přípravou knihovny. Příprava knihovny zahrnuje naštěpení vstupního materiálu na kratší fragmenty. Jednotlivým fragmentům se dále upravují enzymaticky konce a při zpracování více vzorků najednou se provádí i ligace adaptorů. Adaptory umožňují rozeznat ke kterému vzorku fragment patří. Následně probíhá amplifikace knihovny, kde se vytvořená knihovna namnoží pomocí PCR (polymerázová řetězová reakce, polymerase chain reaction). Amplifikace knihovny je nezbytným krokem druhé generace sekvenování a u každé sekvenační platformy se odlišuje metoda provedení PCR. [2]

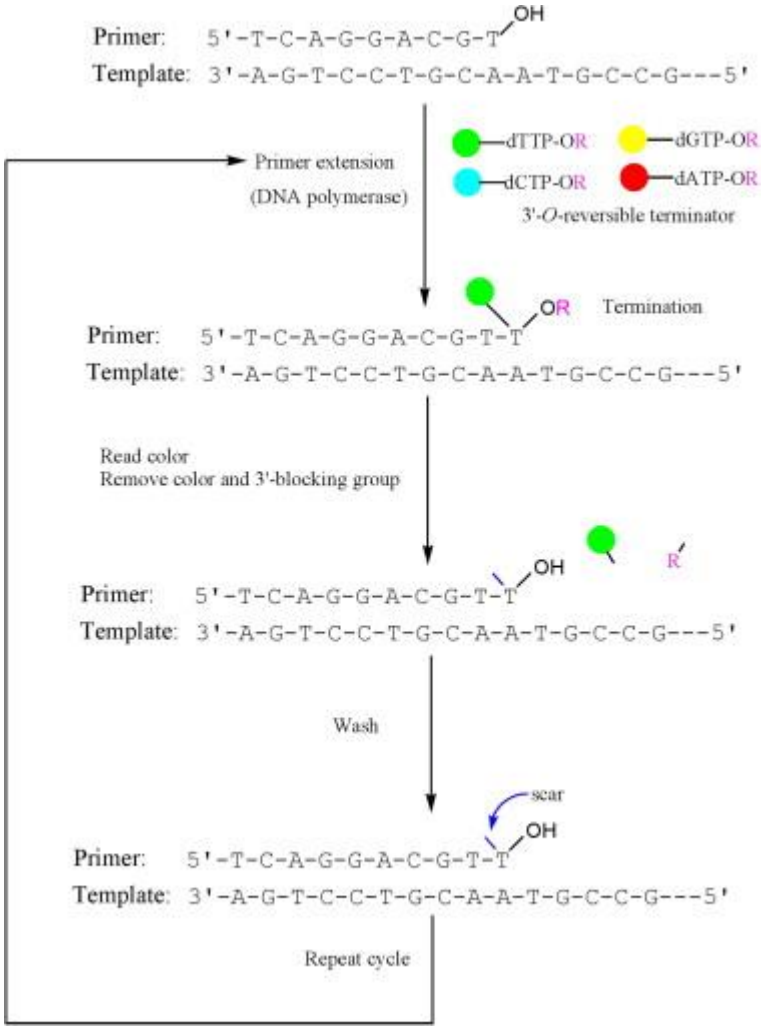
Volitelným krokem sekvenování je využití metod obohacení DNA. Ve chvíli, kdy nechceme sekvenovat celý genom se můžeme díky metodám obohacení DNA soustředit pouze na vybrané úseky (cílené sekvenování, sequence capture, target capture). [2]

Ve chvíli, kdy je připravená a namnožená knihovnu DNA, se přichází k samotnému sekvenování. Máme dva základní type sekvenování. Jedná se o sekvenování ligací také známe jako sekvenování hybridizací a sekvenování syntézou. [2]

Sekvenování ligací je založené na hybridizaci fluorescenčně značených krátkých oligonukleotidových prób a následně dochází k uvolňování fluoroforu odpovídající specifické bázi. [2]

Sekvenování syntézou postupně přidává nukleotid jako při syntéze DNA a tím se vyvolá specifický signál (fluorescence, změna pH, světlo), který se následně pomocí sekvenátoru detekuje a analyzuje viz kapitola 2.4.1. Sekvenování syntézou může buď fungovat na principu přidání jednoho nukleotidu nebo využívat metod cyklické reverzibilní terminaci. [2]

Cyklická reverzibilní terminace je typ sekvenování syntézou. Čtení sekvence probíhá postupným prodlužováním primeru na templátu. Začíná imobilizací templátu a primeru. Pokračuje prodloužením primeru o jednu nukleotidovou bázi, detekováním signálu fluorescence a následnou terminací. Terminace uvolní fluorescenční značku a 3‘ blokující skupinu. 3‘ skupina blokovala syntézu dalšího nukleotidu. Následuje očištění reakce a cyklus pokračuje navázáním další nukleotidové báze, dokud se nepřečte celá sekvence DNA. Celý tento proces je zobrazený na Obrázek 2.8. [21]



Obrázek 2.8 Vykreslená sekvenace syntézou s využitím cyklická reverzibilní terminace. [21]

Všechny metody druhé generace jsou schopny paralelního sekvenování, a tedy zpracování tisíců a milionů rozdílných molekul DNA najednou. Proto jsou metody druhé generace sekvenování rychlejší a levnější než Sangerova metoda. Naopak jejich nevýhoda je že produkují velmi malá čtení (ready). Read znamená sekvenci DNA z jednoho fragmentu vytvořeného během přípravy knihovny. Velikost jednotlivých čtení se pohybuje kolem 100-500 bází. Sangerova metoda oproti tomu produkuje čtení o velikosti přibližně až 1000 bází. [22]

Příkladem firem vyvíjející sekvenační platformy jsou Ion Torrent, Illumina, SOLiD, GenapSys, BGI Group.

Níže je popsána pouze platforma Illumina, protože právě na ní byla sekvenována genetická data zpracovávaná v této práci.

Platforma Illumina využívá sekvenování syntézou metodou cyklické reverzibilní terminace popsané výše a zobrazené na Obrázek 2.8. Knihovna DNA obsahuje rozštěpené fragmenty dlouhé ~300 bp. Každý konec je ligován odlišným adaptorem, který se využívá při můstkové amplifikaci. Proces amplifikace se několikrát opakuje,

dokud nevnikne dostatečný počet kopií původních fragmentů (tisíce kopií). Sekvenace probíhá s pomocí odlišně fluorescenčně označených nukleotidů. V každém cyklu sekvenace se na templátové fragmenty knihovny DNA naváže jeden nukleotid, který vyšle fluorescenční signál. Tento signál se detekuje laserovým paprskem a zaznamenává citlivou kamerou. Cykly sekvenace se opakují asi 100x až 300x a počet se liší podle délky čtení (fragmentů). [2]

2.3.3 Třetí generace sekvenování

U třetí generace sekvenování se DNA knihovna nemnoží, jako předloha se používá jediná molekula DNA a je možnost číst až 10 i více tisíc bází během jedné analýzy.

Třetí generece neprovádí amplifikaci DNA a tím se ztrácí se limitace související s použitím PCR metod jako jsou chybná inkorporace nukleotidů polymerázou, tvorba chiméry a alelické výpadky (preferenční amplifikace jedné alely) způsobující umělé volání homozygotů. [23]

Hlavními nevýhodami tření generace jsou nízká přesnost a omezené množství metod a technologií pro bioinformatickou analýzu výstupů třetí generace. I přes tyto nevýhody nepřestává vývoj technologií třetí generace. Postupně se zvyšuje přesnost metod a jejích budoucnost je stále otevřená. [24]

V současné době jsou 2 hlavní komerční řešení.

První řešení využívá jedné cirkulární molekuly nazývané SMRT (single-molecule real-time) bell na kterou se vážou fluorescenčně označené nukleotidy. Bylo představeno firmou Pacific Biosciences v roce 2011 sekvenátorem PacBio RS II. [23]

Další komerčně úspěšnou firmou v oblasti třetí generace sekvenování je Oxford Nanopore. Jejich řešením je protahování molekuly DNA mikroskopickým pórem na syntetické membráně a detekováním elektrických změn při průchodu nukleotidů. Sekvenátor MinION využívající řešení firmy je Oxford Nanopore začal byl v roce 2015 komerčně dostupný. [23]

2.3.4 Aplikace NGS (nové generace sekvenování) sekvenování

Metody a technologie popsané výše v kapitolách 2.3.2 a 2.3.3 umožňují sekvenování celého exomu a genomu.

Sekvenování genomu vrací všechny báze v sekvenovaném genomu. Oproti tomu sekvenování exomu reportuje pouze protein kódující regiony. Další možností využití NGS je targetované sekvenování. Targetované sekvenování se zaměřuje na sekvenování genů asociovaných s léčenou nemocí pacienta. [25]

Čím menší oblast genomu se sekvenuje, tím je sekvenování a jeho následné zpracování rychlejší a levnější. Proto je v klinické praxi běžné targetované sekvenování.

Celogenomové sekvenování se však stává cenově dostupnější a studie v posledních letech ukázali jeho přínosy. [25]

Celogenomové sekvenování pomáhá ve výzkumu odhalit zcela nové varianty. Má významný význam v diagnóze vzácných jedno genových poruch u dětí jako je cystická fibróza nebo srpkovitá anémie. Také pomáhá s odhalením somaticky vznikajících variant v tumorových buňkách. [25]

Existují i další specifické aplikace NGS sekvenování jako je například jedno buněčné sekvenování (Single-cell sequencing). Nové aplikace NGS přinášejí nové možnosti výzkumu a v budoucnu mohou být důležitou součástí bioinfromatické praxe. [26]

2.4 Bioinformatická analýza dat druhé generace sekvenování

Bioinformatické zpracování dat ze sekvenátorů druhé generace sekvenování se dělí na primární, sekundární a terciální analýzu. Principy analýz se mezi jednotlivými platformami neliší až na malá specifika platforem. V této práci však budou jednotlivé analýzy popsány na základě platformy Illumina. Důvodem je, že na této platformě byli sekvenovány data zpracovávaná v rámci této práce.

2.4.1 Primární analýza

Primární analýza se skládá z detekce sirových signálových dat ze sekvenátoru, z volání bází (určení báze A,T,C,G) a ze skórování kvality bází. Výstupem primární analýzy bývá FASTQ soubor.

Signál u Illuminy je fluorescenční záření detekované laserem. Volání bází využívá obrazových snímků a z nich se měří vlnová délka a intenzita signálu vyvolaná syntézou báze viz kapitola 2.3.2. Vlnová délka a intenzita signálu určuje nukleotidovou bázi.

Kontrola kvality přiřazuje bázím Phred skóre. Phred skóre je vypočítané z pravděpodobnosti chyby viz vzorec 2.1,

$$Q = -10 \log 10\, E \qquad (2.1)$$

kde E je pravděpodobnost chyby a v Phred skóre (Q) je chyba interpretovaná jako záporný logaritmus. Čím vetší je hodnota Phred skóre, tím se snižuje pravděpodobná chyba a zvyšuje pravděpodobná přesnost čtení báze (kvalita určení báze). [27]

Například Phred skóre o hodnotě 10 ukazuje na chybovost 10 % a přesnost 90 %. Naopak Phred skóre hodnoty 60 znamená chybovost 0.0001% a přesnost 99.9999%. [27]

Existuje několik nástrojů pro kontrolu kvality. Těmito nástroji jsou například NGS QC toolkit, QC-Chain a FastQC. Phred skóre se dále využívá pro filtraci a trimming.

Filtrování odstraňuje ready o špatné kvalitě bází (nízké hodnotě Phred skóre). Dále odstraňuje krátké ready, které by se mapovali na více míst v genomu v rámci sekundární analýzy a snižovali by kvalitu mapování viz dále kapitola 2.4.2.

Po filtrací readů je dalším krokem demultiplexing. Sekvenátory druhé generace umožňují paralelní zpracování více vzorků. Přiřazení readů správným vzorkům je provedeno právě již zmíněným krokem demultiplexing. Výsledek demuplexingu je uložením čtení do separovaných souborů podle čárových kódů přiřazeným pro každý vzorek. Proces demultiplexingu je specifikovaný od výrobce platforem sekvenátorů.

