UNIVERZITA PALACKÉHO V OLOMOUCI

Přírodovědecká fakulta

Katedra biochemie

Zpracování informací z databázových a predikčních softwarových nástrojů pro anotaci necharakterizovaných proteinů pomocí bioinformatických metod

BAKALÁŘSKÁ PRÁCE

Autor:	**Alois Kozubík**
Studijní program:	B1406 Biochemie
Studijní obor:	Bioinformatika
Forma studia:	Prezenční
Vedoucí práce:	**Mgr. Zdeněk Perutka**
Rok:	2019

Prohlašuji, že jsem bakalářskou práci vypracoval/a samostatně s vyznačením všech použitých pramenů a spoluautorství. Souhlasím se zveřejněním bakalářské práce podle zákona č. 111/1998 Sb., o vysokých školách, ve znění pozdějších předpisů. Byl/a jsem seznámen/a s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, ve znění pozdějších předpisů.

V Olomouci dne 21. 5. 2019 .....

Poděkování:

Rád bych poděkoval vedoucímu mé bakalářské práce Mgr. Zdeňku Perutkovi za cenné podněty a zpětnou vazbu v průběhu celého projektu. Dále bych rád poděkoval Mgr. Ivo Chamrádovi, Ph.D. za užitečné rady, poskytnutí vstupních dat a pomoc při jejich interpretaci.

Bibliografická identifikace

Jméno a příjmení autora	Alois Kozubík
Název práce	Zpracování informací z databázových a predikčních softwarových nástrojů pro anotaci necharakterizovaných proteinů pomocí bioinformatických metod
Typ práce	Bakalářská
Pracoviště	Katedra biochemie
Vedoucí práce	Mgr. Zdeněk Perutka
Rok obhajoby práce	2019
Klíčová slova	Proteom, proteomika, BLAST, CELLO2GO, NucPred, Protein Cutter, WegoLoc, LOCALIZER, NSAF, Python, Biopython, ExPASy, UniProt, Swiss-Prot, Hordeum vulgare, hmotnostní spektrometrie
Počet stran	34
Počet příloh	1 CD
Jazyk	Český

Bibliographical identification

Autor's first name and surname:	Alois Kozubík
Title:	Compiling predicted data and database information for annotation of unknown proteins by using bioinformatic methods
Type of thesis:	Bachelor
Department:	Department of biochemistry
Supervisor:	Mgr. Zdeněk Perutka
The year of presentation:	2019
Keywords:	Proteome, proteomics, BLAST, CELLO2GO, NucPred, Protein Cutter, WegoLoc, LOCALIZER, NSAF, Python, Biopython, ExPASy, UniProt, Swiss-Prot, Hordeum vulgare, mass spectrometry
Number of pages:	34
Number of appendices:	1 CD
Language:	Czech

OBSAH

1 ÚVOD

Proteom (kombinace slov protein a genom) je ucelený soubor všech proteinů, které se v daném čase a za daných podmínek vyskytují v konkrétní buňce, tkáni či organismu. Proteomika je vědní obor zabývající se studiem struktury, funkce a dalších vlastností proteinů na úrovni proteomu – kromě analýzy struktury a funkce jednotlivých proteinů se zabývá i vlastnostmi celých skupin proteinů sestavených např. na základě podobnosti funkce, lokalizace v organele, podmínek exprese, posttranslačních modifikací aj.

Cílem teoretické části práce je vypracování literární rešerše na téma zpracování výsledků proteomické analýzy s ohledem na predikci jaderné lokalizace zkoumaných proteinů.

Cílem praktické části této práce je vytvoření programu, který usnadní a částečně automatizuje proces anotace nepopsaných proteinů z ječmene setého (*Hordeum vulgare*) identifikovaných na základě analýzy štěpných peptidů technikou tandemové hmotnostní spektrometrie spojené s kapalinovou chromatografií. Hlavní důraz při anotaci bude kladen možnosti vyhledání a zhodnocení informací, zda se neznámé proteiny vyskytují v buněčném jádře. Proces anotace bude založen na použití bioinformatického nástroje pBLAST (protein Basic Local Alignment Search Tool) k nalezení podobných, již popsaných proteinů v databázi Swiss-Prot. Souběžně s tím budou proteinové sekvence analyzovány pomocí nástrojů sloužících k predikci lokalizace (CELLO2GO, LOCALIZER, NucPred a WegoLoc) a k výpočtu základních fyzikálních vlastností proteinů (Protein Cutter). Na základě kombinace predikčních dat s informacemi z databáze bude rozhodnuto, s jakou pravděpodobností jsou jednotlivé proteiny jaderného původu. Ke každému proteinu původního vzorku budou také kombinací fyzikálních a experimentálních dat vypočteny hodnoty SAF (spectral abundance factor) a NSAF (normalised spectral abundace factor), umožňující protein kvantifikovat. Výstupem programu bude tabulka obsahující všechny dílčí informace získané ke každé položce původního seznamu proteinů.

2 SOUČASNÝ STAV ŘEŠENÉ PROBLEMATIKY

2.1 Analýza proteomu

Obecným cílem proteomické analýzy je identifikace struktury, funkce a množství jednotlivých proteinů obsažených v proteomu, což umožňuje následně zkoumat složitější vztahy a vlastnosti, např. posttranslační modifikace, vzájemné proteinové interakce či změny v míře exprese. Nejrozšířenější metody, která se k tomuto účelu v současnosti využívají jsou založeny na technikách hmotnostní spektrometrie.

2.2 Hmotnostní spektrometrie

Hmotnostní spektrometrie je analytická metoda založená na principu separace ionizovaných částic analytu v elektromagnetickém poli. K této separaci dochází na základě rozdílného poměru hmotností a nábojového čísla (m/z) zkoumaných částic, přičemž tento poměr je vyjádřen v daltonech (Da; jeden dalton je hmotnost vodíkového atomu) nebo v atomových hmotnostních jednotkách (jedna jednotka je pak jednou dvanáctinou klidové atomové hmotnosti izotopu uhlíku $^{12}_{6}C$). Jednotka daltonu je nejčastěji využívána biochemiky, přičemž v případě makromolekul se výsledek obvykle vyjadřuje v kilodaltonech (kDa) nebo v megadaltonech (MDa) (Busch, 2003). Výstupem hmotnostního spektrometru je graf závislosti intenzity iontů na jejich m/z, tzv. hmotnostní spektrum.

Hmotnostní spektrometr se skládá ze tří základních částí:

- iontového zdroje, který umožňuje ionizaci analytu a jeho převedení do plynného stavu např. pomocí elektronového paprsku nebo laserového impulsu
- hmotnostního analyzátoru, který na základě odlišných fyzikálních vlastností rozdělí ionty např. pomocí vychýlení jejich trajektorie v elektromagnetickém poli, jedná se tedy o separační proces;
- detektoru, který zaznamenává přítomné částice v závislosti na typu analyzátoru, umožňující jim přiřadit rozdílné m/z a určit jejich relativní zastoupení.

2.2.1 Iontové zdroje

V raných typech hmotnostních spektrometrů se ionizovalo pouze elektrony, jejich kinetická energie obvykle dosahovala hodnoty 70 eV, a to v uzavřeném prostoru s při tlaku okolo 10^{-4} Pa, přičemž tyto hodnoty dostačovaly na formaci a transmisi elektronů do hmotnostního analyzátoru; při vyšším tlaku by docházelo k reakcím mezi elektronizovanými ionty a molekulami plynného média (Chen *et al*., 2014). Při této metodě, která se řadí do skupiny technik tzv. „tvrdé“ ionizace nejčastěji vzniká kladně nabitý radikál molekuly spolu s bohatým množstvím charakteristických fragmentů, kterých lze se následně využít k určení chemické struktury dané sloučeniny. V případě této metody lze proto také s výhodou využít knihoven spekter za účelem interpretace získaného spektra. Elektronová ionizace se proto i dnes využívá k analýze organických molekul, nicméně je nevhodná pro vysokomolekulární a teplotně nestabilní látky (Dass, 2007, Friedecký et Lemr, 2012).

Mezi tzv. „měkké“ ionizační techniky patří nízkoenergetická chemická ionizace, která je svou povahou vhodná k látkám nevhodným pro výše popsanou tvrdou ionizaci elektrony. Při této metodě se prvně ionizuje vysokotlaký reakční plyn (nejčastěji metan, isobutan, amoniak, dimetyleter, diisopropyleter, aceton, acetaldehyd, benzen či iodometan) (Gross, 2004) a od ní následně pak molekula analytu. V porovnání s ionizací elektronem vzniká méně fragmentů. Výhodou chemické ionizace je její univerzálnost a citlivost (je až stonásobně citlivější než elektronová ionizace), navíc se dá využít i ve spojení se separačními metodami, a to kupříkladu v plynové chromatografii, nicméně i zde je třeba těkavých analytů (Dass, 2007, Hunt *et al.*, 2002).

Ještě jemnější technikou je ionizace polem, kdy se vzorek v plynné fázi ionizuje ve velmi silném elektrickém poli pomocí rozžhaveného wolframového vlákna. Zde pak nedochází k téměř žádné fragmentaci, nicméně hrozí tepelný rozklad analytu. Tato metoda se využívá zejména pro ionizaci málo polárních či tepelně nestálých látek (Hoffman et Stroobant, 2007).

Pro aplikaci v proteomice je důležitá technika ionizace vzorku laserem za přítomnosti matrice neboli MALDI (z anglického Matrix Assisted Laser Desorption Ionization), která je často kombinována s analyzátorem doby letu TOF (time-of-flight), jenž umožňuje určit dobu letu jednotlivých částic po definované trajektorii, a z ní pak

vypočítat rychlost dané částice a následně hodnotu m/z. V tomto případě je vzorek spolu s matricí (nejčastěji ve formě aromatické organické kyseliny) nanesen na pevný poklad a ionizován krátkým intenzivním pulsem laseru. Jak již bylo naznačeno, tato metoda byla původně vyvinuta pro kvalitativní určení bílkovin či samotných peptidů, nicméně v současnosti se s výhodou využívá i pro analýzu nukleových kyselin či látek o nízké molekulární hmotnosti. Mezi hlavní přednosti MALDI patří vysoká citlivost a rychlost (Friedecký et Lemr, 2012, Lemaire *et al.*, 2006).

V neposlední řadě se využívá ionizace elektrosprejem. Tato metoda se využívá k analýze polárních molekul včetně bílkovin, oligonukleotidů, polárních lipidů či sacharidů (Mann *et al*., 2001). Proces ionizace elektrosprejem spočívá v kumulaci náboje na povrchu kapiček analytu, které se neustále zmenšují působením sušícího plynu až dojde k uvolnění samotných iontů. Náboj je na kapičky analytu vnesen pomocí kapiláry, na kterou je vloženo vysoké napětí a je jí sprejován tekutý analyt před vstupem do hmotnostního spektrometru. Ionizace elektrosprejem našla uplatnění ve spojení s kapalinovou chromatografií (Fenn *et al.*, 1989, Takats, 2004).

2.2.2 Hmotnostní analyzátory

Před samotnou detekcí je nutné ionty rozdělit podle poměru hmotnosti a náboje m/z. K tomuto účelu slouží hmotnostní analyzátory. Podobně jako u iontových zdrojů existuje celá řada technik, kterými lze měřit hmotnost částic procházejících hmotnostním spektrometrem, v současné proteomice se využívají čtyři základní typy hmotnostních analyzátorů, a to kvadrupólový hmotnostní analyzátor, iontová past, detektor doby lety neboli výše zmíněný TOF a iontová cyklotronová rezonance s Fourierovou transformací (Han *et al.*, 2008; Aebersold et Mann, 2003). Všechny jmenované analyzátory mohou fungovat samostatně, nebo v tandemu.

Kvadrupólový hmotnostní analyzátor patří mezi nejčastěji používané hmotnostní analyzátory. Jedná se o soubor čtyř rovnoběžných tyčí, kterými proudí stejnosměrné i vysokofrekvenční střídavé napětí, přičemž tyče ležící proti sobě mají vždy stejnou polaritu. Podle velikosti stejnosměrného napětí a amplitudy střídavého napětí ve stejném momentu dochází k pohybu iontů s danou hodnotou m/z po stabilní trajektorii směrem k dalšímu analyzátoru nebo rovnou k detektoru. Ionty s nestabilní dráhou se vychýlí směrem k tyčím, na detektor tedy nakonec nedopadnou (Lane, 2005; Mann *et al.*, 2001).

Iontová past působí na princip záchytu iontů v třídimenzionálním elektrickém poli a jejich následném vypouštění. Samotné zařízení se skládá z prstencové elektrony a dvou koncových elektrod, přičemž past je vyplněna inertním plynem, nejčastěji v podobě helia (Lane, 2005).

Analyzátor doby letu (TOF) se od ostatních metod hmotnostní analýzy odlišuje tím, že elektrické pole používá pouze k urychlení iontů, které následně procházejí trubicí o dané délce a dopadají na detektor. Měří se doba, po kterou trval průlet iontu. Jelikož náboj elektrického pole je pevně dán a rychlost částic závisí na jejich poměru hmotnosti a náboje (těžší částice mají nižší rychlost letu), lze takto přesně určit daný ion (Lane, 2005).

2.2.3 Detektory

Poslední komponentou hmotnostních spektrometrů jsou detektory částic. Ty fungují na principu detekce indukovaného náboje nebo elektrického proudu na ploše, do které narazí nabitý ion, který prošel hmotnostním analyzátorem. Podle pozice detektoru a času, ve kterém došlo k zaznamenání dopadu, lze přesně určit poměr m/z dopadajícího iontu a tím i kvantifikovat množství iontů o dané m/z, které se ve vzorku nacházejí. Výstupem měření je graf, ve kterém jsou na ose x vyneseny hodnoty m/z a na ose y je pro každou z těchto hodnot vynesena zaznamenaná četnost, představuje tedy tzv. hmotnostní spektrum.

2.2.4 Kapalinová chromatografie a tandemová hmotnostní spektrometrie

Určení složení proteinu na základě jednoduché hmotnostní spektrometrie by vzhledem k velikosti a komplexitě proteinů bylo velmi složité. Z toho důvodu se k proteomické analýze používá kombinace metod hmotnostní spektrometrie a kapalinové chromatografie. Principem teto metody je enzymatické štěpení proteinů na peptidy, jejich separace pomocí kapalinové chromatografie a následně provedení několika měření hmotnostní spektrometrii za sebou. Prvním měřením dojde k detekci a následné separaci peptidů o různých hodnotách s m/z, tzv. Prekurzoru,v následujícím druhém měření dochází k analýze fragmentu, které vzniknou rozbitím prekurzoru na různě dlouhé fragmenty aminokyselin představující část sekvence prekurzoru . Vhodným výběrem proteolytických enzymů lze dosáhnout toho, že je možné získat definované

predikovatelné peptidy a umožňující na zaklade odečtení rozdílu m/z mezi jejich fragmenty tomuto rozdílu přiradit aminokyselinový zbytek. Takto je možné přečíst pořadí aminokyselin v peptidech za sebou, což v kombinaci s moznosti separace složitějších smesi peptidu pomoci spojeni s kapalinovou chromatografii umožňuje i přečtení celé proteinové sekvence a také identifikaci více proteinů z jejich složitějších peptidových směsí (Hunt *et al.,* 1986)

2.3 Nástroje pro analýzu a charakterizaci proteinových sekvencí

2.3.1 Basic Local Alignment Search Tool

BLAST je název algoritmu, který se využívá pro srovnávání primárních sekvenčních dat nukleotidů či aminokyselin. Umožňuje porovnání jedné sekvence s celou databází a nalezení sekvence se zadanou či vyšší mírou podobnosti. Byl vyvinut v roce 1990 a stal se jedním z nejpoužívanějších bioinformatických nástrojů současnosti. (Altschul *et al.,* 1990)

2.3.2 Swiss-Prot

Swiss-Prot je manuálně anotovaná neredundantní databáze proteinových sekvencí a všech informací k nim relevantních. Je součástí komplexní proteinové databáze UniProtKB vytvořené konsorciem UniProt sestávajícím z Evropského bioinformatického institutu (EBI), Švýcarského institutu bioinformatiky (SIB) a společnosti Protein Information Resource (PIR).

Údaje o proteinech obsažené v databázi Swiss-Prot pocházejí výhradně z vědecké literatury nebo z výpočetních modelů, které byly ověřeny kurátory databáze. Všechny záznamy jsou navíc pravidelně kontrolovány a případně upraveny podle nejnovějších dostupných dat. Díky tomu představuje tato databáze vysoce spolehlivý zdroj informací.

Mimo samotných sekvencí aminokyselin obsahuje Swiss-Prot i popisy funkce proteinů, jejich doménové struktury, posttranslační modifikace, termíny genové ontologie a mnoho dalších.