Po demultiplexingu přichází poslední fáze primární analýzy. Trimming odstraňuje adaptéry sekvencí přiřazené během přípravy knihovny. Může také odstraňovat báze se špatnou kvalitou z konce readů. Nástroje pro trimming jsou BTrim, IeeHom, AdapterRemoval, a Trimmonatic. [23]

2.4.2 Sekundární analýza

Sekundární analýza se skládá ze dvou částí. První je získání celé původní sekvence vzorku pomocí mapování readů (alignment) či de novo assembly. Druhou částí sekundární analýzy je variant calling (volání variant). [23]

De novo assembly je metoda pro sestavení genomu bez předlohy a je užitečná u organismů ke kterým nemáme referenci. Alignment naopak využívá referencí, kde mapování je klasickou úlohu bioinformatiky. Jelikož se při aligment mapují krátké ready oproti velmi dlouhé sekvenci reference je potřeba při softwarovém zpracování využívat heuristického přístupu. Používané reference lidského genomu jsou hg19 nebo hg38 a existuje více jak 60 nástrojů pro mapování readů. [23]

Výstupním souborem alignmentu či de novo assembly je soubor SAM/BAM. BAM soubor je zkomprimovaná binární verze SAM souboru. Data uložená v těchto souborech jsou potřeba před variant callingem upravit a provést takzvaný Post-Alignment Processing. Jako první se odstraňují duplicitní ready vytvořené sekvenováním jedné identické molekuly DNA. Dalším krokem je intensive local realignment což znamená že se naleznou podezřelý úseky readů které se algoritmus pokusí přemapovat. Posledním krokem je přepočítání skóre kvality bází podle metrik zarovnání. Nástroji pro Post-Alignment Processing například jsou SAMtools, Genome Analysis Toolkit (GATK) a Picard. [23]

Volání variant je část sekundární analýzy sloužící pro detekci variant na sekvenovaném genomu vůči referenci. Vstupním souborem volání variant je předzpracovaný BAM soubor a výstupním je buď gVCF soubor obsahující informace o každé bázi genomu sekvenovaného vzorku nebo ve většině případů VCF soubor obsahující informace pouze o těch bázích, které se lišili od reference. VCF soubory jsou několikanásobně menší než gVCF soubory. Soubory VCF a gVCF jsou detailně

popsané v kapitole **Chyba! Nenalezen zdroj odkazů.**. Nástroji pro volání variant jsou například samtools nebo GATK. [23]

2.4.3 Terciální analýza

Terciální analýza má za úkol nalezeným variantám ve VCF souborech dát biologický smysl. [23]

Prvním krokem je anotace variant. Dalšími kroky terciální analýzy jsou filtrace, prioritizace a případná vizualizace vybraných variant. [23]

Anotace variant a nástroje pro anotaci jsou rozepsané v kapitole 2.5. Další kroky terciální analýzy: filtraci, prioritizaci a vizualizace variant zajišťují nástroje popsané v kapitole 2.9. Jedná se o nástroje umožňující interpretaci variant a jejich částí může být anotace variant provedena implementací anotačních nástrojů.

Populační databáze mohou být pomocníkem výše zmíněných fází terciální analýzy, jelikož anotační nástroje často anotují frekvence varianty v odlišných populacích.

2.5 Anotace variant a přehled anotačních nástrojů

Anotace variant je součástí terciální bioinformatické analýzy genetických dat, jak již bylo zmíněno v kapitole 2.4.3.

Anotace variant přidává k variantám funkcionální informace standardizované společností Human Genome Variation Society (HGVS), frekvence z populačních databází, přiřazuje variantě ID, lokalizuje její polohu v genu a další.

Informace o tom, jak varianta ovlivňuje transkript se nazývá transkript anotace a u liského genomu pro ni slouží především dvě transkriptové databáze: GENCODE, Reference Sequence (RefSeq). Reportování variant s využitím nomenklatury od už zmíněné společnosti Human Genome Variation Society (HGVS) je také založené na anotaci transcriptu a popisuje funkcionální informace o variantě. Výběr isoformy a verze databáze pro transkript anotaci může výrazně ovlivnit výsledky anotace. Proto anotační nástroje často umožňují volbu z více databází pro transkript anotaci. [28]

Pro anotaci variant existuje nepřeberné množství nástrojů a v řadě věcí jsou si navzájem podobné. Dále této v kapitole jsou popsané nástroje VEP, Nirvana, ANNOVAR a SnpEFF. Za zmínku však stojí i další anotační nástroje: Alamut®, VarAFT, 2.5.4SnpEff nebo například FAVOR. Anotace variant je velmi důležitá fáze interpretace variant, a proto neustále vznikají další nástroje pro přesnější a lepší anotaci variant.

2.5.1 VEP

Variant Effect Predictor (VEP) je anotační nástroj ze souboru nástrojů Ensembl Tools. VEP lze použít buď skrz webové rozhraní nebo příkazovou řádku. [28]

VEP umožňuje požít obě transkriptové databáze GENCODE, RefSeq.

Výhody:

1. Jednoduše implementovatelný v rámci Hail nástroje (popsaný v kapitole 2.9.4).
2. Podporuje až 5000 transkriptů organismů.
3. Umožňuje cloudové využití.
4. Podporuje API přístup pomocí Perl nebo REST.
5. Podporuje na vstupu formáty VCF, rsID, HGVS

Nevýhody:

1. Oproti ostatním nástrojů je anotace celo genomových dat pomalejší
2. Neumožňuje Nonsense mediate decay assessment
3. Nepodporuje BED soubory

2.5.2 Nirvana

Nirvana lze spouštět jako samostatný balíček, jako AWS lambda funkce nebo být integrovaná do jiného softwaru jakým je například Hail. Umožňuje požít obě transkriptové databáze GENCODE, RefSeq . [29]

Výhody:

1. Jednoduše implementovatelná v rámci Hail nástroje (popsaný v kapitole 2.9.4).
2. Umožňuje rychlejší anotaci než VEP

Nevýhody:

1. Náročnější implementace než VEP

2.5.3 ANNOVAR

Je nástroj zaměřený především na přístup pomocí příkazové řádky. Umožňuje anotaci podle genu, pozice a také identifikovat varianty dokumentované v databázích variant. Umožňuje požít obě transkriptové databáze GENCODE, RefSeq. [30]

2.6 Úvod do big data

Big data jsou charakteristické:

1. velikostí
2. rychlostí se kterou vznikají
3. jejich rozlišností, ne vždy jsou strukturovaná a schopná být uložená do relační databáze
4. nemusí být 100 % přesná
5. jejich hodnota vzniká až zpracováním dat

V dnešním světe je několik odvětví produkující big data o velikosti v rámci PB (petabytů) a více. Jedná se například o sociální sítě, bankovnictví, vesmírná data, zdravotnictví a další. Například na YouTube platformě vznikne ročně 1–2 EB (exabytů) dat a podle statistiky z roku 2020 Facebook generuje 4 PB dat za jediný den. [31]

S vývojem sekvenačních technologií nové generace (druhá a třetí generace sekvenování viz kapitola 2.3.2 a 2.3.3), celkovému zlevnění sekvenování a příchodem celogenomového sekvenování se zvětšuje velikost dat, které se generují při zpracovávání genetických informací. Data vygenerovaná zpracováním genetických informací se blíží 40 EB za rok a 40x překonávají data generovaná na platformě YouTube. [31]

2.7 Nástroje pro zpracování big data

Nástroje pro zpracování big data můžeme rozdělit do 4 kategorií:

1. úložiště
2. mining (těžba dat – získávání užitečných informací z čistých dat)
3. analytické nástroje
4. vizualizační nástroje [32]

V rámci naší práce se využívají nástroje pro ukládání dat a analytické nástroje.

2.7.1 Úložiště pro big data

Pro ukládání big data můžeme použít NoSQL databáze jako je například MongoDB či Cassandra. Další možností jsou nástroje (frameworky) sloužící pro ukládání dat jako například Apache Hadoop. [32]

Všechny výše zmíněné nástroje jsou open source (zdarma volně dostupné).

MongoDB je NoSQL databáze. Ukládá data v JSON souborech namísto tabulek a má možnosti, které relační databáze postrádají (dynamické dotazování, sekundární indexování, jednoduchou agregaci dat). [32]

Cassandra využívá clusterů a decentralizuje tím data, podporuje ACID (Atomicity, Consistency, Isolation, and Durability) databáze, podporuje integraci Hadoop, Apache Hive & Apache Pig. Zvládá zpracování několika souborů dat v reálném čase. Podle potřeby se může přirozeně rozšiřovat jak v rámci uživatelů, tak v rámci dat. Pro komunikaci s databází se využívá CQL jazyk. [32]

Apatche Hadoop je framework postavený na jazyku Java. Hlavní modul je Hadoop Common, který obsahuje základní nástroje podporující další moduly. Modul frameworku fungující jako úložiště je HDFS (Hadoop Distributed File System). HDFS funguje jako distribuovaný souborový systém a ukládá data do několika nodů v jednom clusteru pro prevenci ztráty dat. Další části frameworku umožňují například managování výpočetních zdrojů (YARN) či paralelní zpracování velkých datasetů (MapReduce). Zpracování dat na hadoopu využívá disk-based computing. Disk-based computing výpočty ukládá na disk a nevyužívá RAM pamětí. [33]

2.7.2 Analytické nástroje

Analytických nástrojů pro zpracování big data je nepřeberné množství. Apache Kafka (analytický nástroj pro streamovací služby jako je Netflix), Splunk (analytický nástroj pro analýzu dat generovaných webovými stránkami, aplikacemi, senzory a dalšími zařízeními), KNIME (analýza, reportování, integrace velkých datasetů). [32]

Nejrozšířenějším nástrojem v dnešní době je však Apache Spark. Je to open source a umožňuje In-Memory Computing (IMC). [34]

In-Memory Computing znamená že ukrádá data v RAM pamětích skrz počítačové clustery a umožnuje v nich paralelní zpracování dat. Hlavní spouštěný program je v terminologii Apache Spark nazvaný jako Driver program. Driver program obsahuje SparkContext objekt. Ten řídí procesy na jednotlivých clusterech pomocí komunikace s řadičem clusterů (Cluster Manager). Jednotlivé clustery mají uzly zpracování (Worker nodes) na kterých vznikají Executory (spuštěné procesy spolu s daty). [34] [35]

Díky In-Memory Computing je Spark výkonnější v iterativních výpočtech oproti Hadoop frameworku. [34]

Spark umožňuje správu paměti, obnovu chyb, plánování, distribuci a monitorování úloh a interakci se systémy úložiště. Spark ale nemá vlastní datové úložiště. Proto se často používá v kombinací s Hadoop nebo jiným úložištěm pro big data. Spark podporuje API pro několik programovacích jazyků: Java, Scala, R a Python. [34]

2.8 Popis vcf a gvcf souborů

Tato práce pracuje na vstupu s daty ze sekundární analýzy, s gvcf soubory. Vcf a gvcf soubory obsahují informace o variantách, jak už bylo zmíněno výše v kapitole 2.4.2.

Vcf i gvcf soubor se rozděluje na část hlavičky a část se záznamy. Hlavička obsahuje následující informace [36]:

- fileformat – verze formátu souboru
- INFO – popis významu zkratek uložených v INFO sloupci
- FILTER – informace o použitých filtrech
- FORMAT – popis významu zkratek souvisejících s genotypem uložených ve FORMAT sloupci
- ALT (označení jakékoliv alternativy v souboru)
- contig – informace o označení chromozomů a jejich délce v záznamech souboru
- reference – typ a verze použité reference

Po hlavičce následuje jeden řádek popisující jednotlivé sloupce tabulky se záznamy. Vcf soubor obsahuje tyto fixní sloupce: CHROM, POS, ID, REF, ALT, QUAL, FILTER, INFO. [36]

Záznamy v souborech vcf a gvcf obsahují následující informace.

- CHROM – označení chromozomu
- POS – přesnou pozici
- ID – id varianty (volitelný identifikátor varianty)
- REF – alelu reference
- ALT – alelu alternativy (varianty)
- QUAL – skóre pravděpodobnosti výskytu polymorfizmu
- FILTER – použité filtry
- INFO – obsahuje anotaci k variantě, nemusí být ve vcf souboru
    - DP – počet filtrovaných čtení (readů) pro daný lokus (bázi)
    - AF – alelická frakce je proporce readů podporujících variantu
    - a další…
- FORMAT – obsahuje informace o genotypu
    - GT – genotyp (formát 0/0 - homozygot na referenci, 0/1 - heterozygot, 1/1 – homozygot na alternativě)
    - PL – pravděpodobnost možných genotypů (diploidní organismy mají 3 možné genotypy)
    - GQ – pravděpodobnost správného výběru GT (genotypu), vypočítává se z druhé nejnižší a nejnižší hodnoty PL ale jeho maximální hodnota je omezená na 99.
    - AD – počet nefiltrovaných readů podporující nalezené alely
    - DP – počet filtrovaných readů podporující nalezené alely
    - a další…

Ukázka vcf souboru je vidět na Obrázek 2.9

```
##fileformat=VCFv4.0
##fileDate=20090805
##source=myImputationProgramV3.1
##reference=1000GenomesPilot-NCBI36
##phasing=partial
##INFO=<ID=NS,Number=1,Type=Integer,Description="Number of Samples With Data">
##INFO=<ID=DP,Number=1,Type=Integer,Description="Total Depth">
##INFO=<ID=AF,Number=.,Type=Float,Description="Allele Frequency">
##INFO=<ID=AA,Number=1,Type=String,Description="Ancestral Allele">
##INFO=<ID=DB,Number=0,Type=Flag,Description="dbSNP membership, build 129">
##INFO=<ID=H2,Number=0,Type=Flag,Description="HapMap2 membership">
##FILTER=<ID=q10,Description="Quality below 10">
##FILTER=<ID=s50,Description="Less than 50% of samples have data">
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype Quality">
##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Read Depth">
##FORMAT=<ID=HQ,Number=2,Type=Integer,Description="Haplotype Quality">
#CHROM POS     ID        REF ALT    QUAL FILTER INFO                              FORMAT      NA00001        NA00002        NA00003
20     14370   rs6054257 G      A       29   PASS   NS=3;DP=14;AF=0.5;DB;H2           GT:GQ:DP:HQ 0|0:48:1:51,51 1|0:48:8:51,51 1/1:43:5:.,.
20     17330   .         T      A       3    q10    NS=3;DP=11;AF=0.017               GT:GQ:DP:HQ 0|0:49:3:58,50 0|1:3:5:65,3   0/0:41:3
20     1110696 rs6040355 A      G,T     67   PASS   NS=2;DP=10;AF=0.333,0.667;AA=T;DB GT:GQ:DP:HQ 1|2:21:6:23,27 2|1:2:0:18,2   2/2:35:4
20     1230237 .         T      .       47   PASS   NS=3;DP=13;AA=T                   GT:GQ:DP:HQ 0|0:54:7:56,60 0|0:48:4:51,51 0/0:61:2
20     1234567 microsat1 GTCT   G,GTACT 50   PASS   NS=3;DP=9;AA=G                    GT:GQ:DP    0/1:35:4       0/2:17:2       1/1:40:3
```

Obrázek 2.9 Ukázka vcf souboru [37]

Gvcf je formát založený na vcf formátu. Obsahuje stejné informace a jeho specifikace je stejná jako pro vcf. Hlavní rozdílem je, že gvcf obsahuje záznamy o všech bázích genomu jedince. Cílem gvcf formátu je mít v souboru zastoupeny informace o sekvenování všech bází i pokud se shodovaly s referencí. Na úkor toho se však výrazně zvětšuje velikost gvcf souboru oproti vcf formátu. [38]

2.9 Přehled nástrojů pro zpracování genetických variant uložených v souborech vcf/gvcf

V této kapitole budou rozebrány nástroje postavené na technologiích pro zpracování big data popsané v kapitole 2.7 a v kontrastu i nástroje založené na klasických relačních databázích jako je například SQLite.

SQLite sice není typickou technologií pro zpracování big data, ale její výhodou je jednoduchá syntaxe, jednoduchost sdílení, dostupnost a rychlost oproti jiným relačním databázím.

2.9.1 GATK

GATK umožňuje paralelní zpracování s využitím Apache Spark a nabízí specializované nástroje schopné zpracování FastQ souborů až po terciální analýzu (volání variant, jejich filtraci, anotaci a ohodnocení varianty). [39]

Výhody:

1. Rozsáhlá dokumentace detailně vysvětlující jednotlivé kroky GATK workflows a také detailně popisující jednotlivé metody a jejich výstupy.
2. Rozsáhlý nástroj umožňující veškeré kroky bioinformatické analýzy genetických dat.

Nevýhody:

1. Specializovaný na hledání a interpretaci jednotlivých variant a neumožňuje sjednocování vcf/gvcf souborů a následnou tvorbu datasetů.

2.9.2 GEMINI

GEMINI je framework zaměřený na anotování variant a zjednodušené prohlížení vcf souborů. Jedná se o zdarma dostupný open source framework postavený převážně na pythonu a části citlivé na výkon jsou napsány v C a C++. Framework načítá vcf soubor do SQLite databáze. V GEMINI databázi jsou varianty indexovány, a to umožňuje rychlé vyhledávání varianty. [40]

Výhody:

1. přenositelnost databáze
2. jednoduché SQL dotazování
3. jednoduché použití a přehledná dokumentace a aktivní podpora
4. zdarma dostupný open source

Nevýhody:

1. SQLite databáze (nedostačující pro práci s celými gVCF soubory)
2. omezené možnosti a funkce oproti nástroji Hail

2.9.3 GLOW

GLOW je nástroj pro zpracování genetických variant postavený na Apache Spark a Delta Lake. Umožňuje načíst data do Delta Lake úložiště, sjednotit genotypy skrz genotypové data, dělat statistické výpočty, kontrolu kvality dat, a asociační studie. Může také vytvořit znovupoužitelné pipeliny zpracování. Jedná se o nástroj velmi podobný Hailu jen s menšími možnostmi a funkcemi. Umožňuje trasformaci Hail MatrixTable do Spark DataFrame. [41]

Výhody:

1. Lze jednoduše spouštět na Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP) pomocí Databricks.
2. Má základní nástroje na zpracování vcf.
3. Umožňuje transformaci Hail MatrixTable do Spark DataFrame.
4. Zdarma dostupný open source.

Nevýhody:

1. Neefektivní sjednocování datasetů (více vcf do jednoho datasetu).
2. V době psaní této práce je poslední vydaná verze v1.2.1 ze dne 21.4.2022.
3. Chybí některé funkce, které Hail nabízí.

2.9.4 Hail

Hail je python knihovna zaměřená na analýzu genomických dat. Je založena na Apache Spark spolu s Hadoopem. Hail je zejména knihovna pro analýzu strukturovaných genetických dat. Je vhodná jak pro interpretaci jednotlivých vcf souborů a jednotlivých variant, tak pro tvorbu velkých datasetů o velikosti jako je gnomAD populační databáze (popsaná dále v kapitole 2.10). [42]

Výhody:

1. Modul vds byl vytvořený jako odezva na potřeby tvorby populačních databází jako je gnomAD a tím vhodný pro naše specifické řešení.
2. Velmi rozsáhle množství funkcí a metod pro práci a zpracování genetických dat.
3. Rozsáhlá dokumentace a přístupnost zdrojových kódů Hail.
4. Aktualizovaná a dále vyvíjející se knihovna – v době psaní této práce je poslední verze v0.2.116 vydaná dne 4.5.2023.
5. Volně dostupné diskusní fórum a aktivní podpora.
6. Zdarma dostupný open source.

Nevýhody:

1. Některé funkce a metody nemají ošetřené odlišnosti v záznamech vcf/gvcf souborů. Příkladem jsou:
    a. Haiploidní varianty mají v PL sloupci pouze dvě hodnoty místo tří jako diploidní záznamy.
    b. ChrM může mít specifický počet alel.
2. Rozsáhlost knihovny vede k pomalé křivce učení práce s knihovnou.

2.10 Populační databáze genetických variant

Populační databáze jsou důležitým nástrojem pro diagnostiku a studium genetických variant. Díky populačním databázím je možné rozlišit vzácnou variantu zapříčiňující vážnou nemoc a běžnou variantu pro konkrétní populaci. [43]

1000 Genomes project započal éru populačních databází a je základem pro velké množství studií a vědecká komunita tuto databází používá hojně i v dnešní době. Data se můžou zobrazit v prohlížeči projektu, v EnsEMBL nebo stáhnout pomocí FPT stránky. [43]

Další populační databází, která obsahovala 6500 Evropských a Africko-amerických jedinců, se nazývá Exome Sequencing Project. [43]

V roce 2014 vznikla databáze obsahující data 60 000 lidí s cílem vytvořit více diverzifikovanou databázi. Její název v té době byl Exome Aggregation Consortium, ale později se přejmenovala na Genome Aggregation Database (gnomAD). [43]

GnomAD obsahuje data jedinců z celého světa a této chvíli je dostupný ve třech verzích. Verze v2 reprezentuje 125,748 exomů a 15,708 genomů (zpracovaných podle reference GRCh37), v2.1 reprezentuje 10,847 genomů (zpracovaných podle reference GRCh37) a v3.1 76,156 genomů (zpracovaných podle reference GRCh38). [44]

Současně s dalším rozvojem databáze gnomAD vznikají populační databáze jednotlivých zemí. Příkladem je human genetic variation database (HGVD) obsahující data 1208 japonských jedinců a 287 588 SNV identifikované celo-exomovým sekvenováním. Databáze je dostupná na webu (grafické rozhraní je vytvořeno na JBrowse 1.4) a je založena na databázi PostgreSQL (relační databáze) a dostupná na Apache 2.2.3 serveru. [45]

3 Cíle práce

Cílem práce je navrhnout metody, vybrat technologie a vytvořit skripty sloužící pro vytvoření populační databáze.

Prvním krokem je seznámit se s oblastí genetiky, variability lidského genomu a seznámit se s důležitostí populačních databází.

Díky nabytým znalostem navrhnout a popsat konkrétní řetězec zpracování gvcf souborů, obsahují varianty lidského genomu. Tento řetězec musí obsahovat kroky nahrání dat do databáze, anotaci, filtraci a agregaci dat. Zároveň bude řetězec popisovat metody použité pro zpracování. Navrhnutý řetězec a metody jsou popsané v kapitole 4.

Dalším krokem je vybrat nástroje pro implementaci navrhnutého řetězce. Součástí je také výběr databáze. Volba nástrojů bude vycházet z provedené rešerše shrnuté v kapitolách 2.5, 2.7 a 2.9.

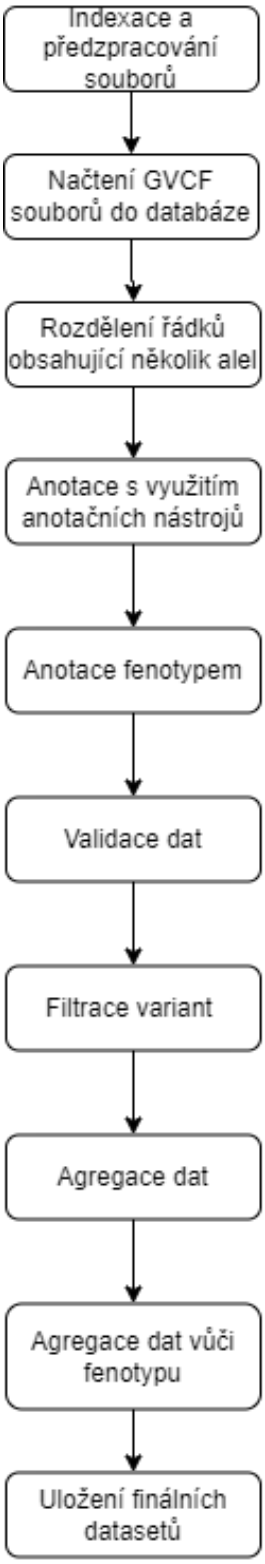
Posledním krokem je implementovat řetězec zpracování pomocí vybraných nástrojů. Součástí implementace je vytvoření prostředí, do kterého se nahrají gvcf soubory pro zpracování a potřebné scripty potřebné pro vytvoření finální populační databáze.