2.3.3 Predikční nástroje

CELLO2GO je veřejně dostupný systém určený k odhadování vlastností a lokalizace cílových proteinů na základě kombinace vyhledávání homologů v databázích a predikce buněčné lokalizace na základě metody strojového učení. (Yu *et al*., 2014)

Na principu rozpoznávání lokalizace pomocí algoritmu strojového učení pracují i nástroje LOCALIZER (který je specificky trénován na rostlinné proteiny, zatímco CELLO2GO rozpoznává obecnější eukaryontní sekvence spojené s buněčnou lokalizací) a WegoLoc (Sperschneider *et al*., 2017; Chi *et* Nam, 2012)

Nástroj NucPred je specializován čistě na predikci, s jakou pravděpodobností stráví analyzovaný protein alespoň nějakou dobu uvnitř buněčného jádra. Je taktéž postaven na metodě strojového učení, kdy jsou přiřazovány určité regulární výrazy ke

zkoumané aminokyselinové sekvenci. Výstupem nástroje je desetinné číslo mezi 0 a 1 které představuje jaderné skóre. Hodnota 0,5 značí dle tabulky poskytnuté autory nástroje 70 % specifitu a 62 % senzitivitu k rozpoznávání jaderné lokalizace zkoumaného proteinu (Brameier *et al*., 2007)

Nástroj Protein Cutter byl vytvořen na oddělení biochemie proteinů a proteomiky UP v Centru Regionu Haná panem doktorem Rausem a panem profesorem Šebelou. Slouží ke zjištění základních fyzikálních charakteristik proteinových sekvencí (např. délka proteinu dle počtu aminokyselin, výpočet izoelektrického bodu aj.) na základě výpočtu z fyzikálních hodnot jednotlivých aminokyselin, ze kterých je protein složen.

3 PRAKTICKÁ ČÁST

3.1 Použité technologie

3.1.1 Python

Python je vysokoúrovňový, multiparadigmatický, dynamicky interpretovaný programovací jazyk. Byl vyvinut mezi lety 1989 a 1991 nizozemským programátorem Guido van Rossumem z Národního výzkumného institutu pro matematiku a informatiku (CWI) v Amsterdamu. V principu je podobný výukovému programovacímu jazyku ABC, na jehož vývoji van Rossum pracoval v polovině 80. let. Python má kompaktní jádro základní funkcionality nazývané *standard library*. Tuto základní knihovnu je možné snadno rozšířit dodatečnými balíčky, jako je například knihovna pro vědecké výpočty NumPy, nebo TensorFlow pro strojové učení. Výraznou charakteristikou jazyka je minimalizace používání závorek – namísto nich je zde pro vyznačení vnitřních částí cyklů, podmínek a funkcí používáno tabulátorové odsazení, což zvyšuje přehlednost kódu. Oproti staticky typovaným či kompilovaným jazykům jako jsou C++ a Java jsou programy psané v Pythonu typicky třetinové až pětinové délky a umožňují okamžité spuštění bez předchozí kompilace, což značně zvyšuje efektivitu práce. (Lutz *et* Ascher, 2003)

3.1.2 Biopython

Biopython je název volně dostupné open-source knihovny bioinformatických nástrojů pro jazyk Python, jejímž účelem je zprostředkovat či usnadnit práci s nukleotidovými a proteinovými sekvencemi či záznamy z biologických databází (např. Swiss-Prot, GenBank, Prosite). Většinu bioinformatických dat Biopython reprezentuje pomocí datových struktur (v Pythonu nazývaných třídy), což umožňuje rychlé čtení či zápis a následnou práci se sekvencemi či anotacemi v těchto třídách obsažených. Mimo to obsahuje Biopython řadu podknihoven, které rozšiřují jeho funkcionalitu o provádění operací s bioinformatickými daty. Pro sekvenční data jsou to např. funkce sekvenčního přiložení, nalezení reverzního komplementu, provedení transkripce či translace (Zdrojový kód 1).

```
>>> from Bio.Seq import Seq
>>> from Bio.Alphabet import IUPAC
>>> sekvence_dna = Seq('ACCCATAGAGACATAGACCACCGGTAT', IUPAC.unambiguous_dna)
>>> sekvence_dna.reverse_complement()
Seq('ATACCGGTGGTCTATGTCTCTATGGGT', IUPACUnambiguousDNA())

>>> sekvence_dna.transcribe()
Seq('ACCCAUAGAGACAUAGACCACCGGUAU', IUPACUnambiguousRNA())

>>> sekvence_dna.translate()
Seq('THRDIDHRY', IUPACProtein())
```

Zdrojový kód 1: Vytvoření objektu sekvence_DNA obsahujícího nukleotidovou sekvenci a následné operace nalezení komplementu, transkripce a translace v jazyce Python za použití knihovny Biopython.

3.2 Vstupní data

Vstupními daty programu jsou dva soubory. Prvním je tabulka naměřených fyzikálních dat získaných pro proteiny identifikované na základě analýzy tryptických peptidů z proteinů extrahovaných z 10 milionů mitotických chromozomů technikou tandemové hmotnostní spektrometrie online spojené s nanoprůtokovou kapalinovou chromatografií na reverzní fázi C18. Tato data byla získána zpracováním výstupních spektrometrických dat ve formátu mgf (mascot generic format) pomocí nástroje Search GUI 3.3.13.

Použité algoritmy: X! Tandem; MS Amanda; OMSSA; při zadané přesnosti měření 50 ppm pro prekurzorové ionty, 0,05 Da pro ionty fragmentů; povolené modifikace peptidů: variabilní: acetylace N-konce proteinů, oxidace methioninu, deaminace Q a N; fixní modifikace: karbamidomethylace cysteinu Jako proteasa byl zadán trypsin, povoleno bylo pouze specifické štěpení na obou koncích polypeptidového řetezce a maximalně 2 přeskočená štěpná místa. Takto zpracovaná data byla prohledána programem Peptide shaker 1.16.38. Pozitivní identifikace byla omezena parametrem FDR (False Discovery Rate) <1 % na úrovni přiřazených spekter, na úrovni identififikovaných peptidů i proteinů. Použitá databáze proteinových sekvencí obsahovala celkem 39850 položek včetně běžných kontaminantů.

Druhým vstupem programu je FASTA soubor obsahující pouze aminokyselinové sekvence nalezených proteinů odpovídající položkám ze vstupní tabulky na základě jejich identifikátoru. Vstupní tabulka neobsahuje aminokyselinové sekvence identifikovaných proteinů.

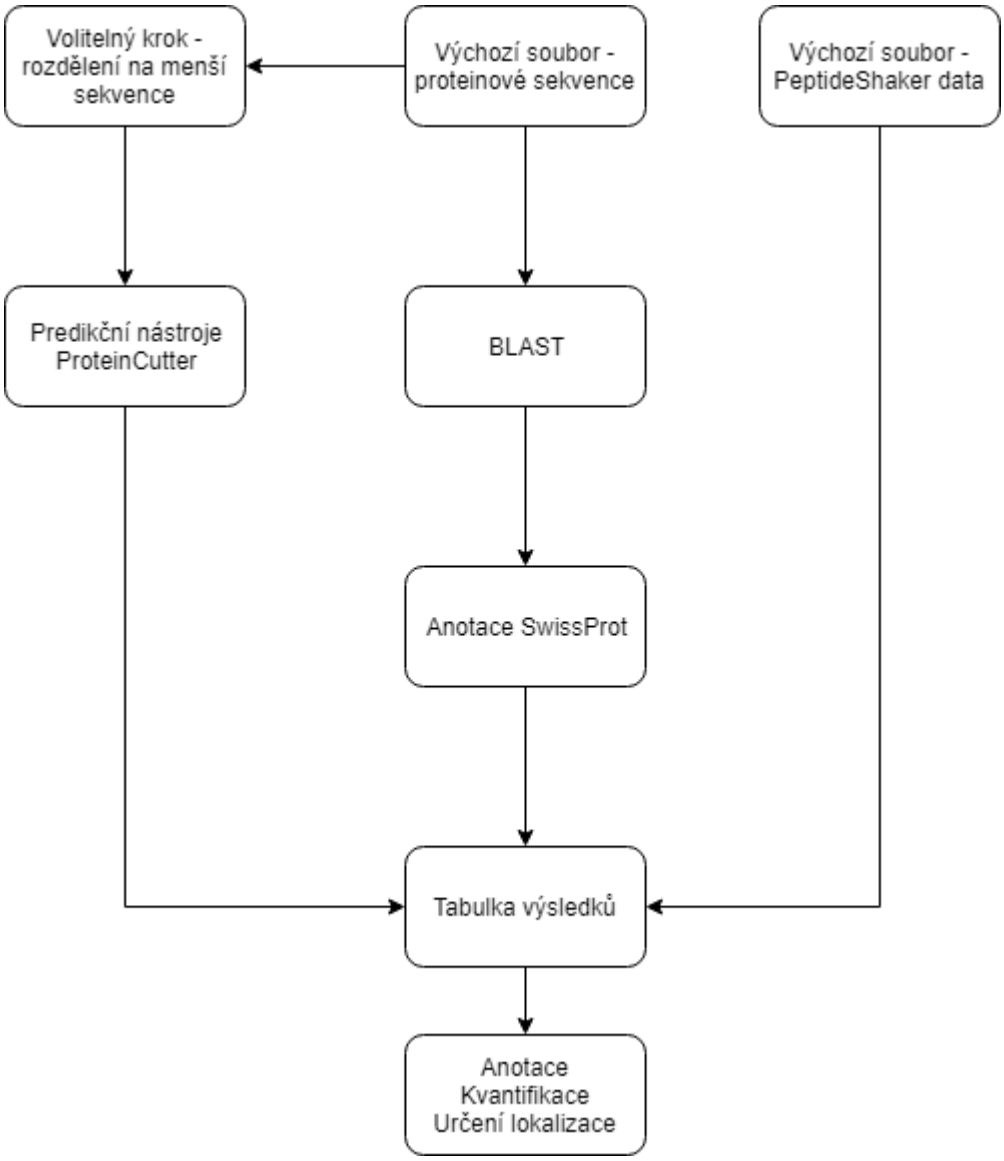
3.3 Návrh programu

Hlavním cílem navrhovaného programu je minimalizovat množství práce, kterou musí uživatel vynaložit na anotaci a určení lokalizace zkoumaných proteinových sekvencí. Tohoto cíle lze dosáhnout automatizací co největší části pracovního procesu a uživatelského usnadnění těch dílčích úkonů, které nelze jednoduše automatizovat.