Výsledkem práce je přenositelné a reprodukovatelné prostředí v Dockeru, řetězec zpracování spolu se skripty a hotová databáze vytvořena z testovacích vzorků.

4 Návrh zpracování a metody

Tato kapitola popisuje návrh řetězce zpracování spolu s detailnějším popisem jednotlivých kroků a využitých metod. Dále se tato kapitola zabývá výběrem použitých nástrojů pro implementaci. Z vybraných nástrojů vyplývají softwarové a hardwarové požadavky.

4.1 Návrh řetězce zpracování



Obrázek 4.1 Diagram zobrazující návrh řetězce zpracování.

Na vstupu řetězce zpracování jsou gvcf soubory. Ty se jako první předzpracují a až poté načtou do databáze jako jeden dataset. Ve vzniklém datasetu se rozdělí řádky s více variantami, aby mohla následně proběhnout anotace dat. Po anotaci se validují a přepočítávají potřebné údaje. Dále se filtrují záznamy a na vyfiltrovaném datasetu se vypočítávají agregované údaje. Ve finále je dataset uložen. Jednotlivé kroky analýzy jsou vidět na diagramu na Obrázek 4.1.

4.1.1 Indexace a předzpracování souborů

Jako první se získají názvy všech souborů pro zpracování. Ze všech těchto souborů se odebere Mitochondriální chromozom (chrM). ChrM má specifickou analýzu z důvodů jiné struktury, než jakou mají chromozomy v jádře viz kapitola 2.1.3. Kvůli tomu se v gvcf souborech na místě chrM objevuje netypický počet alel a analýza chrM by obsahovala výpočet jiných statistických údajů než na chromozomech v jádře. Z toho důvodu, a po konzultaci s vedoucím práce, se chrM nebude v rámci této práce zpracovávat.

Soubory s odebraným chrM se znovu komprimují a indexují. Bez indexace souborů není možné nahrát soubory do databáze.

4.1.2 Načtení gvcf souborů do databáze

Při ukládání gvcf souborů do databáze se soubory sjednotí přes lokusy a alely. Vytvoří se dvě tabulky. Jedna tabulka referencí a druhá tabulka variant.

Tabulka referencí obsahuje lokusy, které se ve všech souborech shodovaly s referencí. Ale není nutné v ní zachovávat veškeré genotypové informace z gvcf souborů.

Tabulka variant oproti tomu popisuje lokusy s alespoň jednou nalezenou variantou v alespoň jednom souboru. V této tabulce se záznamy GT, AD a PL uloží v závislosti na nově vytvořeným lokálním indexu alely (LA). Záznamy a struktura vstupních gvcf souborů je popsaná v kapitole 2.8.

4.1.3 Rozdělení řádků obsahující několik alel

V každém souboru se mohou objevit na diploidních místech až dvě varianty. Na haploidních místech bude v souboru gvcf vždy pouze jedna variant.

Při sjednocení gvcf souborů do databáze přes lokusy a alely vznikají v databázi řádky obsahující i několik variant.

Výsledkem tohoto kroku je upravení řádků a záznamů tak, aby jeden řádek obsahoval informace pouze o jedné variantě vůči referenci. Zaniká lokální index LA. Hodnoty GT, AD a PL se ukládají způsobem, aby odpovídaly referenci a variantě na rozdělených řádcích.

Rozdělení řádků zjednoduší následnou filtraci variant a agregaci dat. Tento krok je ale především nepostradatelným pro následnou anotaci anotačními nástroji.

Rozdělení variant může způsobit nekonzistenci některých údajů. Kapitola 4.1.6 popisuje přepočítání nevalidních hodnot potřebných pro další kroky analýzy.

4.1.4 Anotace s využitím anotačních nástrojů

Anotační nástroje přidávají údaje sloužící pro interpretaci varianty viz kapitola 2.5. Anotace přidá velké množství informací. Nejdůležitější pro nás jsou frekvence alel z jiných populačních databází a funkcionální informace o variantě.

4.1.5 Anotace fenotypem

Co to je fenotyp a jeho význam je popsaný v kapitole 2.2.4. Zkráceně fenotypová data v populačních databázích popisují viditelné vlastnosti a znaky člověka spolu s informacemi o národnosti a vlivy prostředí.

Anotace fenotypem v naší databázi bude obsahovat např. následující data: pohlaví, původ otce a matky a typ sekvenování (genomové či exomové).

4.1.6 Validace dat

Přepočítáváme alelickou frakci a nikoli další hodnoty, které nejsou validní po rozdělení řádků. Důvodem je, že ji využíváme v dalších částech analýzy (filtrace).

Alelická frakce nebo taky označovaná jako alelická bilance (allelic balance) je proporce readů podporujících variantu. Tuto hodnotu nalezneme v gvcf souborech, ale po rozdělení variant je nutné ji přepočítat z hodnoty uložené v AD (počet nefiltrovaných readů).

Hodnota AF (alelická frakce) je přepočítána následujícím vzorcem 4.1

$$AF = [\frac{AD\ reference}{AD\ reference + AD\ varianty}, \frac{AD\ varianty}{AD\ reference + AD\ varianty}] \qquad (4.1)$$

V tomto kroku se přepočítává hlavně hodnota AF, jelikož je důležitá pro následnou filtraci.

Je důležité neplést stejné označení alelické frakce a alelické frekvence. Oba termíny jsou zkráceně označované jako AF, ale jejich význam je odlišný.

4.1.7 Filtrace variant

Filtrace je důležitým krokem při vytváření populační databáze. Hlavním cílem je odstranit varianty, které by mohly být falešně pozitivní.

Filtrovaní se zbavuje variant s nízkou kvalitou a variant podpořených malým počtem čtení. Tyto údaje jsou uloženy v záznamech GQ, DP a AF (alelická frakce přepočítaná v předchozím kroku řetězce zpracování).

4.1.8 Agregace dat

Důležitým prvkem populačních databází jsou agregované údaje skrz celou populaci (všechny vzorky). Tyto údaje mohou vypovídat například o tom, zda je varianta běžná ve studované populaci nebo která báze měla jak velké zastoupení čtení.

Hlavními vypočítávanými parametry jsou hodnoty AC (počet alel pro každou alternativní alelu ve volaném genotypu), AN (celkový počet alel ve volaném genotypu), AF (frekvence alely), střední hodnota pokrytí báze, počet homozygotů.

AC, AN a počet homozygotů jsme schopný získat z GT. Hodnoty vypočítané pro jeden vzorek budeme nazývat jako lokální. Lokální hodnoty se dále budou agregovat součtem skrz všechny vzorky v databázi. AF se vypočítá z podílu agregovaných AC a AN viz vzoreček 4.2

$$AF = \frac{AC}{AN} \qquad (4.2)$$

Střední hodnota pokrytí báze je vypočítána z průměrné hodnoty DP skrz všechny vzorky.

4.1.9 Agregace dat vůči fenotypu

Agregace dat vůči fenotypu využívá hodnot vypočítaných v předchozím kroku. Vypočítané lokální hodnoty AC, AN a počet homozygotů se agreguje vždy skrz určitý výběr vzorků. Zajímá nás hlavně agregace podle pohlaví a podle typu sekvenování (celogenomové či celoexomové sekvenování).

4.1.10 Uložení finálního datasetu

Po provedení všech kroků se výsledná databáze uloží.

4.2 Výběr nástrojů a databáze

V této kapitole jsou rozepsané vybrané nástroje pro vypracování předchozího navrženého řetězce zpracování. Byl vybrán Hail pro tvorbu a práci s databází, VEP jako anotační nástroj a dále se využilo nástrojů Vcftools, Gunzip, Tabix a Python knihovny Pandas. Virtuální prostředí bylo zajištěno nástrojem Docker.

4.2.1 Tvorba prostředí – nástroj Docker

Docker je virtualizační nástroj. Nevyužívá však virtuální stroje, ale pracuje s kontejnery. Ty mají sdílené jádro operačního systému a spouštějí pouze aplikaci a její závislosti. Narozdíl od virtuálních strojů nereprodukují celý virtuální hardware počítače a celý operační systém. Proto jsou kontejnery menší a rychleji se nasazují.

Kontejner v Dockeru je tvořený podle Docker image (obrazu). Image je přenositelný a podle jeho předlohy může vzniknou několik stejných kontejnerů. Tyto kontejnery se vzájemně neovlivňují – jedná se o oddělené prostředí. Docker image se mohou stahovat z webu https://hub.docker.com/ nebo sestavit z Docker file.

Docker file jsou soubory, které slouží jako předpis sestavení image a popisují jaké součásti Docker image má obsahovat. Příklady informací v Docker file jsou například verze kernelu operačního systému, baličky k instalaci a další.

4.2.2 Nástroje pro tvorbu a práci s databázi

Hail byl vybrán jako nástroj pro nahrání gvcf souborů do jedné databáze a následnou práci s ní. Ostatní nástroje pro zpracování genetických souborů vcf a gvcf a jejich výhody a nevýhody jsou v kapitole 2.9.

Hail je python knihovna pro analýzu strukturovaných dat, zejména genetických dat. Využívá technologií Hadoop a Apach Spark popsaných v kapitolách 2.7.1 a 2.7.2.

Hlavní důvody volby Hailu:

1. Jedná o aktualizovanou a stále se rozšiřující knihovnu.
2. Je vhodný pro zpracování velký dat a jejich uložení do připravených datových struktur (Table, MatrixTable, VariantDataset).
3. Obsahuje metody pro tvorbu populačních databází a pro přímé spuštění anotacích nástrojů.

Hail umožňuje ukládání dat do dvou základních datových struktur Table a MatrixTable.

Table je ekvivalentem pro SQL tabulku nebo Pandas dataframe. Má 2D strukturu a skládá se z řádků a sloupců. Slouží především k načítání dat z tsv nebo csv souborů obsahující často fenotypové údaje. Datová struktura Table může využívat agregační funkce, sjednocovat několik tabulek dohromady a později sloužit jako zdroj anotace pro MatrixTable.

MatrixTable je rozšíření datové struktury Table o 1 dimenzi a jedná se o 3D strukturu. Mimo řádky a sloupce obsahuje také Entry fields. Entry fields je 2D matice se záznamy indexovaných pomocí klíče řádku a sloupce. Další částí MatrixTable jsou Global fields (globální údaje konstantní pro každý záznam). Import funkce načítá do MatrixTable formáty obsahující genetické varianty, nejčastěji uložené ve vcf souborech.

MatrixTable umožňuje načítat soubory variant, sjednocovat soubory, agregovat a filtrovat data, anotovat varianty a ukládat výsledky.

Pro uložení sjednocených gvcf souborů se v Hail využívá datové struktury VariantDataset. Tato datová struktura je vhodná pro uložení populační databáze a vznikla jako odezva na gnomAD projekt popsaný v kapitole 2.10. VariantDataset se skládá ze dvou částí, reference_data a variant_data. Obě tyto části jsou Matrix tabulky a dá se na ně použít všech metod této struktury. VariantDataset je součástí modulu vds, který nabízí navíc metody, které Matrix tabulka nenabízí.

Hail se rozděluje do několika modulů. V této práci se využívaly zejména moduly vds, aggregators, experessions.

Vds modul obsahuje metody a funkce pro tvorbu datové struktury VariantDataset a práci s ním.

Modul aggregators zjednodušuje práci s daty uloženými v datových strukturách. Umožňuje například agregaci dat (součet, počet dat), vypočítání důležitých statistických charakteristik jako je střední hodnota či korelace, ale také zobrazení dat v histogramu či vyhodnocení bool hodnot.

Hail nevyužívá běžných datových typů a ani jejich metod. Místo toho data ukládá do expressions (výrazů), které mají přidružený hailovský datový typ. Příkladem jsou Int64Expression – typu tint64, ArrayExpression – typu tarray, CallExpression – typu tcall a další. Modul expressions zajišťuje tvorbu a práci s výrazy.

Výše popsané možnosti Hailu neuvádí všechny jeho možnosti. Knihovna Hail nabízí více modulů a metod pro práci s genetickými daty.