Pro anotaci proteinů dle zadaných specifikací je nutné provést pět hlavních úkonů:

- Nalezení homologních proteinů metodou BLAST v databázi Swiss-Prot
- Určení pravděpodobné lokalizace analyzovaných proteinů pomocí predikčních nástrojů. K provedení tohoto kroku může být nutné vstupní soubor se sekvencemi rozdělit na menší, jelikož některé nástroje umožňují predikci jen pro omezené množství sekvencí
- Spojení dílčích výsledků do jedné tabulky
- Určení potenciálně jaderných proteinů na základě shody mezi predikčními a databázovými údaji
- Kvantifikace proteinů ve vzorku pomocí výpočtu NSAF

Postup je znázorněn na obrázku č. 1.



Obrázek 1: Pracovní postup při anotaci proteinových sekvencí.

Bioinformatická knihovna Biopython umožňuje téměř úplnou automatizaci prvního kroku pomocí funkcí podporujících přímé vyhledávání homologních proteinových sekvencí pomocí nástroje BLAST a získání dostupné anotace těchto homologů v databázi Swiss-Prot.

Zvolené predikční nástroje podobný přístup neumožňují, proto je nutné je využívat skrz dostupné webové rozhraní. Automatizace by v tomto případě vyžadovala využití metod automatizace webového prohlížení (*browser automation*) a těžby dat (*data mining*), tudíž by byla neúměrně složitá. Použití predikčních nástrojů tedy bude krokem, kde bude i s využitím navrhovaného programu nutný zásah uživatele.

Spojení dílčích výsledků do jednoho souboru je z praktického hlediska pouze problémem manipulace s textovými daty a třídění na základě shody jedné položky (v tomto případě identifikátoru proteinů), proto lze tento úkon zcela automatizovat.

Problém určení lokalizace proteinů na základě výsledků predikčních nástrojů a nalezeného homologního proteinu lze zredukovat na algoritmus, který vyhodnotí řadu logických formulí, a na základě jejich hodnot určí výsledek.

Kvantifikace proteinů ve vzorku z hmotnostního spektrometru se provádí výpočtem SAF z počtu validovaných přiřazení peptidů ke spektrům (*validated peptide-spectrum matches*, VPSMs) a počtu aminokyselin (*protein length*, l), ze kterých se daný protein skládá (Vzorec 1).

$$SAF = \frac{VPSMs}{l}$$

Vzorec č. 1: Výpočet faktoru spektrálního přebytku.

Ze všech hodnot SAF v daném vzorku se následně hodnota NSAF pro každý protein vypočítá jako podíl SAF daného proteinu a sumy hodnot SAF všech proteinů přítomných ve vzorku (Vzorec 2).

$$NSAF_i = \frac{SAF_i}{\sum_{k=1}^{N} \mathrm{SAF}_k}$$

Vzorec č. 2: Výpočet normalizovaného faktoru spektrálního přebytku.

Výpočet výše uvedených hodnot lze kompletně provést uvnitř programu, čímž je dále usnadněna činnost uživatele.

Tímto je pokryt celý pracovní postup anotace. Jelikož nalezení homologních proteinů a predikce lokalizace může pro obsáhlé vzorky dat trvat řádově i hodiny, je vhodné program rozdělit na samostatně spustitelné moduly, které se budou zabývat jen určitými částmi pracovního postupu. První z těchto modulů by měl mít na starost vhodnou úpravu dat pro další analýzu.

Jelikož pořadí proteinů v obou vstupních souborech nemusí být stejné, ale zároveň žádný z predikčních nástrojů pořadí sekvencí, které jsou do nich vloženy nemění, je

z hlediska efektivity nejvhodnější, aby prvním krokem bylo seřazení vstupních souborů tak, aby během slučování dílčích dat nebylo nutné v každém souboru vyhledávat správnou sekvenci, a místo toho je jednoduše spojit za sebe. Po seřazení dat následuje (v případě, že je pro některý z nástrojů objem dat příliš velký) jejich rozdělení na menší soubory a pak samotné využití predikčních nástrojů a nalezení homologů. Protože hledání homologů metodou BLAST a vyhledáváním v databázi Swiss-Prot lze automatizovat, program by měl být navržen tak, aby umožňoval jejich paralelní provedení.

Druhý samostatně spustitelný modul programu by měl být schopen spojit dílčí výsledky jednotlivých predikčních nástrojů do jednoho souboru.

Třetí modul pak bude sloužit k závěrečnému sloučení všech dat do jednoho souboru, určení konsenzu predikčních nástrojů ohledně lokalizace proteinu a k výpočtu hodnot SAF a NSAF.

4 VÝSLEDKY A DISKUZE

4.1 Program ProteinScout

Pro účely této práce byl vyvinut program ProteinScout. Je psán v jazyce Python verze 3.7.2 rozšířeném o bioinformatický balíček Biopython verze 1.73. Program sestává ze tří samostatně spustitelných modulů, které činností odpovídají funkčnímu rozdělení dle výše uvedeného návrhu. Prvním vstupem programu je soubor proteinových sekvencí ve formátu FASTA, kdy každá sekvence na prvním řádku obsahuje identifikační záhlaví (začínající symbolem „>“) a na druhém řádku sekvenci aminokyselin zapsanou standardní proteinovou jednopísmennou abecedou obohacenou o symbol X značící libovolnou aminokyselinu. Soubor se sekvencemi je klíčový k nalezení homologních proteinů a predikci lokalizace.

Druhým vstupem programu je tabulátorem oddělovaný soubor dat získaných z programu PeptideShaker, který obsahuje zpracované informace z hmotnostního spektrometru. Tento soubor vedle fyzikálních charakteristik proteinů obsahuje i údaj o PSMs, který je klíčový ke kvantifikaci proteinů ve vzorku.

4.1.1 Modul 1

Práce se vstupními daty je zahájena spuštěním modulu 1. Účelem modulu 1 je vytvořit adresářovou strukturu pro dílčí a závěrečné výsledky, setřídit sekvence ve vstupních souborech tak, aby bylo jejich pořadí identické (což výrazně usnadní závěrečné spojování souborů do výsledku), a na závěr provést proteinový BLAST a následnou anotaci vstupních sekvencí pomocí homologních proteinů nalezených v databázi Swiss-Prot.

Uživatel specifikuje vstupní FASTA a PeptideShaker soubory a novou či již existující složku, do které budou ukládány dílčí výsledky. Dále je možné zadat volitelnou identifikační předponu vznikajících souborů. Modul 1 ve zvolené složce vytvoří podsložku WEBTOOL_RESULTS pro dílčí výsledky (slučovány modulem 2) a podsložku FINAL pro soubory připravené k závěrečné anotaci (provedené modulem 3).

Následuje načtení vstupních FASTA dat. Pořadí sekvencí v prvním a druhém vstupním souboru se může lišit a je nutné je před zahájením vlastní analýzy vhodně seřadit. Vstupní soubory mohou obsahovat tisíce sekvencí, a proto není efektivní je řadit

procházením celých souborů a hledáním shody. Z tohoto důvodu bylo využito funkce Pythonu umožňující ukládání dat do slovníku ve formě hašovací tabulky. Tento způsob zpracování dat zároveň výrazně urychluje následné vyhledávání (Cormen *et al.*, 2009).

Seřazený FASTA soubor je poté uložen do cílové složky. Tento soubor může uživatel zadat jako vstup pro jednotlivé webové nástroje zmíněné v kapitole 2. Pokud soubor obsahuje více než 500 sekvencí, modul 1 ve složce WEBTOOL_RESULTS vytvoří odpovídající počet rozdělených a vzestupně očíslovaných FASTA souborů. Důvodem k této parcelizaci souboru je omezení některých webových nástrojů pro maximální počet vložených proteinových sekvencí, které lze analyzovat v jednom FASTA souboru. Uživatel v takovém případě data k analýze nahrává a výsledky z k nim ukládá průběžně.