4.2.3 Nástroje pro anotaci

Pro anotaci dat se zvolil anotační nástroj VEP. Jeho spuštění je snadno implementovatelné v nástroji Hail, což byla hlavní premisa volby tohoto nástroje. Zároveň anotuje veškeré pro nás důležité informace od frekvencí z jiných populačních databází až po transkriptové důsledky.

Další výhody VEPu oproti ostatním anotačním nástrojům jsou popsány v kapitole 2.5.1.

Prostředí vytvořené pomocí Docker nástroje je zároveň připravené se všemi závislostmi pro implementaci anotačního nástroje Nirvana.

4.2.4 Další použité nástroje

Dalšími využitými nástroji jsou:

1. Vcftools pro odebrání chrM
2. Gunzip pro kompresy souborů.

3. Tabix od Samtools pro indexaci souborů.
4. Python knihovna Pandas pro čtení cvs souborů obsahující fenotypová data.
5. Z důvodů přenositelnosti prostředí a zajištění kompatibility jednotlivých verzí nástrojů se využívá virtualizace pomocí nástroje Docker.

4.2.5 Návrh databáze

Data z gvcf souborů se ukládají do datové struktury VariantDataset. VariantDataset je datová struktura nástroje Hail popsaného výše v kapitole 4.2.2.

VariantDataset obsahuje dvě MatrixTable pojmenované variant_data a reference_data. Tabulka variant_data obsahuje záznamy, kde je zaznamenaná varianta na alespoň jednom vzorku. Tabulka reference_data naopak obsahuje záznamy kde se všechny vzorky shodovaly s referencí.

Obě tabulky jsou datové struktury MatrixTable se rozdělují na 4 základní části: Global fields, Column fields, Row fields, Entry fields. Řádky jsou konstantní pro všechny sloupce a sloupce jsou konstantní pro všechny řádky MatrixTable. Globální údaje jsou konstantní pro všechny záznamy a shrnují veškerá data v datasetu. Entry fields je 2D matice se záznamy indexovaných pomocí klíče řádku a sloupce.

Unikátní klíč sloupců je *s* což je string označující název původního gvcf souboru.

Unikátní klíč řádků se skládá ze dvou klíčů: *locus* a *alleles*.

Locus klíč je hailovského datového typu locus<GRCh38> a obsahuje informaci o pozici. Pozice je složena z názvu chromozomu – locus.coting a z přesné pozice – locus.position.

Alleses klíč je typu array<str> (pole stringů) a obsahuje všechny alely nalezené v gvcf souborech na daném lokusu včetně alely původní reference.

Ve výsledku mají datasety na každém řádku pouze původní referenci a jen jednu variantu. Výsledky anotace VEP jsou uložené v Row fields a stejně tak finální agregované údaje. Anotace fenotypem je uložena v Global fields jako datová struktura.

4.2.6 Softwarové požadavky

Tato kapitola popisuje softwarové požadavky pro správné fungování vybraných nástrojů zmíněných výše (Hail, Apache Spark, VEP)

Obecné softwarové požadavky:

1. operační systém Ubuntu (verze 20.04)
2. Python 3.9
3. nástroje unzip, tar a gzip, git, wget, tabix
4. balíčky cpanminus, libmysqlclient-dev, libbz2-dev, liblzma-dev, tzdata

Python knihovny:

1. Numpy
2. Hail
3. Pandas
4. Docopt

Softwarové požadavky pro Hail:

1. Openjdk-8-jre-headless (je požadovaná Java 8 nebo 11 JDK)
2. PySpark (instalace probíhá v rámci instalace Hailu)

Softwarové požadavky pro VEP (instalace z git repositáře):

1. balíčky libpng-dev, zlib1g-dev, libbz2-dev, liblzma-dev
2. Perl (perl-base, perl, bioperl)
3. curl

Softwarové požadavky pro Nirvanu (instalace z git repositáře):

1. Dotnet-sdk-5.0
2. Dotnet-sdk-6.0
3. Packages-microsoft-prod.deb
4. Apt-transport-https

Všechny výše zmíněné požadavky jsou sepsané v docker file. Docker image je připravený pro práci s Nirvanou, ale Nirvana nebyla nakonec implementována.

4.2.7 Hardwarové požadavky

V celém procesu se využívalo Apache Spark clusterů. Architektura frameworku Spark je popsaná v kapitole 2.7.2.

Hardwarové požadavky jsou především na nastavení operační paměti RAM driveru a executorovi ve Spark architektuře.

Pro driver je nastavení 4GB a pro každý executor je nastaveno 8GB operační paměti.

5 Implementace

V této kapitole je popsaná tvorba prostředí, jednotlivé skripty a konfigurační soubory. Zároveň je zde ukázáno, jak se skripty pracovat a jak s jejich pomocí vytvořit vlastní populační databázi. Skripty součástí přílohy, ale také jsou nahrány v git repozitáři, který je napsaný v Docker file.

5.1 Tvorba prostředí

Implementace proběhla ve virtualizovaném Docker kontejneru vytvořeném z Docker image. Ten byl sestaven podle Docker file, který je součástí přílohy této práce.

5.1.1 Docker file

V příloze této práce je Docker file s názvem Dockerfile_hail. Je založený na Ubuntu operačním systému verze 20.04. Docker file nastavuje správně čas a časovou zónu, vytváří základní adresářovou strukturu, instaluje požadovaný software (popsaný v kapitole 4.2.6) a stahuje námi vytvořené skripty z našeho git repositáře. Tyto skripty jsou zároveň součástí přílohy, ale doporučení je využívat skriptů stažených z git repositáře.

5.1.2 Vytvoření Docker image

Pro vytvoření Docker image se využívá příkazového řádku. Jako první je nutné se přepnout do adresáře obsahujícího Docker file viz příkaz v bloku Kód 5.1.

```
cd cesta_k_docker_filu
```

Kód 5.1 Změna adresáře.

Po přepnutí do správného adresáře lze spustit příkaz, viz Kód 5.2. Ten sestaví Docker image s označením dnai/hail a s tagem hail:agg1.0. Parametr -f je pro název Docker file podle kterého se image sestaví.

```
docker build -t dnai/hail:agg1.0 -f Dockerfile_hail .
```

Kód 5.2 Sestavení Docker image.

5.1.3 Spuštění Docker kontejneru a připojením složky s daty

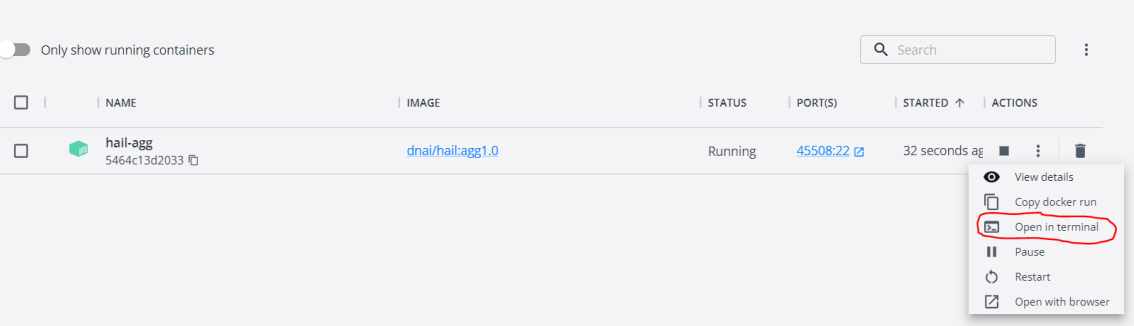
Pro tvorbu Docker kontejneru spolu s připojením (mount) složky hostujícího zařízení se složkou uvnitř Docker kontejneru slouží příkaz viz Kód 5.3.

```
docker run -d -t -v cesta_k_adresáři_dat:/opt/data --name hail-agg
-p 45508:22 dnai/hail:agg1.0
```

Kód 5.3 Spuštění Docker kontejneru s připojením složky s daty.

5.1.4 Práce v Docker kontejneru a spuštění skriptů

V aplikaci Docker Desktop se dá jednoduše připojit do Docker kontejneru kliknutím na tlačítko Open in terminal, jak je vidět na Obrázek 5.1.



Obrázek 5.1 Připojení do Docker kontejneru skrz grafické uživatelské rozhraní.

Dalším způsobem je využít příkazové řádky hostujícího zařízení a připojit se pomocí příkazu, viz Kód 5.4.

```
docker container attach hail-agg
```

Kód 5.4 Připojení do Docker kontejneru skrz příkazový řádek.

Docker má v sobě implementované python venv environment. V něm jsou všechny potřebné knihovny pro správné spuštění python skriptů, které jsou součástí této práce.

Před spouštěním skriptů je proto potřeba se přepnout do venv prostředí pomocí následujícího příkazu, viz Kód 5.5.

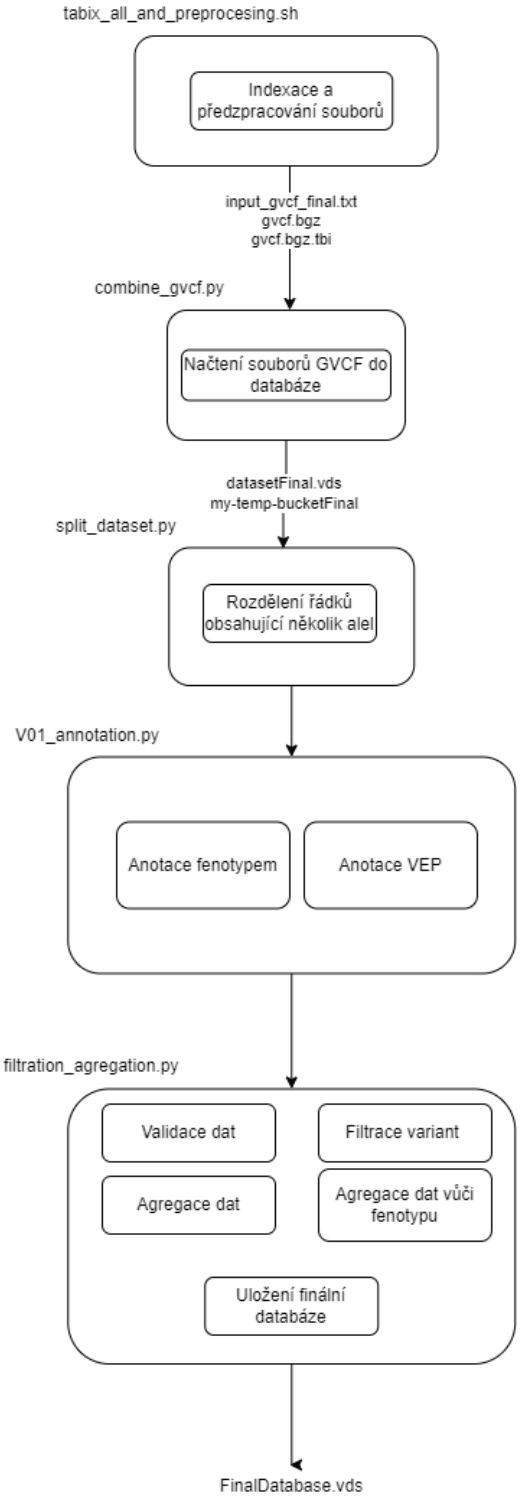
```
. /opt/gvcf_database/.venv/bin/activate
```

Kód 5.5 Příkaz pro přepnutí do venv prostředí.

5.2 Implementace navrhnutého řetězce zpracování

V našem prostředí se zpracovávalo celkem 10 gcvf souborů obsahujících celý genom sekvenovaného subjektu. Z důvodu ochrany osobních údajů však nemůže být součástí přílohy soubor s celým genomem. Proto se vytvořily testovací gvcf soubory obsahující jen několik náhodně vybraných záznamů. Tyto soubory jsou přiložené v příloze. V příloze jsou zároveň veškeré skripty, konfigurační soubory, Docker file, pomocné soubory (například soubor input_gvcf_final.txt obsahující cesty ke gVCF souborům) a výsledná databáze vytvořena z testovacích dat.