Souběžně s touto činností uživatele modul 1 pokračuje zahájením proteinového BLASTu. Uživatel může nastavit parametry vyhledávání nebo využít rychlé nastavení, které bylo vytvořeno na základě specifických požadavků pro tento účel (tzn. zaznamenán je pouze nejlepší výsledek, skórovací matice BLOSUM62, nastavení gap costs 11-1, velikost slova 6, vyhledávání v databázi Swiss-Prot s výsledky omezenými pouze na proteiny z Arabidopsis thaliana). Modul 1 k provedení BLAST využívá knihovnu Biopythonu Bio.NCBIWWW, která obsahuje funkci qblast(). Ta zprostředkuje zaslání FASTA sekvencí do systému NCBI QBLAST (URL 1), který provede BLAST se zadanými parametry a vrátí výsledná data ve formátu XML (Zdrojový kód 1). Podobně jako při využití nástroje BLAST přímo na webových stránkách NCBI závisí doba trvání této operace na vytížení serverů. Takto získaná data lze poté snadno převést na seznam objektů v Pythonu pomocí funkce parse() z Biopython knihovny Bio.SearchIO. Každý objekt v seznamu pak představuje jeden nalezený homologní protein.

```
entrquery = '"Arabidopsis thaliana"[organism]'

# Spuštění nástroje BLAST se zadanými parametry pro sekvence ze souboru mainfastafile.
result_handle = NCBIWWW.qblast(program="blastp",
                               database="swissprot",
                               sequence=mainfastafile.read(),
                               entrez_query=entrquery,
                               hitlist_size=1)

xml_filename = prepid+'blast_raw.xml'
save_file = open(xml_filename, "w")
save_file.write(result_handle.read()) # Uložení výsledků do výstupního XML souboru
save_file.close()
mainfastafile.close()
result_handle.close()
```

Zdrojový kód 1: Využití funkce qblast() k získání výsledků BLAST v jazyce Python.

Modul 1 následně z tohoto seznamu po jednom extrahuje přístupové kódy homologů nalezených ve Swiss-Prot databázi (Swiss-Prot accession number). Tyto přístupové kódy jsou následně předány jako argument funkci get_sprot_raw() z knihovny Bio.ExPASy, která k danému přístupovému kódu nalezne příslušný záznam ze Swiss-Prot databáze a vrátí jej jako objekt. Z tohoto objektu nakonec program zapíše potřebné informace do výstupního souboru BLAST_tab.tsv (Zdrojový kód 2).

```
with open(tsv_filename, mode='w', newline='') as BLAST_input: # Otevření nového souboru pro výstupní data
    output_writer = csv.writer(BLAST_input, delimiter='\t', quotechar='"', quoting=csv.QUOTE_MINIMAL)
    # Zápis záhlaví výstupní tabulky
    output_writer.writerow(['Entry', 'SwissProt Entry', 'Gene names', 'Protein names', 'Status',
                            'Organism', 'e-value', 'Bitscore', '% identity', '% positives',
                            'Gene ontology(biological process)', 'Gene ontology(cellular component)',
                            'Gene ontology(molecular function)', 'Subcellular location [CC]'])

    qresults = SearchIO.parse(xml_filename, 'blast-xml')  # načtení XML souboru s výsledky BLAST
    queryno = 0
    for qresult in qresults:
        try:  # Postupný průchod jednotlivými záznamy, hledání, zda v objektu existuje výsledek
            tophit = qresult[0]
            has_results = True
        except:
            has_results = False
        # Pokud výsledek existuje, jsou potřebná data uložena a je zavolána funkce pro přístup do SwissProt
        if has_results:
            tophsp = qresult[0][0]
            seqlen = tophit.seq_len       # Délka shody sekvence
            positives = round(((tophsp.pos_num / seqlen) *100), 4)# Pozitivita výsledku vyjádřená v procentech
            identity = round(((tophsp.ident_num / seqlen) * 100), 4)# Identita výsledku vyjádřená v procentech
            evl = tophsp.evalue                                    # Chybová hodnota e-value
            btscr = tophsp.bitscore                                # Skóre výsledku
            swissacc = tophit.accession                            # Přístupový kód SwissProt
            swisshandle = ExPASy.get_sprot_raw(swissacc)           # Stažení záznamu z databáze SwissProt
            records = SwissProt.parse(swisshandle)                 # Přepsání staženého záznamu do objektu
            for record in records:
                datatype = record.data_class # Typ záznamu ("Verified")
                organism = record.organism   # Zdrojový organismus
                var = str(record.gene_name)  # Textové pole se všemi názvy a zkratkami asociovaného genu
                genestr = strappend(textcutter(var, r"\S*=", ";"))  # Extrakce pouze potřebných názvů
                var = str(record.description)             # Textové pole s popisem struktury a funkce proteinu
                proteinstr =strappend(textcutter(var, r"\S*=", ";"))# Extrakce pouze potřebných částí textu
                var = str(record.comments)
                # Extrakce údaje o lokalizaci proteinu v buňce
                ccstr = strappend(textcutter(var, "SUBCELLULAR LOCATION: ", "."))
                cmpstr=''
                funstr=''
                prcstr=''
                var = str(record.cross_references)
                xrefsource = textcutter(var, "'GO', '", "I", "', '", 2) # Oddělení termínů genové ontologie
                for x in xrefsource:
                #  ověření, zda se GO termín týká procesu, funce či organely
                    if x[1][0]=='C':
                        cmpstr=cmpstr+str(x)+separator
                    elif x[1][0]=='F':
                        funstr=funstr+str(x)+separator
                    elif x[1][0]=='P':
                        prcstr=prcstr+str(x)+separator
                    else:
                        continue # GO termín se netýká ani jedné ze sledovaných oblastí
            queryno = queryno + 1 # inkrementace čítače záznamů
            print("%i\t\tQuery %s\t\tsuccessfully annotated at %s.\n" % (queryno, qresult.id, datetime.now()))
            # zápis záznamu do výsledné tabulky
            output_writer.writerow([qresult.id, swissacc, genestr, proteinstr, datatype, organism, evl, btscr,
                                    identity, positives, prcstr, cmpstr, funstr, ccstr])

        else:
            # Záznam neobsahuje žádný výsledek, do tabulky je uloženo pouze původní ID neznámé sekvence
             queryno = queryno + 1
            output_writer.writerow([qresult.id])
```

Zdrojový kód 2: Proces vyhledávání a zápisu dat z databáze Swiss-Prot v jazyce Python.

4.1.2 Modul 2

Funkcí tohoto modulu je sloučení dílčích výsledků z nástrojů CELLO2GO, LOCALIZER, NucPred a WegoLog do jednoho souboru pro každý webový nástroj. Platným vstupem programu je adresa složky, do které uživatel uložil dílčí výsledky ve formátu textových souborů. Tyto textové soubory musejí být pojmenovány v následujícím formátu:

- libovolná předpona následována výrazem ‚result_‘
- název webového nástroje, ze kterého výsledek pochází (např. ‚nucpred_‘)
- dvouciferné číslo, které označuje pořadí, v jakém mají být soubory sloučeny. Toto číslo musí být shodné s číslem dílčího FASTA souboru, ze kterého výsledek vznikl
- přípona ‚.txt‘. Výstup všech použitých webových nástrojů je buď v tzv. plaintext nebo v tabulátorem odděleném formátu, které lze bezpečně otevřít v textovém editoru

Výstupem modulu 2 jsou textové soubory pojmenované jako ‚FINAL_‘ a název nástroje, z jehož souborů vznikly (např. ‚FINAL_wegoloc.txt‘). Tyto soubory následně uživatel může přesunout do složky FINAL, kde dojde k jejich závěrečnému sloučení a vyhodnocení modulem 3.

Při použití modulu 2 je důležité dodržet správné číslování jednotlivých souborů, které jsou z důvodu časové a výpočetní úspory tříděny pouze na samotném počátku zpracování vstupních dat modulem 1. Nedodržení číslování souborů vede k předčasnému zastavení chodu modulu 3, aby se předešlo nesprávnému vložení jednotlivých záznamů do výsledné tabulky.

4.1.3 Modul 3

Funkcí tohoto modulu je extrakce všech dosavadních výsledků, jejich spojení do jednoho souboru a závěrečné vyhodnocení. Vstupem je uživatelem specifikovaná složka obsahující všech sedm souborů připravených k závěrečné analýze (tzn. výstupy z pěti webových nástrojů, tabulku s výsledky BLAST a textový soubor z programu PeptideShaker). Druhým vstupem je pak prahová hodnota pro výstupní skóre z nástroje NucPred, jejíž účel byl popsán v kapitole 2.