Celý řetězec zpracování je implementovaný pomocí pěti skriptů: tabix_all_and_preprocesing.sh, combine_gvcf.py, split_dataset.py, V01_annotation.py a filtration_agregation.py. Na Obrázek 5.2 jsou vypsané skripty a o jaké kroky se starají z navrženého řetězce zpracování viz kapitola 4.1.



Obrázek 5.2 Zobrazené kroky řetězce zpracování v jednotlivých skriptech, jejich návaznost na sebe, vstupy a výstupy.

5.2.1 Indexace a předzpracování souborů

Prvním krokem řetězce zpracování je odstranění Mitochondriální DNA (chrM) a indexace souborů. Slouží pro to skript *tabix_all_and_preprocesing.sh*.

Ten si na vstupu bere cestu k adresáři, kde jsou uložená data (gVCF soubory). Ukázka spuštění skriptu *tabix_all_and_preprocesing.sh* je vidět v Kód 5.6.

```
tabix_all_and_preprocesing.sh cesta_k_souborům_gvcf
```

Kód 5.6 Ukázka spuštění skriptu *tabix_all_and_preprocesing.sh*.

Skript *tabix_all_and_preprocesing.sh* zpracovává každý souboru ve vstupním adresáři s koncovkou gvcf.gz. Všem odebere chrM pomocí nástroje *vcftools*. Nástroj umožňuje filtraci variant podle pozice (v našem případě se jedná o odebrání celého chrM). Výstup z vcftools se dále komprimuje pomocí nástroje gzip.

To vytvoří data blokově komprimovaná. Pro správnou funkci Hailu je však potřeba soubor znovu uložit s pomocí bgzip do souboru s koncovkou .bgz.

Takto upravený soubor se indexuje pomocí nástroje tabix, čímž vzniká nový soubor s přidanou koncovkou .tbi.

Použití vcftools, komprimace a indexace souborů je vidět v Kód 5.7.

```
vcftools --gzvcf "$file" --not-chr chrM --recode --recode-INFO-all
--stdout | gzip -c > "$1"/GVCF_"$ID_NUMBER"_chrM_out.gvcf.gz

gunzip -c "$1"/GVCF_"$ID_NUMBER"_chrM_out.gvcf.gz | bgzip >
"$1"/GVCF_"$ID_NUMBER"_chrM_out.gvcf.bgz

tabix -f -p vcf  "$1"/GVCF_"$ID_NUMBER"_chrM_out.gvcf.bgz
```

Kód 5.7 Příkaz pro odebrání chrM pomocí vcftools a následná komprimace.

Skript zároveň vytvoří ve složce soubor *input_gvcf_final.txt*. Ten v sobě má cesty ke gvcf souborům a slouží jako vstup do dalšího kroku (načtení předzpracovaných gvcf souborů do databáze).

Testovací gvcf soubory obsažené v rámci přílohy byly generovány podle velmi podobného skriptu *testing_dataset.sh*, který je také součástí přílohy. Rozdíl ve skriptech je, že se neodebírá chrM, ale zachovává se 36 záznamů podle souboru *positions_to_testing_gvcf.txt*. Tento soubor obsahuje na každém řádku contig (název chromozomu) a tabulátorem oddělenou pozici na chromozomu.

Zároveň jsou pomocí skriptu *testing_dataset.sh* odebrány z hlavičky souboru nepotřebné contig informace. Skript *testing_dataset.sh* zajišťuje i indexaci souborů, a proto není potřeba testovací data před načítáním do databáze dále upravovat.

5.2.2 Načtení gvcf souborů do databáze

Načtení souborů do databáze je implementované ve *skriptu combine_gvcf.py*. Skript si na vstupu bere parametr s označením -i. Tento parametr slouží pro načtení souboru s cestami ke gvcf souborům. Jedná se o již zmíněný soubor *input_gvcf_final.txt*, který vzniká použitím skriptu *tabix_all_and_preprocesing.sh* viz předchozí kapitola 5.2.1. Pokud chceme získat soubor *input_gvcf_final.txt* pro testovací data, které není potřeba předzpracovávat pomocí skriptu *tabix_all_and_preprocesing.sh,* tak součástí přílohy je také skript *get_all_input_files.sh*. Ten slouží pouze k načtení cest všech souborů s koncovou *gvcf.bgz* v daném adresáři.

Spuštění skriptu *combine_gvcf.py* je ukázáno v Kód 5.8.

```
python3 /opt/gvcf_database/scripts/combine_gvcf.py -i
/opt/data/input_gvcf_final.txt
```

Kód 5.8 Spuštění skriptu combine_gvcf.py s parametrem -i (souboru s cestami gvcf).

V první části skript *combine_gvcf.py* načte do pole cesty k souborům gVCF ze souboru daném na vstupu v argumentu -i.

Další část využívá vds modulu Hailu pro načtení souborů do datové struktury VariantDataset sloužící jako databáze. Specificky se využívá metody *hl.vds.new_combiner()* sloužící pro definování kombineru. V naší implementaci se využilo v rámci metody *hl.vds.new_combiner()* těchto parametrů: *output_path* (cesta uložení výsledného VariantDatasetu), *temp_path* (cesta uložení dočasných souborů vzniklých při tvorbě VariantDatasetu), *gvcf_paths* (cesty k gvcf souborům), *use_genome_default_intervals=True*, *reference_genome =‘GRCh38‘* (použitá reference pro tvorbu gvcf souborů).

Definovaný kombiner se následně spouští pomocí metody *.run()*. Kombiner gvcf soubory sjednotí přes lokusy a alely a vytvoří lokální indexy alel (LA). Tento index slouží pro indexaci hodnot GT, AD a PL. Ty jsou ve VariantDatasetu následně uloženy jako LGT, LAD a LPL a jsou přístupné právě přes hodnotu LA. Na závěr kombiner ukládá VariantDataset do adresáře zaznamenaném v parametru *output_path*.

5.2.3 Rozdělení řádků s více variantami (multialelické záznamy)

Pro tento krok slouží skript *split_dataset.py* spuštěný viz kód 5.9. Ten načítá VariantDataset vytvořený v minulém kroku. Pro načtení VariantDatasetu se využívá funkce *hl.vds.read_vds()* viz Kód 5.10.

```
python3 /opt/gvcf_database/scripts/split_dataset.py
```

kód 5.9 Spuštění skriptu split_dataset.py

```
vds = hl.vds.read_vds('/opt/data/vds_datasets/datasetFinal.vds')
```

Kód 5.10 Funkce pro načtení VariantDatasetu.

Před spuštěním funkce pro rozdělení řádků se zahazují LPL záznamy pomocí metody *drop()* viz Kód 5.11

```
vds.variant_data = vds.variant_data.drop('LPL')
```

Kód 5.11 Příkaz pro zahození LPL záznamu.

Důvodem je, že Hail má v této chvíli problém s načítáním PL hodnot pomocí *hl.vds.new_combiner()* použitého během předchozího kroku. Na většině řádků ve VariantDatasetu se LPL hodnota načte správně. Ale jsou lokusy (hlavně haploidní varianty), na kterých Hail není schopný PL načíst a vznikají ve VariantDatasetu prázdná LPL pole. Ty poté dělají problém při rozdělování řádků s multialelickými záznamy.

Pro rozdělování řádků se využívá Hail funkce *hl.vds.split_multi(vds).* Tato funkce rozdělí všechny řádky, které mají více jak jednu variantu. Zároveň se upravují záznamy následujícím způsobem: Pole LA zaniká a z polí LGT, LAD vznikají GT a AD odpovídající rozděleným řádkům. DP je beze změny. GQ se má přepočítávat z PL hodnoty (vzniklé z LPL pole). Pokud PL neexistuje jako v našem případě, tak GQ není přepočítáno a je zachována jeho původní hodnota. Pole RGQ (kvalita genotypu reference) a END jsou zachovány beze změny.

Výstupem skriptu je nový VariantDataset datasetSplitFinal.vds s rozdělenými řádky.

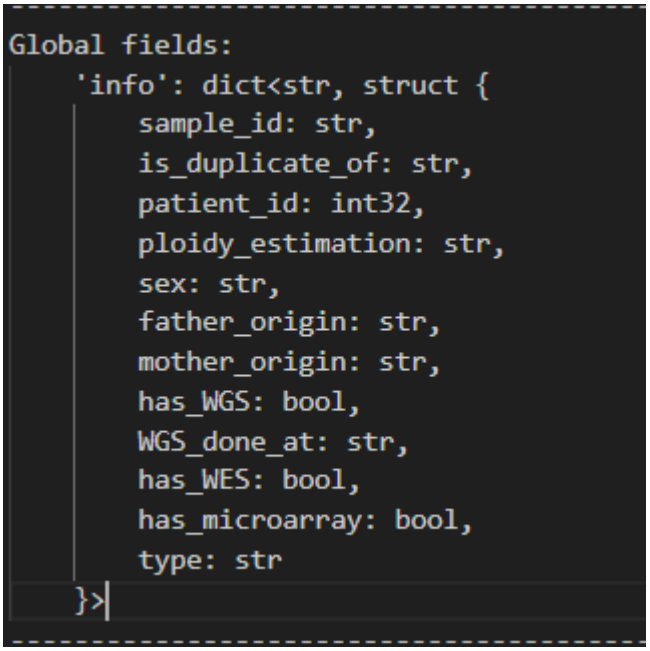
5.2.4 Anotace

Pro anotaci slouží skript *V01_annotation.py* spuštěný viz Kód 5.12. Skript používá VariantDataset datasetSplitFinal.vds z minulého kroku.

```
python3 /opt/gvcf_database/scripts/V01_annotation.py
```

Kód 5.12 Spuštění skriptu V01_annotation.py

První částí anotace je anotace fenotypovými daty. Fenotyp je uložen v souboru *donor_data.csv* viz příloha A. Fenotypová data se k VariantDatasetu nahrávájí podle klíče sloupců ‚s' (obsahuje název původního souboru) a přidávají následující globální údaje viz Obrázek 5.3



Obrázek 5.3 Fenotypová data přidaná do databáze.

Druhou částí je anotace nástrojem VEP. Pro anotaci VEPem v Hailu slouží metoda *hl.methods.vep()* viz Kód 5.13.

```
mt = hl.methods.vep(mt, config, block_size=50000)
```

Kód 5.13 Iniciace VEPu v Hailu.

Tato metoda používá MatrixTable. V našem případě se do ní uloží vds.variant_data a dále anotuje VEPem podle nastavení v konfiguračním souboru.

Konfigurační soubor je v příloze pod názvem *vep-config.json*. Obsahuje příkaz spouštějící VEP a výsledné schéma ve formě json struktury.

Součástí VEP konfiguračního souboru jsou cesty k referenčnímu genomu. Soubory s referencí nemohou být součástí přílohy ani součástí git repositáře, jelikož jejich velikost dosahuje desítek GB. Proto je nutné pro fungování VEP stáhnout následující soubory zvlášť a poté je nahrát do adresáře napsaném v konfiguračním souboru:

1. fasta soubory reference hg38.fa
2. Ensembl cache soubory: homo_sapiens_vep_107_GRCh38.tar.gz [46]
3. Data dbNSFP databáze: dbNSFP4.3c_grch38.gz [47]

VEP anotuje varianty v blocích a blok_size parametr v Kód 5.13 určuje počet variant v jednom bloku zpracování.

5.2.5 Validace, Filtrace, Agregace dat a uložení finálního datasetu

Zbylé kroky z navrhnutého řetězce zpracování (validace, filtrace, agregace a uložení finálního datasetu) jsou implementované ve skriptu *filtration_agregation.py* spuštěného viz. Kód 5.14.

```
mt = hl.methods.vep(mt, config, block_size=50000)
```

Kód 5.14

Krok validace zahrnuje přepočítání alelické frakce (AF záznam). Nejprve se zahodí původní AF a poté je vypočítána z hodnot v AD viz Kód 5.15

```
vds.variant_data = vds.variant_data.drop('AF')

vds.variant_data = vds.variant_data.annotate_entries(AF =
[vds.variant_data.AD[0]/(vds.variant_data.AD[0]+vds.variant_data.AD
[1]),
vds.variant_data.AD[1]/(vds.variant_data.AD[0]+vds.variant_data.AD[
1])] )
```

Kód 5.15 Validace AF hodnoty

Poté filtrují záznamy právě podle hodnot AF a DP. V návrhu byla zmíněná filtrace i podle hodnot GQ. Ty ale nebyli přepočítány díky odebrání PL hodnot a proto se v tuto chvíli podle GQ záznamy nefiltrují.

```
vds.variant_data =
vds.variant_data.filter_entries((vds.variant_data.DP >
10)&(vds.variant_data.AF[1] > 0.2))
```

kód 5.16 Filtrace podle DP a AF

Implementace filtrace v rámci skriptu je vidět v kód 5.16. Jsou filtrovány záznamy s DP hodnotou menší jak 10 a s AF hodnotou menší jak 0,2.