Modul před samotnou závěrečnou anotací souběžně otevře a přečte soubory z nástroje Protein Cutter a PeptideShaker a z dat v nich obsažených uloží do seznamu hodnoty délky proteinů, tj. počet aminokyselin tvořící každý protein, a počty validovaných PSMs. Důvodem pro tuto operaci je použití těchto hodnot pro výpočet SAF a NSAF. Pro výpočet NSAF je nutné znát sumu všech hodnot SAF z celého vzorku, a proto nelze tento úkon provést souběžně se závěrečnou anotací, která z důvodu úspory paměti probíhá vždy po jednom řádku. Před vlastní anotací tedy modul vypočte a do seznamu uloží všechny hodnoty SAF a jejich sumu. Z těchto hodnot pak do nového seznamu vypočítá všechny hodnoty NSAF (Zdrojový kód 3).

Následuje závěrečná anotace. Modul otevře všech sedm vstupních souborů a do výstupního souboru zapíše záhlaví. Následně postupuje řádek po řádku, kdy z každého vstupního souboru načte potřebná data. Toto čtení je zajištěno funkcemi s příponou single (např. ‚wegolocsingle‘), které přijímají celý řádek textu ze vstupního souboru a na výstupu vrací jednotlivé hodnoty získané buď pomocí regexové formule nebo rozdělením přes oddělovač funkcí partition() (Zdrojový kód 4).

```
# Načtení seznamu hodnot PSMs pomocí funkce pro extrakci hodnot ze souboru PeptideShaker
# Proměnná True signalizuje, že místo hodnoty pro jeden řádek funkce vrací pole všech hodnot
valpsmsarr = shakersingle(shakerfile, True)

# Načtení seznamu délek proteinů pomocí funkce pro extrakci hodnot ze souboru Protein Cutter
protlengtharr = protcutsingle(proteincutterfile, True)

NSAFarr = [] # Inicializace seznamu výsledků NSAF

# Vytvoření seznamu výsledků SAF vydělením hodnoty validovaných PSMs a délky proteinu,
# zaokrouhleno na 10 desetinných míst
SAFarr = [round(x/y, 10) for x, y in zip(valpsmsarr, protlengtharr)]

# Výpočet sumy všech hodnot SAF
nsafcoef = round(sum(SAFarr), 10)
```

Zdrojový kód 3: Část hlavního kódu modulu 3 zajišťující výpočet SAF a NSAF v jazyce Python

```
def wegolocsingle(inputfile):
    """Vrací seznam s hodnotami ID a buněčné lokalizace pro jeden řádek zdrojového souboru"""
    checkline = inputfile.readline() # Přečtení řádku ze vsttupního souboru
    while checkline: # Ověření existence dat na řádku
        try:
            eofbool = checkline[0].isdigit() # Ověření, že řádek obsahuje výsledek
                                             #(řádky s výsledky začínají číselným pořadím)
        except:
            return None # Přerušení nahrání, funkce vrací hodnotu None
        if eofbool:     # Řádek obsahuje výsledek
            resfull = checkline.partition("\t ")[2] # Oddělení výsledku od pořadí
            resfull = resfull.partition("\t") # Oddělení složek výsledku od sebe
            resid = resfull[0] # Uložení první složky výsledku jako ID
            resloc = resfull[2].partition("\t")[0] # Uložení druhé složky jako výsledku lokalizace
            resfull = [resid, resloc] # Spojení složek výsledku do seznamu
            return resfull # Výstup funkce
        else: # Řádek neobsahoval výsledek, tzn. aktuální pozice v souboru je buď přede všemi výsledky,
              # nebo v oblasti mezi jednotlivými spojenými výsledky. Funkce se v tom případě posunuje
              # na další řádky dokud nenarazí na výsledek nebo konec souboru
            continue
```

Zdrojový kód 4: Funkce pro načtení výsledků z výstupního souboru predikčního nástroje WegoLoc psaná v jazyce Python

Po dokončení anotace jsou data zápsána do výstupní tabulky. Ta je formátována jako tabulkový .csv soubor používající středník jako oddělovač. Důvodem k tomuto formátování je jeho snadná konverze na formát .xlsx MS Excel, který umožňuje přidání dodatečných funkcí, jako je filtrování či řazení tabulky dle hodnot jednotlivých sloupců.

Modul napřed zapíše záhlaví výsledné tabulky složením záhlaví dílčích souborů a přidáním dalších sloupců pro nové výsledky. Následně je zahájen cyklus, ve kterém jsou vždy vykonány následující pokyny:

- načtení jednoho řádku dat ze všech vstupních souborů
- ověření, že data na novém řádku existují (pokud ne, je cyklus ukončen)
- ověření, že se identifikátor sekvence shoduje ve všech souborech
- určení povahy proteinu ze vstupních dat pomocí funkce judge()
- uložení vstupních dat, výpočtu SAF a NSAF pro daný řádek a výstupu funkce judge() na nový řádek výstupního souboru

Funkce judge() představuje automatizaci manuálního rozhodovacího procesu, kterým byly ve výsledné tabulce označeny proteiny, které jsou pravděpodobně lokalizovány v buněčném jádře a případně proteiny jejichž funkce souvisí se strukturou či funkcí chromozomů. Funkce judge() jako vstup přijímá hodnoty ze čtyř predikčních nástrojů, textové pole obsahující všechny termíny genové ontologie cílového proteinu a prahovou hodnotu pro nástroj NucPred. Na základě míry shody jednotlivých údajů funkce klasifikuje protein jedním z následujících hodnocení:

- Validated nuclear – protein se dle termínů GO vyskytuje v buněčném jádře, a navíc jej jako jaderný vyhodnocují alespoň tři ze čtyř predikčních nástrojů
- Possibly chromosomal – protein má dle termínů GO funční souvislost s chromozomy
- Combined nuclear – protein se dle termínů GO vyskytuje v buněčném jádře a jako jaderný jej vyhodnocují dva ze čtyř predikčních nástrojů
- Predicted nuclear – protein nemá přítomen žádný záznam v databázi Swiss-Prot a zároveň jej jako jaderný vyhodnocují alespoň tři ze čtyř predikčních nástrojů
- Discrepancy – UniProt – protein se dle termínů GO nevyskytuje v jádře, ale jako jaderný jej vyhodnocují alespoň tři ze čtyř predikčních nástrojů
- Discrepancy – Prediction – protein se dle termínů GO vyskytuje v jádře, ale jako jaderný je vyhodnocen pouze jedním z predikčních nástrojů
- Neklasifikováno – protein nesplňuje podmínky žádné z předchozích klasifikací. Do této kategorie náleží zaprvé proteiny, které nemají ani pozitivní záznam v databázi Swiss-Prot, ani nejsou predikovány jako jaderné

Tohoto vyhodnocení je ve funkci judge() dosaženo výpočtem predikčního skóre proteinu, které může nabývat celočíselné hodnoty 0 až 4 (podle počtu predikčních nástrojů, které protein označují jako jaderný), určením, zda jsou v záznamu přítomny jaderné termíny GO a následným vyhodnocením série logických formulí (Zdrojový kód 5).

```
# Funkce pro rozhodnutí, zda je protein  jaderný
# Vstupní proměnné: Textové pole s GO termíny, % podobnosti se SwissProt sekvencí,
# prahová % hodnota podobnosti, hodnota NucPred, prahová hodnota NucPred, hodnota WegoLoc,
# hodnota LOCALIZER, hodnota CELLO2GO
def judge(GO, GOperc, GOtresh, nucval, nuctres, wegoval, nlsval, celloval):
    score = 0 # Počáteční skóre je 0
    # Načtení globálních proměnných s počtem doposud klasifikovaných sekvencí
    global validnuc, combnuc, prednuc, disuni, dispred


def judge(GO, GOperc, GOtresh, nucval, nuctres, wegoval, nlsval, celloval):
    score = 0
    global validnuc, combnuc, prednuc, disuni, dispred
    verdictstr = ''

    if GOperc == -1:
        flag = 0 # Není k dispozici SwissProt záznam
    if GOperc >= GOtresh and ('nucleus' in GO or 'Nucleus' in GO)
    and ('chromosom' in GO or 'Chromosom' in Go):
        flag = 1 # SwissProt indikuje jadernou lokalizaci proteinu a možnou chromozomální funkci
        verdictstr = 'Possibly chromosomal - '
    elif GOperc >= GOtresh and ('nucleus' in GO or 'Nucleus' in GO):
        flag = 1 # SwissProt indikuje jadernou lokalizaci proteinu bez chromozomální funkce
    else:
        flag = 2 # SwissProt indikuje nejadernou povahu proteinu
    if nucval >= nuctres:
        score = score + 1 # zvýšení skóre pokud je hodnota NucPred nad prahovou hodnotou
    if 'nucleus' in wegoval:
        score = score + 1 # zvýšení skóre pokud WegoLoc predikuje výskyt v jádře
    if 'Y' in nlsval:
        score = score + 1 # zvýšení skóre pokud LOCALIZER predikuje výskyt v jádře
    if 'Nuclear' in celloval:
        score = score + 1 # zvýšení skóre pokud CELLO2GO predikuje výskyt v jádře

    # Většina predikčních nástrojů i SwissProt se shodují
    if flag == 1 and score >= 3 :
        validnuc = validnuc + 1
        verdictstr = verdictstr + 'Validated nuclear'
        return verdictstr

    # Polovina predikčních nástrojů a SwissProt se shodují
    elif flag == 1 and score == 2 :
        combnuc = combnuc + 1
        verdictstr = verdictstr + 'Combined nuclear'
        return verdictstr

    # Většina predikčních nástrojů se shoduje a SwissProt není k dispozici
    elif flag == 0 and score >= 3 :
        prednuc = prednuc + 1
        verdictstr = verdictstr + 'Predicted nuclear'
        return verdictstr

    # Většina predikčních nástrojů se shoduje ale SwissProt jadernou lokalizaci vylučuje
    elif flag == 2 and score >= 3 :
        disuni = disuni + 1
        verdictstr = verdictstr + 'Discrepancy - Uniprot'
        return verdictstr

    # Pouze SwissProt a zároveň jeden predikční nástroj ukazují na jadernou lokalizaci
    elif flag == 1 and score == 1 :
        dispred = dispred + 1
        verdictstr = verdictstr + 'Discrepancy - prediciton'
        return verdictstr

    # Výsledky nenaznačují jadernou lokalizaci
    else:
        return verdictstr
```