Po filtraci se provádí krok agregace s využitím to metody *hl.variant_qc()*. Tato metoda vypočítává hodnot DP, AC, AN, AF, počet homozygotů skrz všechny vzorky a anotuje je v rámci Rows fields.

Mimo funkci hl.variant_qc() se ve skriptu *filtration_agregation.py* vypočítávají lokální AC a lokální AN (zvlášť pro každý vzorek). Důvodem je jejich následné použití při agregace vůči fenotypu.

Agregace podle fenotypu se provádí dočasnou filtrací podle fenotypu. Vzorky se filtrují podle pohlaví a podle typu sekvenování (celogenomové či celoexomové sekvenování). Na vyfiltrovaných datech se provede agregace lokálních AC, lokálních AN a počtu homozygotů, které se anotují v rámci Rows fields.

5.3 Licence a možnosti použití kódu

Celá práce byla vyvíjená na licenci Apache License 2.0. Součástí přílohy je soubor LICENSE-2.0.txt definující práva vztahující se na Apache License 2.0. [48]

6 Výsledky

Navržený řetězec zpracování z kapitoly 4.1 byl implementovaný pomocí vybraných nástrojů v kapitole 4.2. S využitím vytvořených skriptů byla v Docker kontejneru vytvořena populační databáze na vzorku deseti gvcf souborů. Výsledná databáze není součástí přílohy z důvodu ochrany osobních údajů a z důvodu její velikosti. V příloze je však ukázková databáze s názvem *FinalDatabase.vds* a její struktura je však popsaná v rámci této kapitoly.

Výsledná databáze odpovídá návrhu v kapitole 4.2.5 Je uložena v datové struktuře VariantDataset python knihovny Hail. Obsahuje dvě MatrixTable, jedna s názvem reference_data a druhá s názvem variant_data.

Tabulka variant_data obsahuje záznamy, kde je zaznamenaná varianta na alespoň jednom vzorku. Tabulka reference_data naopak obsahuje záznamy, kde se všechny vzorky shodovaly s referencí. Data byla zpracovávaná pouze v tabulce variant_data. Důvodem je, že reference_data slouží ve chvíli přidaní nových gvcf souborů do databáze, aby umožnila agregaci dat skrz všechny vzorky. A to i na lokusech, kde v původní databázi nebyla variant, ale po přidání nových gvcf souboru je.

Částečná struktura výsledné tabulky variant_data je vidět v Kód 6.1 a v Kód 6.2. Kompletní struktura tabulky je součástí přílohy v souboru *struktura_výsledné_databáze.txt.*

Data v tabulce jsou uloženy ve čtyřech částech. Část Row fields je nejrozsáhlejší především díky tomu, že obsahuje anotaci vytvořenou pomocí nástroje VEP. Krom toho jsou její součástí agregovaná data vytvořená funkcí *hl.variant_qc()* a agregovaná data vůči fenotypu. Část Global fields obsahuje anotaci fenotypem a část Column fields obsahuje názvy původních gvcf soubrů, které fungují jako klíč tabulky.

Entry fields jsou záznamy z původních gvcf souborů spolu s lokálně agregovanými daty vypočítané pro jeden vzorek.

Row fields:

'locus': locus<GRCh38>

'alleles': array<str>

'rsid': str

'a_index': int32

'was_split': bool

'vep': struct {

pole s sfunkční dopady varianty

frequencies: dict<str, struct {*obsahuje frekvence alely v různých populacích}*

'vep_proc_id': struct {*id procesu vep anotace*}

'variant_qc': struct {*agregační hodnoty vytvořené funkcí hl.variant_qc()*. }

dp_stats: struct {mean, stdev, min, max},

gq_stats: struct {mean, stdev, min, max},

AC: array<int32>,

AF: array<float64>,

AN: int32,

homozygote_count: array<int32>, }

'genome_allele_stats': struct {*agregované hodnoty fenotypu pro vzorky s celým genomem*

all: struct { AC, AN, homozygote_count },

XX: struct { AC, AN, homozygote_count},

XY: struct { AC, AN, homozygote_count

}

'exome_allele_stats': struct { *agregované hodnoty fenotypu pro vzorky exomem*

all: struct { AC, AN, homozygote_count },

XX: struct { AC, AN, homozygote_count},

XY: struct { AC, AN, homozygote_count }, }

Kód 6.1 Row field variant_data tabulky

Global fields:

'info':

dict<str, struct {sample_id, is_duplicate_of, patient_id, ploidy_estimation, sex, father_origin, mother_origin, has_WGS, WGS_done_at,has_WES, has_microarray, type }

Column fields:

's': str

Kód 6.2 Global fields a Column fields variant_data tabulky

7 Diskuse

Pro implementaci návrhu řešení byla klíčová Python knihovna Hail. V průběhu vypracování práce jsem narazila na několik limitací této knihovny.

Na problém jsem narazila hned na začátku implementace řetězce zpracování. A to při načítání mitochondriálního chromozomu. Funkce sloužící pro načítání gvcf souborů do VariantDatasetu není optimalizovaná pro záznamy v mitochondriálním chromozomu. V rámci této práce jsme se s vedoucím rozhodli mitochondriální chromozom nezpracovávat. Pokud bychom ale rozhodnutí přehodnotili, bylo by nutné funkci pro načítání gvcf souborů přepsat.

Ta samá funkce má zároveň problém s načítáním PL hodnot, které slouží pro určení genotypu a vypočítává se z nich GQ (genotypová kvalita). Z důvodu absence PL hodnot během fáze přepočítávání GQ, se podle hodnot GQ nemohly ve finále filtrovat záznamy, tak jak bylo myšleno v návrhu. V tuto chvíli se filtrace provádí pouze na základě hodnot AD a AF (alelická frakce) a možným vylepšení práce je oprava načítání PL hodnot.

Problém s načítáním PL jsem více jak 2 měsíce řešila s vývojáři nástroje Hail skrze jejich diskusní fórum a neúspěšně (nedostala jsem ani jednu odpověď). Uznávám ale, že můj problém byl velmi specifický. Tvorba populační databáze a zpracování gvcf souborů není nejběžnějším využitím Hailu. Dle mého pozorování Hail team na jiné dotazy odpovídá velmi rychle, proto nehodnotím diskusní fórum Hail nástroje negativně.

I přes výše zmíněné limitace je Hail jedním z nejlepších nástrojů pro práci s vcf soubory a díky množství dostupných metod, rozsáhlosti dokumentace a aktivnímu rozvoji knihovny.

Výsledkem práce bylo úspěšné vytvoření populační databáze ze vzorku deseti gvcf souborů. Výsledná databáze má veškeré náležitosti důležité pro populační databázi.

Databáze obsahuje rozdělené varianty a má filtrovaná data. Dále obsahuje anotace fenotypem i anotačním nástrojem VEP (známé důsledky varianty, frekvence z jiných populačních databází a další informace). Také obsahuje vypočítaná agregovaná data (AC, AN, AF, počet homozygotů a střední hodnota pokrytí). Agregovaná data jsou vypočítána včetně závislosti na fenotypu.

Pokud bereme v potaz ukázková schémata databáze 1000 genome v Hail dokumentaci, tak je schéma naší výsledné databáze s nimi velmi podobné.

Hlavní rozdílem je rozdělení datasetů. V ukázkových schématech je databáze rozdělena na autozomy, chrX a chrY. Důvodem může být lepší přístupnost či možné

nuance v implementaci pro rozdílné chromozomy. V budoucnu by bylo dobré prozkoumat výhody rozdělení databáze to více datasetů a případně vylepšit náš návrh a implementaci.

8 Závěr

Cílem této práce bylo navrhnout řetězec zpracování pro vytvoření populační databáze. Vybrat metody a nástroje pro následnou implementaci. Dalším krokem bylo implementovat řešení od tvorby skriptů až po tvorbu prostředí v rámci Docker nástroje a spustit celý implementovaný řetězec zpracování na malém vzorku dat.

V rámci práce jsem se seznámila s tvorbou populačních databází a navrhla řetězec zpracování pro vlastní populační databázi. Spolu s tím jsem vybrala metody a nástroje sloužící pro následnou implementaci navrženého řetězce zpracování.

V rámci implementace se mi podařilo vytvořit skripty odpovídající krokům v návrhu a také jsem vytvořila prostředí pro jejich spuštění v Docker kontejneru.

Splněny byly veškeré cíle práce až po vytvoření populační databáze z deseti gvcf souborů. Jediným nedostatkem je filtrace dat. Z důvodu nepřepočítaných GQ hodnot se filtruje pouze na základě DP a AF.

Možné využití této práce je použití výsledného řešení pro vytvoření české populační databáze.

9 Bibliografie

1] Úvod do buněčné biologie. In: ALBERTS, Bruce, Dennis BRAY, Karen HOPKIN, Alexander JOHNSON, Julian LEWIS, Martin RAFF, Keith ROBERTS a Peter WALTER. *Essential cell biology*. 4th ed. New York: Garland Science, 2014, s. 1-18. ISBN 978-0-8153-4454-4.

2] PŘISTOUPILOVÁ, Anna. *Nové techniky sekvenace lidského genomu a jejich uplatnění ve studiu molekulární podstaty a diagnostiky dědičně podmíněných onemocnění*. Praha, 2008. Bakalářská práce. UNIVERZITA KARLOVA 1. lékařská fakulta. Vedoucí práce Ing. Stanislav Kmoch, CSc.

3] Schéma exprese genetické informace. In: *Bmedic online* [online]. Brno, Česká republika: Bmedic online s.r.o., 2016 [cit. 2023-05-17]. Dostupné z: https://bmedic-online.cz/lekce/transkripce/?v=928568b84963

4] Genetika. In: NUSSBAUM, Robert, Roderick MCINNES a Huntington WILLARD. *Thomson and Thomson Genetics in medicine*. 8th ed. Philadelphia, PA: Elsevier, 2016, s. 3-11. ISBN 978-1-4377-0696-3.

5] HOCEK, Michal, Jitka DAĎOVÁ a Petra MÉNOVÁ. Schéma struktury DNA dvoušroubovice a párování bázi. *Vesmír*. 2015, **94**(2), 1.

6] Replikace. In: *Studiumbiochemie* [online]. Praha, Česká republika: -, 2013 [cit. 2023-05-18]. Dostupné z: http://www.studiumbiochemie.cz/replikace.html

7] Mitochondriální genom: The mitochondrial genome: structure, transcription, translation and replication. *Biochimica et Biophysica Acta (BBA) - Bioenergetics*. 1999, (1410), 103-123. ISSN ISSN 0005-2728. Dostupné z: doi:doi.org/10.1016/S0005-2728(98)00161-3

8] ALBERTS, Bruce, Alexander JOHNSON, Julian LEWIS, Martin RAFF, Keith ROBERTS a Peter WALTER. *Molecular biology of the cell* [online]. 4th ed. New York: Garland Publishing, 2002 [cit. 2023-05-17]. ISBN ISBN-10: 0-8153-3218-1. Dostupné z: https://www.ncbi.nlm.nih.gov/books/NBK21054/

9] TEPLÁ, Milada. Sekundární struktura tRNA. In: *Studiumbiochemie* [online]. Praha, Česká republika: -, 2013 [cit. 2023-05-17]. Dostupné z: http://www.studiumbiochemie.cz/na.html

10] INTRON. In: *National Human Genome Research Institute* [online]. Rockville Pike Bethesda, USA: National Institutes of Health, 2014 [cit. 2023-05-17].