Zdrojový kód 5: Funkce judge() rozhodující o klasifikaci proteinu.

Jakmile jsou v daném cyklu shromážděna všechna potřebná data, jsou uložena do jednoho řádku výstupního souboru. Cyklus končí ve chvíli, kdy se ve vstupních

souborech již nenacházejí další data. Po ukončení cyklu modul zobrazí jednotlivé klasifikace a počet proteinů, které byly do dané třídy zařazeny. (Zdrojový kód 6)

```
# Otevření souboru
with open('FINAL_annotation.csv', mode='w', newline='') as BLAST_input:
    output_writer = csv.writer(BLAST_input, delimiter=';', quotechar='"', quoting=csv.QUOTE_MINIMAL)
    otherheader = ['ID', 'SAF', 'NSAF', 'NucPred Score', 'WegoLoc', 'NLS', 'Cello2GO', 'FINAL']
    finalheader = reduce(add,[otherheader[:1], shakerheader[:2], cutterheader[1:],
                              shakerheader[2:], otherheader[1:3], blastheader[:], otherheader[3:]])
    output_writer.writerow(finalheader) # Zápis záhlaví výstupního souboru
    maincount = 0
    iter = -1
    while True: # Hlavní cyklus závěrečné anotace
        result7 = cello2gosingle(cello2gofile) # Načtení výsledku CELLO2GO
        iter = iter + 1
        if result7: # Ověření existence výsledku
            try:
                # Načtení jednoho řádku ze všech vstupních souborů
                result1 = list(shakersingle(shakerfile))
                result2 = protcutsingle(proteincutterfile)
                result3 = next(blasttab)
                result4 = nucpredsingle(nucpredfile)
                result5 = wegolocsingle(wegolocfile)
                result6 = localizersingle(localizerfile)

                # Ověření, že všechny řádky obsahují výsledky pro stejnou sekvenci
                if result1[1] == result2[0] == result3[0] == result4[0] == result5[0] == result6[0] == result7[0]:
                    GOcheck = result3[1]                                # Ověření počtu terminů genové ontologie
                    if len(GOcheck) == 0:
                        verdict = judge(str(result3[11]+result3[13]), # Funkce pro rozhodnutí o pravděpodobnosti
                                        -1,                           # jaderné lokalizace proteinu v případě,
                                        result4[1],                   # že záznam neobsahuje termíny GO
                                        treshold_nuc,
                                        result5[1],
                                        result6[1],
                                        result7[1])
                    else:
                        verdict = judge(str(result3[11]+result3[13]), # Stejná funkce v případě, že je potřeba ověřit
                                        float(result3[9]),            # termíny GO
                                        result4[1],
                                        treshold_nuc,
                                        result5[1],
                                        result6[1],
                                        result7[1])

                    valarr = [SAFarr[iter], NSAFarr[iter], verdict]   # Seznam výsledků vypočítávaných modulem 3
                    resultarr = reduce(add, [result1[:3],             # Uložení hodnot a výsledků do jednoho seznamu
                                             result2[1:],
                                             result1[3:],
                                             valarr[:2],
                                             result3[:],
                                             result4[1:],
                                             result5[1:],
                                             result6[1:],
                                             result7[1:],
                                             valarr[3:]])
                    resultarr.append(verdict)
                    output_writer.writerow(resultarr)                 # Zápis výsledného seznamu do výstupního souboru
                    print("Sequence %s\t\tannotated" % (result1[1]))
                    maincount = maincount + 1
                else: # Neshoda identifikátorů sekvencí - přerušení funkce
                    print("ID Mismatch at index %s!\n" % (result1[1]))
                    break
            except Exception as e: # Selhání programu při pokusu o načtení dalšího řádku.
                print("Critical error during file handling!\n")
                break
        else: # Úspěšné ukončení anotace, tisk statistiky
            print("Full annotation finished! File saved as %s.\n\n" % (usrpath+'\FINAL_annotation.tsv'))
            "nsafcoef:\t\t\t\t\t%f\n"
            "Total sequences:\t\t\t\t\t%i\n"
            "Validated nuclear: \t\t\t\t\t%i (%% %i)\n"
            "Predicted nuclear: \t\t\t\t\t%i (%% %i)\n"
            "Combined nuclear: \t\t\t\t\t%i (%% %i)\n"
            "Uniprot discrepancies: \t\t\t\t\t%i (%% %i)\n"
            "Prediction discrepancies:\t\t\t\t\t%i (%% %i)\n" % (usrpath+'\FINAL_annotation.tsv', nsafcoef,
                                                                  maincount, validnuc, validperc, prednuc,
                                                                  predperc, combnuc, combperc, disuni,
                                                                  disuperc, dispred, dispperc))
```

Zdrojový kód 6: Část modulu 3 zodpovídající za vyhodnocení a zápis proteinů do výstupního souboru.

4.2 Výstupní data

Po dokončení vývoje a ladění programu byla pro ověření jeho funkčnosti použita vstupní data zmíněná v kapitole 3. Ta byla analyzována jak programem ProteinScout, tak i manuálně. Pro proteinový BLAST bylo zvoleno standardní nastavení s filtrováním na modelový organismus *Arabidopsis thaliana*. Minimální procentuální skóre pozitivity bylo nastaveno na 30 %, které odpovídá minimální hodnotě pozorované v manuálně anotovaném souboru. Nástroj WegoLoc byl nastaven na rostlinný Höglund dataset při prahu multiplex a e-value 1,0. Nástroj CELLO2GO byl nastaven na predikci eukaryontních jaderných proteinů při prahové hodnotě e-value 0,001. Prahová hodnota NucPred byla nastavena na 0,5. Nástroj LOCALIZER byl nastaven na predikci pro data typu Full plants sequences.

Výstupní tabulka byla následně porovnána s dříve vytvořenou, manuálně anotovanou tabulkou vycházející ze stejných vstupních dat. Následně byly z obou vytvořených výstupů vytříděny položky patřící k proteinovým kontaminantům (jmenovitě se jedná o následující záznamy: ALBU_BOVIN, CAS1_BOVIN, K1C10_HUMAN, K1C15_SHEEP, K1C9_HUMAN, K22E_HUMAN, K2C1_HUMAN, REF_HEVBR, RS27A_HUMAN a TRYP_PIG). Rozdíly v četnosti přiřazení jednotlivých hodnocení jsou porovnány v tabulce X.