Dostupné z: https://www.genome.gov/genetics-glossary/Intron

11] Codon. In: *Labster Theory* [online]. -: Labster, - [cit. 2023-05-17]. Dostupné z: https://theory.labster.com/codon/

12] *The Cell*. 2nd ed. Boston, USA: Sinauer Associates, 2000. ISBN 0-87893-106-6.

13] EICHLER, Evan E. Genetic Variation, Comparative Genomics, and the Diagnosis of Disease. *New England Journal of Medicine*. 2019, **381**(1), 64-74. ISSN 0028-4793. Dostupné z: doi:10.1056/NEJMra1809315

14] Types of DNA variant: structural variation. In: *Garvan Institute of Medial Research* [online]. Darlinghurst, Australia: Garvan Institute, - [cit. 2023-05-17]. Dostupné z: https://www.garvan.org.au/research/kinghorn-centre-for-clinical-genomics/learn-about-genomics/dna-base/collection1/structural-variation

15] CAMPBELL, Molly. Missense, Nonsense and Frameshift Mutations: A Genetic Guide. In: *Technology networks* [online]. Sudbury, Velká Británie: Technology Networks Ltd., - [cit. 2023-05-17]. Dostupné z: https://www.technologynetworks.com/genomics/articles/missense-nonsense-and-frameshift-mutations-a-genetic-guide-329274

16] Genotype vs Phenotype: Examples and Definitions. In: *Technology networks* [online]. Sudbury, Velká Británie: Technology Networks Ltd., - [cit. 2023-05-17]. Dostupné z: https://www.technologynetworks.com/genomics/articles/genotype-vs-phenotype-examples-and-definitions-318446

17] GJUVSLAND, Arne B., Erik PLAHTE, Tormod ÅDNøY, Stig W. OMHOLT a Justin O. BOREVITZ. Allele Interaction – Single Locus Genetics Meets Regulatory Biology. *PLoS ONE*. 2010, **5**(2), -. ISSN 1932-6203. Dostupné z: doi:10.1371/journal.pone.0009379

18] HEATHER, James M. a Benjamin CHAIN. The sequence of sequencers: The history of sequencing DNA. *Genomics*. 2016, **107**(1), 1-8. ISSN 08887543. Dostupné z: doi:10.1016/j.ygeno.2015.11.003

19] Klasické metody sekvenování. In: *LAB Guide: Průvodce laboratoří* [online]. -: -, - [cit. 2023-05-17]. Dostupné z: https://labguide.cz/metody/sekvenovani-dna/klasicke-metody-sekvenovani/

20] SKRYPNYKOVA, Ievgeniia. *Nová generace sekvenování - její principy a využití v mikrobiologii*. Brno, 2015. Bakalářská práce. MASARYKOVA UNIVERZITA PŘÍRODOVĚDECKÁ FAKULTA ÚSTAV EXPERIMENTÁLNÍ BIOLOGIE. Vedoucí práce RNDr. Monika Dolejská, PhD.

CHEN, Fei, Mengxing DONG, Meng GE, Lingxiang ZHU, Lufeng REN, Guocheng LIU a Rong MU. The History and Advances of Reversible Terminators Used in New Generations of Sequencing Technology. *Genomics, Proteomics & Bioinformatics*. 2013, **11**(1), 34-40. ISSN 16720229. Dostupné z: doi:10.1016/j.gpb.2013.01.003 [21]

KOLÍSKO, Martin. Moderní metody sekvenování DNA. *Živa*. Academie věd Česká republika: Nakladatelství Academia, 2017, **2017**(3), 73-76. [22]

PEREIRA, Rute, Jorge OLIVEIRA a Mário SOUSA. Bioinformatics and Computational Tools for Next-Generation Sequencing Analysis in Clinical Genetics. *Journal of Clinical Medicine*. 2020, **9**(1), -. ISSN 2077-0383. Dostupné z: doi:10.3390/jcm9010132 [23]

IDTechEx Assesses the Potential of Third Generation DNA Sequencing. In: *IDTechEx* [online]. -: -, - [cit. 2023-05-17]. Dostupné z: https://www.idtechex.com/en/research-article/idtechex-assesses-the-potential-of-third-generation-dna-sequencing/27629 [24]

BERBERICH, Amanda J., Rosettia HO a Robert A. HEGELE. Whole genome sequencing in the clinic: empowerment or too much information?. *Canadian Medical Association Journal*. 2018, **190**(5), 124-125. ISSN 0820-3946. Dostupné z: doi:10.1503/cmaj.180076 [25]

HWANG, Byungjin, Ji Hyun LEE a Duhee BANG. *Single-cell RNA sequencing technologies and bioinformatics pipelines*. 2018, **50**(8), 1-14. ISSN 1226-3613. Dostupné z: doi:10.1038/s12276-018-0071-8 [26]

ZHANG, Sheng, Bo WANG, Lin WAN a Lei M. LI. Estimating Phred scores of Illumina base calls by logistic regression and sparse modeling. *BMC Bioinformatics*. 2017, **18**(1), -. ISSN 1471-2105. Dostupné z: doi:10.1186/s12859-017-1743-4 [27]

MCLAREN, William, Laurent GIL, Sarah E. HUNT, Harpreet Singh RIAT, Graham R. S. RITCHIE, Anja THORMANN, Paul FLICEK a Fiona CUNNINGHAM. The Ensembl Variant Effect Predictor. *Genome Biology*. 2016, **17**(1), -. ISSN 1474-760X. Dostupné z: doi:10.1186/s13059-016-0974-4 [28]

Nirvana github [online]. -: Illumina, - [cit. 2023-05-18]. Dostupné z: https://illumina.github.io/NirvanaDocumentation/ [29]

ANNOVAR Documentation [online]. 2010: -, - [cit. 2023-05-18]. Dostupné z: https://annovar.openbioinformatics.org/en/latest/ [30]

Big Data among Big Data: Genome Data. In: *3billion* [online]. Teheran-ro,

31] Gangnam-gu, Seoul: 3billion, Inc., - [cit. 2023-05-17]. Dostupné z: https://3billion.io/blog/big-data-among-big-data-genome-data

32] Top Big Data Technologies You Must Know [2023]. In: *InterviewBit* [online]. -: InterviewBit, 2021 [cit. 2023-05-17]. Dostupné z: https://www.interviewbit.com/blog/big-data-technologies/

33] Apache Hadoop. In: *Apache Hadoop* [online]. -: The Apache Software Foundation, c2006-2023 [cit. 2023-05-17]. Dostupné z: https://hadoop.apache.org

34] Introduction to Apache Spark. In: *AWS amazon* [online]. -: Amazon Web Services, Inc., - [cit. 2023-05-17]. Dostupné z: https://aws.amazon.com/big-data/what-is-spark/

35] Cluster Mode Overview. In: *Apache Spark* [online]. -: The Apache Software Foundation, 2018 [cit. 2023-05-17]. Dostupné z: https://spark.apache.org/docs/latest/cluster-overview.html

36] VCF - Variant Call Format. In: *GATK* [online]. Cambridge, USA: Broad Institute, - [cit. 2023-05-17]. Dostupné z: https://gatk.broadinstitute.org/hc/en-us/articles/360035531692-VCF-Variant-Call-Format

37] PAILA, Umadevi, Brad A. CHAPMAN, Rory KIRCHNER, Aaron R. QUINLAN a Paul P. GARDNER. GEMINI: a flexible framework for exploring genome variation: Integrative Exploration of Genetic Variation and Genome Annotations. *PLoS Computational Biology*. 2013, **9**(7), . ISSN 1553-7358. Dostupné z: doi:10.1371/journal.pcbi.1003153

38] What is a GVCF and how is it different from a 'regular' VCF?. In: *Broad Institute - GitHub* [online]. Cambridge, MA: Broad Institute of MIT and Harvard, * [cit. 2023-02-13]. Dostupné z: https://github.com/broadinstitute/gatk-docs/blob/master/gatk3-faqs/What_is_a_GVCF_and_how_is_it_different_from_a_%27regular%27_VCF%3F.md

39] *GATK* [online]. Cambridge, USA: Broad Institute, 2023 [cit. 2023-05-17]. Dostupné z: https://gatk.broadinstitute.org/hc/en-us

40] PAILA, Umadevi, Brad A. CHAPMAN, Rory KIRCHNER, Aaron R. QUINLAN a Paul P. GARDNER. GEMINI: Integrative Exploration of Genetic Variation and Genome Annotations. *PLoS Computational Biology*. 2013, **9**(7), -. ISSN 1553-7358. Dostupné z: doi:10.1371/journal.pcbi.1003153

41] *GLOW* [online]. -: Glow Authors, 2019 [cit. 2023-05-17]. Dostupné z: https://glow.readthedocs.io/en/latest/index.html

Hail [online]. -: Hail Team, c2015-2023 [cit. 2023-05-17]. Dostupné z: https://hail.is/docs/0.2/index.html [42]

GUDMUNDSSON, Sanna, Moriel SINGER-BERK, Nicholas A. WATTS et al. Variant interpretation using population databases: Lessons from gnomAD. *Human Mutation*. 2022, **43**(8), 1012-1030. ISSN 1059-7794. Dostupné z: doi:10.1002/humu.24309 [43]

GnomAD [online]. -: -, 2014 [cit. 2023-05-17]. Dostupné z: https://gnomad.broadinstitute.org [44]

HIGASA, Koichiro, Noriko MIYAKE, Jun YOSHIMURA et al. Human genetic variation database, a reference database of genetic variations in the Japanese population. *Journal of Human Genetics*. 2016, **61**(6), -. ISSN 1434-5161. Dostupné z: doi:10.1038/jhg.2016.12 [45]

Variant Effect Predictor - Annotation sources: Caches. In: *Ensembl Variant Effect Predictor* [online]. -: The Ensembl Variant Effect Predictor, 2016 [cit. 2023-05-18]. Dostupné z: https://www.ensembl.org/info/docs/tools/vep/script/vep_cache.html [46]

Dbnsfp - plugin data. In: *Ensembl* [online]. -: The Ensembl, 2016 [cit. 2023-05-18]. Dostupné z: https://www.ensembl.org/info/docs/tools/vep/script/vep_plugins.html#dbnsfp [47]

APACHE LICENSE, VERSION 2.0. In: *The Apache Software Foundation* [online]. -: The Apache Software Foundation, 2004 [cit. 2023-05-18]. Dostupné z: https://www.apache.org/licenses/LICENSE-2.0 [48]

Příloha A: Obsah přiloženého ZIP souboru

Zdrojové kódy, docker image pro tvorbu prostředí a testovací data jsou součástí přílohy A ve formě ZIP souboru

Obsah přílohy:

1. Složka configs
    a. vep_schema.json (popis VEP schema)
    b. vep_schema.py (popis VEP schema)
    c. vep_config.json (configurační soubor pro spuštění VEP)
2. Složka docker
    a. Dockerfile_hail
3. Složka scripts
    a. combine_gvcf.py
    b. filtration_agregation.py
    c. get_all_input_files.sh
    d. split_dataset.py
    e. tabix_all_and_preprocesing.sh
    f. V01_annotation.py
    g. testing_dataset.sh
4. Složka testing_data
    a. Složka anotace
        i. donor_data.csv (fenotyp k datům)
    b. Prázdná předpřipravená složka refs (adresář pro nahrání referencí)
    c. Složka vds_datasets
        i. FinalDatabase.vds výsledná databáze – vytvořená pomocí testovacích dat
    d. Testovací data (3 gvfc soubory – oříznuté na 36 řádků záznamů)
        i. GVCF_1_testing.gvcf.bgz
        ii. GVCF_1_testing.gvcf.bgz.tbi
        iii. GVCF_2_testing.gvcf.bgz
        iv. GVCF_2_testing.gvcf.bgz.tbi
        v. GVCF_3_testing.gvcf.bgz
        vi. GVCF_3_testing.gvcf.bgz.tbi
5. LICENSE-2.0.txt (Apache License 2.0)
6. positions_to_testing_gvcf.txt (soubor s pozicemi bází pro zachování v testovacích datech)
7. requirements.txt (soubor, který využívá Docker file pro stažení požadovaných python balíčků)
8. struktura_výsledné_databáze.txt