Tabulka x: Srovnání četnosti jednotlivých hodnocení v manuálně a automaticky anotovaných výstupních souborech. V první závorce je uveden procentuální podíl četnosti ze všech 2711 záznamů

Klasifikace proteinu	Četnost v manuálně anotovaném souboru	Četnost v souboru anotovaném programem ProteinScout
Validované jaderné proteiny (VN)	279 (10,29 %)	223 (8,23 %)
Předpovězené jaderné proteiny (PN)	148 (5,46 %)	140 (5,16 %)
Neshoda – negativní Swiss-Prot (DS)	96 (3,54 %)	110 (4,06 %)
Neshoda – negativní predikce (DP)	215 (7,93 %)	160 (5,90 %)
Kombinovaná data (CN)	124 (4,57 %)	107 (3,95 %)
Nejaderné proteiny	1849 (68,20 %)	1971 (72,70 %)
Vyřazeno (kontaminanty)	10	10

Logickým důsledkem přísnějšího nastavení prahové hodnoty e-value pro nástroj BLAST by bylo snížení množství záznamů, ke kterým je možné v databázi Swiss-Prot nalézt homologní protein (což znamená snížení počtu VN, DP a CN klasifikací, jelikož ty se odvíjí od nalezení jaderného i nejaderného homologu ve Swiss-Prot). Mělo by také dojít k nárůstu počtu záznamů, ve kterých není homolog ze Swiss-Prot k dispozici (tzn. zvýšení počtu PN a nejaderných proteinů). Jelikož jsou tyto předpovězené trendy ve výstupních datech skutečně pozorovány (pokles VN o 2,06 % celku, pokles DP o 2,03 %, pokles CN o 0,62 %, nárůst počtu nejaderných proteinů o 4,50 %), lze předpokládat, že popsaná příčina alespoň částečně vysvětluje skutečně pozorované rozdíly. Anomálií v tomto předpokladu je skutečnost, že počet hodnocení PN se oproti očekávání nezvýšil, naopak klesnul o 0,30 %. Možným vysvětlením tohoto jevu je přiřazení méně jaderných predikcí nástrojem CELLO2GO v automaticky anotovaném souboru (812) než v souboru s manuálními anotacemi (1084) navzdory tomu, že parametry se kterými byl nástroj spuštěn byly v obou případech shodné (predikce pouze eukaryontních jaderných proteinů, prahová hodnota e-value 0,001, vypnuta možnost TrEMBL usage).

Mimo toto srovnání stojí proteiny označené jako chromozomální (Possibly chromosomal). Toto hodnocení není uváděno samostatně, ale je přiloženo k primární klasifikaci (např. ‚Possibly chromosomal – Validated nuclear') a označuje proteiny, u nichž jsou uvedeny termíny GO spojené s chromozomální funkcí či lokalizací. Takto označené proteiny nebyly v manuálně anotovaném souboru nijak zvýrazněny, nicméně představují specifický předmět zájmu z hlediska následné práce s anotovanými daty. Celkem bylo takovýchto proteinů ve vstupním souboru nalezeno 64. Jejich četnost je uvedena v tabulce níže (Tab. X)

Tabulka x. Četnost proteinů spojených s chromozomální lokalizací či funkcí

Primární klasifikace nalezených chrom. proteinů	Počet identifikovaných proteinů
Chromozomální VN	54
Chromozomální DP	3
Chromozomální CP	7

5 ZÁVĚR

Dle zadaných specifikací byl vytvořen návrh programu určeného k anotaci nepopsaných proteinových sekvencí získaných metodou tandemové hmotnostní spektrometrie. Na základě tohoto návrhu byl v jazyce Python vytvořen program ProteinScout, který umožňuje automatické seřazení vstupních dat a jejich anotaci pomocí nalezených homologních proteinů v databázi Swiss-Prot. Dále umožňuje spojení dílčích výsledků do závěrečné tabulky, kvantifikaci proteinů v původním vzorku metodou výpočtu NSAF a označení proteinů s pravděpodobně jadernou lokalizací. Tento program byl otestován na manuálně anotovaném souboru proteinů a výsledky byly porovnány. Odchylky v počtu jednotlivých klasifikací proteinů mezi manuálně a programem anotovanými soubory byly popsány a byla formulována hypotéza, která tuto odchylku vysvětluje. Závěrem tohoto srovnání je, že výstup vytvořeného programu je prakticky použitelný pro anotaci nepopsaných proteinových sekvencí a umožňuje výrazné zrychlení oproti manuální anotaci dat.

6 LITERATURA

Aebersold, R., & Mann, M. (2003). Mass spectrometry-based proteomics. Nature, 422(6928), 198–207. doi:10.1038/nature01511

Brameier M, Krings A, Maccallum RM NucPred - Predicting Nuclear Localization of Proteins. Bioinformatics, 2007. PubMed id: 17332022

ALTSCHUL, S; GISH, W; MILLER, W; MYERS, E; LIPMAN, D. Basic local alignment search tool. *Journal of Molecular Biology*. 1990, s. 403–410

BUSCH, Kenneth L. Units in Mass Spectrometry. *Current Trends in Mass Spectrometry* [online]. 2003, **18**(5), 32-34 [cit. 2019-05-20].

CHEN, Lee Chuin, Md. Matiur RAHMAN a Kenzo HIRAOKA. Super-Atmospheric Pressure Ion Sources: Application and Coupling to API Mass Spectrometer. *Mass Spectrometry* [online]. 2014, **3**(Special_Issue), S0024-S0024 [cit. 2019-05-20]. DOI: 10.5702/massspectrometry.S0024. ISSN 2187-137X.

Cormen, Thomas H.; Leiserson, Charles E.; Rivest, Ronald L.; Stein, Clifford (2009). *Introduction to Algorithms* (3rd ed.). Massachusetts Institute of Technology. pp. 253–280. ISBN 978-0-262-03384-8.

DASS, Chhabil. Modes of Ionization. *Fundamentals of Contemporary Mass Spectrometry* [online]. Hoboken, NJ, USA: John Wiley & Sons, 2007, s. 15-65 DOI: 10.1002/9780470118498.ch2. ISBN 9780470118498.

FENN, J., M MANN, C. MENG, S. WONG a C. WHITEHOUSE. Electrospray ionization for mass spectrometry of large biomolecules. *Science* [online]. 1989, **246**(4926), 64-71 [cit. 2019-05-20]. DOI: 10.1126/science.2675315. ISSN 0036-8075.

FRIEDECKÝ, D a K LEMR. Úvod do hmotnostní spektrometrie. *Klinická biochemie a metabolismus* [online]. 2012, **20**(41), 152-157 [cit. 2019-05-20].

GROSS, Jürgen H. *Mass spectrometry: a textbook*. New York: Springer, c2004. ISBN 3-540-40739-1.

Han, X., Aslanian, A., & Yates, J. R. (2008). *Mass spectrometry for proteomics. Current Opinion in Chemical Biology, 12(5), 483–490.* doi:10.1016/j.cbpa.2008.07.024

Hoffmann, E., Stroobant, V. (2007): Mass spectrometry: Principles and Applications.Third Edition.John Wiley and Sons Ltd, England.

HUNT, Donald F., Charles N. MCEWEN a T. Michael. HARVEY. Positive and negative chemical ionization mass spectrometry using a Townsend discharge ion source. *Analytical Chemistry* [online]. 2002, **47**(11), 1730-1734 [cit. 2019-05-20]. DOI: 10.1021/ac60361a011. ISSN 0003-2700.

LANE, C. S. Mass spectrometry-based proteomics in the life sciences. *CMLS Cellular and Molecular Life Sciences* [online]. 2005, **62**(7-8), 848-869 [cit. 2019-05-20]. DOI: 10.1007/s00018-005-5006-6. ISSN 1420-682X.

LEMAIRE, R., M. WISZTORSKI, A. DESMONS, J. C. TABET, R. DAY, M. SALZET a I. FOURNIER. MALDI-MS Direct Tissue Analysis of Proteins: Improving Signal Sensitivity Using Organic Treatments. *Analytical Chemistry* [online]. 2006, **78**(20), 7145-7153 [cit. 2019-05-20]. DOI: 10.1021/ac060565z. ISSN 0003-2700.

Lipman D.J., Pearson W.R. (1985): Rapid and sensitive protein similarity searches. *Science*. **227** (4693): 1435–41.

Mann, M., Hendrickson, R. C., Pandey, A. (2001): Analysis of proteins and proteomes by mass spectrometry. Annual review of biochemistry 70(1): 437-473.

Sang-Mun Chi, Dougu Nam, WegoLoc: accurate prediction of protein subcellular localization using weighted gene ontology terms," Bioinformatics, Vol. 28, No. 7, pp. 1028-1030, April. 2012.

Sperschneider, J. *et al*. LOCALIZER: subcellular localization prediction of both plant and effector proteins in the plant cell. *Sci. Rep.* **7**, 44598; doi: 10.1038/srep44598 (2017).

TAKATS, Z. Mass Spectrometry Sampling Under Ambient Conditions with Desorption Electrospray Ionization. *Science* [online]. 2004, **306**(5695), 471-473 [cit. 2019-05-20]. DOI: 10.1126/science.1104404. ISSN 0036-8075. Dostupné z: http://www.sciencemag.org/cgi/doi/10.1126/science.1104404

Yu CS, Cheng CW, Su WC, Chang KC, Huang SW, Hwang JK, and Lu CH*: CELLO2GO: A Web Server for Protein subCELlular LOcalization Prediction with Functional Gene Ontology Annotation. PLoS ONE 2014, 9(6): e99368.

URL1: https://www.ncbi.nlm.nih.gov/Web/Newsltr/Summer99/qblast.html